# Contrôle physique de mouvement de personnages virtuels en environnement complexe

Samuel Carensac

▶ **To cite this version:**

Samuel Carensac. Contrôle physique de mouvement de personnages virtuels en environnement complexe. Synthèse d'image et réalité virtuelle [cs.GR]. Université de Lyon, 2019. Français. NNT : 2019LYSEI037 . tel-02484943v2

**HAL Id: tel-02484943**
**https://theses.hal.science/tel-02484943v2**

Submitted on 28 Feb 2020

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

N°d'ordre NNT : 2019LYSEI037

**THESE de DOCTORAT DE L'UNIVERSITE DE LYON**
opérée au sein de
**INSA Lyon**

**Ecole Doctorale** N° 512
**Ecole doctorale InfoMaths**

**Spécialité/ discipline de doctorat** :
Informatique

Soutenue publiquement le 05/07/2019, par :
**Samuel Carensac**

---

# Contrôle physique de mouvement de personnages virtuels en environnement complexe

---

Devant le jury composé de :

MULTON, Franck Professeur des Universités Université Rennes 2 Rapporteur
REVERET, Lionel Chargé de Recherche HDR INRIA Rhône-Alpes Rapporteur
CANI, Marie-Paule Professeure des Universités Ecole Polytechnique Examinateur
GIBET, Sylvie Professeure des Universités Université de Bretagne Sud Examinateur

BASKURT, Atilla Professeur des Universités INSA-LYON Directeur de thèse
BOUAKAZ, Saïda Professeure des Universités Université Claude Bernard Lyon 1 Co-directrice de thèse
PRONOST, Nicolas Maître de Conférence Université Claude Bernard Lyon 1 Co-encadrant de thèse

Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

# Département FEDORA – INSA Lyon - Ecoles Doctorales – Quinquennal 2016-2020

| SIGLE | ECOLE DOCTORALE | NOM ET COORDONNEES DU RESPONSABLE |
|---|---|---|
| CHIMIE | **CHIMIE DE LYON**<br>**http://www.edchimie-lyon.fr**<br>Sec. : Renée EL MELHEM<br>Bât. Blaise PASCAL, 3e étage<br>**secretariat@edchimie-lyon.fr**<br>INSA : R. GOURDON | M. Stéphane DANIELE<br>Institut de recherches sur la catalyse et l'environnement de Lyon<br>IRCELYON-UMR 5256<br>Équipe CDFA<br>2 Avenue Albert EINSTEIN<br>69 626 Villeurbanne CEDEX<br>**directeur@edchimie-lyon.fr** |
| E.E.A. | **ÉLECTRONIQUE, ÉLECTROTECHNIQUE, AUTOMATIQUE**<br>**http://edeea.ec-lyon.fr**<br>Sec. : M.C. HAVGOUDOUKIAN<br>**ecole-doctorale.eea@ec-lyon.fr** | M. Gérard SCORLETTI<br>École Centrale de Lyon<br>36 Avenue Guy DE COLLONGUE<br>69 134 Écully<br>Tél : 04.72.18.60.97 Fax 04.78.43.37.17<br>**gerard.scorletti@ec-lyon.fr** |
| E2M2 | **ÉVOLUTION, ÉCOSYSTÈME, MICROBIOLOGIE, MODÉLISATION**<br>**http://e2m2.universite-lyon.fr**<br>Sec. : Sylvie ROBERJOT<br>Bât. Atrium, UCB Lyon 1<br>Tél : 04.72.44.83.62<br>INSA : H. CHARLES<br>**secretariat.e2m2@univ-lyon1.fr** | M. Philippe NORMAND<br>UMR 5557 Lab. d'Ecologie Microbienne<br>Université Claude Bernard Lyon 1<br>Bâtiment Mendel<br>43, boulevard du 11 Novembre 1918<br>69 622 Villeurbanne CEDEX<br>**philippe.normand@univ-lyon1.fr** |
| EDISS | **INTERDISCIPLINAIRE SCIENCES-SANTÉ**<br>**http://www.ediss-lyon.fr**<br>Sec. : Sylvie ROBERJOT<br>Bât. Atrium, UCB Lyon 1<br>Tél : 04.72.44.83.62<br>INSA : M. LAGARDE<br>**secretariat.ediss@univ-lyon1.fr** | Mme Emmanuelle CANET-SOULAS<br>INSERM U1060, CarMeN lab, Univ. Lyon 1<br>Bâtiment IMBL<br>11 Avenue Jean CAPELLE INSA de Lyon<br>69 621 Villeurbanne<br>Tél : 04.72.68.49.09 Fax : 04.72.68.49.16<br>**emmanuelle.canet@univ-lyon1.fr** |
| INFOMATHS | **INFORMATIQUE ET MATHÉMATIQUES**<br>**http://edinfomaths.universite-lyon.fr**<br>Sec. : Renée EL MELHEM<br>Bât. Blaise PASCAL, 3e étage<br>Tél : 04.72.43.80.46<br>**infomaths@univ-lyon1.fr** | M. Luca ZAMBONI<br>Bât. Braconnier<br>43 Boulevard du 11 novembre 1918<br>69 622 Villeurbanne CEDEX<br>Tél : 04.26.23.45.52<br>**zamboni@maths.univ-lyon1.fr** |
| Matériaux | **MATÉRIAUX DE LYON**<br>**http://ed34.universite-lyon.fr**<br>Sec. : Stéphanie CAUVIN<br>Tél : 04.72.43.71.70<br>Bât. Direction<br>**ed.materiaux@insa-lyon.fr** | M. Jean-Yves BUFFIÈRE<br>INSA de Lyon<br>MATEIS - Bât. Saint-Exupéry<br>7 Avenue Jean CAPELLE<br>69 621 Villeurbanne CEDEX<br>Tél : 04.72.43.71.70 Fax : 04.72.43.85.28<br>**jean-yves.buffiere@insa-lyon.fr** |
| MEGA | **MÉCANIQUE, ÉNERGÉTIQUE, GÉNIE CIVIL, ACOUSTIQUE**<br>**http://edmega.universite-lyon.fr**<br>Sec. : Stéphanie CAUVIN<br>Tél : 04.72.43.71.70<br>Bât. Direction<br>**mega@insa-lyon.fr** | M. Jocelyn BONJOUR<br>INSA de Lyon<br>Laboratoire CETHIL<br>Bâtiment Sadi-Carnot<br>9, rue de la Physique<br>69 621 Villeurbanne CEDEX<br>**jocelyn.bonjour@insa-lyon.fr** |
| ScSo | **ScSo***<br>**http://ed483.univ-lyon2.fr**<br>Sec. : Véronique GUICHARD<br>INSA : J.Y. TOUSSAINT<br>Tél : 04.78.69.72.76<br>**veronique.cervantes@univ-lyon2.fr** | M. Christian MONTES<br>Université Lyon 2<br>86 Rue Pasteur<br>69 365 Lyon CEDEX 07<br>**christian.montes@univ-lyon2.fr** |

*ScSo : Histoire, Géographie, Aménagement, Urbanisme, Archéologie, Science politique, Sociologie, Anthropologie

Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

# Résumé

Cette thèse traite de l'animation de personnages virtuels composés de corps rigides reliés par des articulations et contrôlés par des interactions physiques (forces et moments). Le contrôleur est le système qui calcule dynamiquement ces interactions. Notre objectif est d'étudier et de réaliser un contrôleur pour la simulation de mouvements d'un personnage en interaction avec un fluide.

La complexité du comportement de tels milieux ne permet pas de prédire les interactions entre le personnage et le fluide. Il en découle que le contrôleur proposé doit être capable de réagir à celles-ci. Nous avons focalisé nos travaux sur la conception d'un contrôleur de type SIMBICON capable de s'adapter aux perturbations apportées par la présence d'un fluide simulé physiquement. Ce choix est motivé par notre contrôleur précédent qui proposait un contrôleur en interaction avec un fluide représenté à travers l'utilisation de formule de dynamique des fluides simples. L'utilisation d'une véritable simulation physique du fluide nous permettrait d'améliorer le réalisme physique de la simulation en prenant en compte l'impact du déplacement du personnage sur le fluide.

Ayant pour objectif un contrôleur interactif nous nous sommes focalisés sur deux axes principaux. Le premier est la conception d'un contrôleur capable de supporter des fréquences de simulation faibles tout en conservant la vitesse de calcul apporté par l'utilisation du modèle SIMBICON. Nous proposons de réduire les instabilités introduites par l'utilisation de fréquences de simulation faibles par un système de feedback utilisant une optimisation en ligne permettant d'obtenir une meilleure stabilité des contacts. Ce système, associé à une étude des paramètres du système en fonction de la fréquence de simulation, nous a permis de proposer un contrôleur capable de supporter des fréquences de simulation allant jusqu'à 225Hz. Le second axe de recherche visait à proposer une implémentation entièrement GPU et interactive d'une simulation lagrangienne de fluide. Nous avons étudié l'impact sur les performances de notre implémentation GPU de plusieurs optimisations proposé par des travaux proposant des implémentations parallèles CPU. Nous proposons également une solution permettant de déplacer la zone de fluide simulé en cours de simulation pour limiter l'espace de simulation du fluide à la proximité immédiate du personnage au cours de son déplacement pour assurer une simulation du fluide en temps interactif.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

# Abstract

This thesis deals with the animation of virtual characters composed of rigid bodies linked by joints and controlled by physic interactions (forces and torques). The controller is the system that dynamically calculates these interactions. Our goal is to study and create a controller that is able to control the character interacting with a fluid.

The complexity of the behavior of such environment renders predicting the interactions between the fluid and the character impossible. Therefore, the controller must be able to react to such interactions. We have focused our works on the conception of a SIMBICON typed controller that is able to handle the perturbations caused by the physically simulated fluid. This choice is motivated by our previous controller that was able to handle the interactions with a simplified fluid based on simplified fluid dynamics equations. The use of the complete fluid dynamic equations should allow for a higher realism by taking into account to impact of the character motion on the fluid.

Since our objective is to obtain an interactive controller, we focus our works on two main axes. The first one is the conception of a controller able to handle low simulation frequencies while keeping the high computation speed brought by the use of the SIMBICON framework. We propose to use a feedback system using an online optimization to reduce the instabilities caused by the of the low simulation frequencies. On top of this system, we study the evolution of the value of the PD-controllers depending on the simulation frequency to be able to propose a controller able to handle simulation frequencies as low as 225Hz. The second research axis aims to conceive a fully GPU implementation of a Lagrangian fluid simulation. We study the impact of various optimization proposed by previous work on our GPU implementation. We also present a system allowing us to move the simulation area of the fluid to be able to keep the character centered in the fluid simulation. This system allows us to only use a small area for the fluid simulation allowing us to propose a system showing interactive execution times.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

# Table des matières


**1 Introduction** **1**

**2 État de l'art : Contrôle de personnages virtuels simulés** **7**
2.1 Simulation physique de corps rigides . . . 7
2.2 Contrôle basé sur la physique de personnages virtuels . . . 9
2.3 Types de contrôleur basé sur la physique . . . 11
2.4 SIMBICON et extensions . . . 17
2.5 Contrôle de personnage virtuel à basse fréquence . . . 25
2.6 Bilan . . . 27

**3 État de l'art : Simulation de fluide** **29**
3.1 Modèles de fluide . . . 30
3.2 Smoothed Particles Hydrodynamics . . . 32
3.3 SPH sur le GPU . . . 37
3.4 Bilan . . . 38

**4 Contrôle à basse fréquence dans l'espace des articulations** **41**
4.1 Vue d'ensemble . . . 41
4.2 Contrôle de la vitesse et de la direction du mouvement . . . 42
4.3 Contrôle à basse fréquence . . . 51
4.4 Résultats . . . 62
4.5 Bilan . . . 76

**5 Simulation lagrangienne de fluide sur GPU** **77**
5.1 Algorithme DFSPH . . . 77
5.2 Structure de données . . . 80
5.3 DFSPH sur GPU . . . 88
5.4 Fenêtre dynamique de simulation pour fluide lagrangien . . . 91
5.5 Expérimentations . . . 101
5.6 Discussions . . . 113
5.7 Bilan . . . 114

**6 Simulation de personnage en interaction avec un fluide** **117**
6.1 Estimation du fluide par modèle simplifié . . . 118
6.2 Simulation de fluide lagrangienne en interaction avec des objets . . . 119
6.3 Résultats . . . 126
6.4 Limitations . . . 139
6.5 Bilan . . . 140

**7 Conclusion** **143**

**Bibliographie** **148**




Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

A Comparaison forces pour une jambe simulée 157



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

# 1 Introduction

L'animation de personnages virtuels est un domaine qui est de nos jours au centre de deux industries importantes : l'industrie vidéoludique et l'industrie cinématographique. La qualité d'une animation est très souvent étroitement jugée au vu de son réalisme. Ce critère est capital pour la réalisation de films où les animations sont intégrées dans des scènes en partie filmées dans un décor réel en particulier lorsque les personnages animés sont des personnages humains. En effet, lorsqu'il est question de projet animé par ordinateur, un manque de réalisme dans le mouvement des personnages ou leurs expressions est très souvent à l'origine de critiques des spectateurs (ex : Berserk 2016). À l'inverse, une animation réaliste permet d'intégrer des personnages irréels avec des acteurs filmés par des caméras sans nuire à son appréciation par les spectateurs comme le montre le succès des nombres films récents de super héros.

Les méthodes d'animation peuvent être classées selon deux approches : l'animation cinématique et l'animation basée sur la physique. L'animation cinématique est l'approche la plus répandue. Le personnage est généralement représenté par des parties rigides reliées entre elles par des articulations permettant un à trois degrés de liberté en rotation. Les approches cinématiques consistent à modifier les orientations et les positions des articulations et des parties rigides au cours du temps pour modifier la pose du personnage. Il existe plusieurs approches possibles pour obtenir les poses successives du personnage. La création de ces poses à la main par un animateur permet d'obtenir un haut niveau de fidélité au mouvement voulu mais rend difficile la création de mouvements réalistes. À l'inverse, l'utilisation de systèmes de capture de mouvements sur des acteurs permet d'assurer un haut niveau de réalisme mais limite les animations possibles aux capacités de l'acteur. La limitation majeure de l'animation cinématique est que chaque animation est spécifique à un scénario précis (vitesse de déplacement, présence d'obstacles, ...). Un exemple simple est d'imaginer une animation où un objet entre en collision avec le personnage. Si la vitesse de l'objet et l'endroit de l'impact avec le personnage ne correspondent pas à ceux prévus par l'animateur, l'animation obtenue perdra de son réalisme physique. Si cette limitation peut être considérée comme négligeable pour la création de film car les animations sont créées pour des scènes spécifiques, ce n'est pas le cas dans l'industrie vidéoludique. Pour la création de contenu interactif, si le monde dans lequel se meut le personnage propose un nombre élevé d'éléments dont le comportement n'est pas connu à l'avance pouvant interagir avec ce personnage, l'approche basée sur la cinématique est limitée. En effet, si l'on veut pouvoir traiter des millions de situations possibles, il serait nécessaire de créer les millions d'animations pour gérer chaque interaction possible. Il est possible de modifier dynamiquement l'animation du personnage pour assurer que certaines règles plus ou moins complexes soient respectées ([Van+10]), par exemple modifier l'angle des genoux pour que les pieds touchent correctement le sol. Cependant, il est impossible de prévoir toutes les règles nécessaires pour permettre au personnage d'interagir avec tous les éléments qui possèdent un comportement non connu à l'avance présents dans un monde virtuel complexe. La seconde approche, l'animation basée sur la physique, offre une solution à la gestion des interactions avec l'environnement. Pour ce second type d'animation, le personnage est physiquement simulé à l'intérieur du monde virtuel et est animé par les lois de la physique par l'intermédiaire d'un moteur physique. Les parties rigides



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

du personnage deviennent des objets physiques pouvant entrer en collision avec les éléments de son environnement. La création de l'animation se fait par l'application d'interactions physiques (forces et moments), que l'on appelle actionneurs, sur les différentes parties du personnage. Cette approche possède deux avantages. Le premier est que, la modification de la pose du personnage étant faite par l'application de phénomènes physiques, le résultat assure forcément un niveau de réalisme physique parfait en vue des lois physiques utilisées. Le second avantage provient de l'intégration du personnage et d'un monde virtuel dans une simulation unique ce qui permet aux éléments du monde virtuel d'avoir un impact réaliste sur le personnage. Si nous reprenons l'exemple de la collision entre un objet et un personnage, une modification de l'endroit de l'impact induira différentes forces sur le personnage ce qui générera un nouveau mouvement sans spécifier des données d'entrée supplémentaires. Dans l'animation basée sur la physique, les interactions entre le personnage et le monde virtuel sont "innées". Cependant, l'intégration innée des interactions avec le monde virtuel est associée à un coût élevé : le contrôle du personnage est très complexe et doit être réalisé en parallèle de la simulation. Il est irréaliste d'espérer pouvoir spécifier les moments à appliquer pour toutes les configurations possibles (cela reviendrait aux mêmes limitations que l'animation cinématique). Pour déterminer les actionneurs à appliquer à chaque instant, il est nécessaire d'utiliser un contrôleur. Le contrôleur répertorie des règles qui calculent les actionneurs à appliquer sur les membres et articulations du personnage en fonction de l'état du personnage et du reste de la simulation. Créer un tel contrôleur est une tâche complexe du fait du grand nombre d'actionneurs nécessaires à chaque pas de temps. Suivant le nombre d'articulations présentes dans le modèle du personnage animé, le contrôleur peut être amené à calculer plus de 15 moments pour chaque pas de simulation. De plus, le moment appliqué sur une articulation peut avoir un impact sur le moment nécessaire pour d'autres articulations. Par exemple, un moment appliqué dans une articulation du dos du personnage aura une influence sur l'orientation de la tête du personnage et devra donc être pris en compte lors du calcul du moment à appliquer sur le cou du personnage. Ce phénomène associé à la complexité de la simulation physique du monde virtuel, rend difficile l'obtention de moments permettant d'obtenir exactement le résultat désiré avant leur application dans la simulation physique. Malgré le niveau de complexité élevé, le réalisme physique de ce type d'animation permet son utilisation pour de nouvelles tâches. En particulier l'intégration innée des interactions avec les éléments du monde virtuel rend l'animation basée sur la physique attractive pour toute animation ou le personnage doit manipuler des objets du monde virtuel. Ce type d'animation peut également être utilisé pour contrôler un objet présent dans le monde réel, par exemple pour animer un robot. De plus, l'animation basée sur la physique ouvre de nouvelles possibilités d'animation réaliste lorsque le personnage interagit avec des milieux complexes tels que des fluides ou des corps mous. De tels milieux sont hautement dynamiques et, par conséquent, sont difficiles voire impossibles à gérer avec une approche cinématique si le réalisme physique est un critère requis. Jusqu'à présent, le contrôle d'un personnage prenant en compte ces interactions complexes n'a que très peu été étudié. Les travaux considérant un personnage en interaction avec un fluide tentent généralement de simuler des mouvements de nage et rarement un déplacement bipède. La différence majeure entre ces deux cas d'application est que lors de la simulation d'un mouvement de nage le personnage est entièrement supporté par le fluide et les forces du fluide sont réparties sur la totalité des membres du personnage. Dans ce cas d'application, le moteur physique ne fait qu'appliquer les valeurs des actionneurs sur les parties du personnage. À l'opposé, dans un déplacement bipède le personnage est en support sur le sol. Par conséquent, en plus de devoir appliquer les valeurs des actionneurs, le moteur physique doit déterminer les points d'intersection entre les pieds du personnage et le sol pour calculer les forces à appliquer sur les pieds de manière à empêcher les intersection entre les pieds et le sol ce qui augmente la complexité des calculs nécessaires.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

Cette thèse s'intéresse au contrôle d'un personnage en interaction avec un fluide. Par conséquent, nous nous sommes non seulement intéressés au domaine de l'animation de personnages, mais également à celui de l'animation de fluides. Similairement à un personnage, l'animation d'un fluide peut être obtenue avec une approche cinématique ou bien par simulation. L'approche cinématique est celle généralement utilisée pour des volumes de liquide importants quand le contenu doit être interactif, particulièrement dans l'industrie vidéoludique. Dans cette approche, le fluide est simplement représenté par sa surface, généralement par un champ de hauteur qui évolue au cours du temps. L'évolution des hauteurs pour être spécifiée manuellement par un artiste ou bien être contrôle par une méthode d'animation procédurale. Bien que pratique pour représenter le mouvement de vagues dans des océans, le manque de réalisme de l'animation obtenue devient vite apparent pour des volumes de fluide plus faibles ou si le fluide est en contact avec des objets solides. L'animation de rivières par cette méthode produit généralement des animations où les turbulences dues aux interactions entre l'eau et les différents obstacles ne sont pas présentes. Pour obtenir une animation de haute qualité, il est préférable d'utiliser une approche basée sur la simulation. L'une des possibilités pour prendre en compte les propriétés physiques d'un fluide est de se baser sur les équations de Navier-Stokes. La propriété principale que les simulations de fluides visent à respecter est l'incompressibilité du fluide de manière à ce que le volume de fluide présent dans la simulation reste constant. La littérature propose deux représentations majeures utilisées pour simuler un fluide suivant si la représentation du fluide est fixée à l'espace de simulation ou au fluide simulé : la représentation eulérienne et la représentation lagrangienne ([TY09]). La représentation eulérienne se focalise à l'espace de simulation. Elle représente l'espace à simuler par une structure, généralement une grille, et déplace le volume de fluide à l'intérieur. À l'inverse, l'approche lagrangienne représente directement le fluide à simuler. Elle discrétise le fluide sous forme d'un ensemble de particules permettant ainsi d'être indépendant de l'espace dans lequel le fluide est simulé. La représentation eulérienne permet d'obtenir une meilleure qualité d'animation. Cependant, la représentation lagrangienne nécessite des temps d'exécution généralement bien moins importants permettant même la création de contenus interactifs tant que le nombre de particules simulées reste relativement faible (inférieur à 100000 particules).

Cette thèse fait suite aux travaux réalisés précédemment dans le cadre d'un master recherche. Ces travaux ont abouti en la conception d'un contrôleur capable de prendre en compte la présence d'un liquide à travers lequel le personnage se déplace [CPB15]. Cependant, des limitations importantes sont présentes dans ce contrôleur, non seulement sur le contrôle du personnage mais également sur l'intégration du fluide dans la simulation. Les limitations du contrôle du personnage sont liées à la fréquence à laquelle la simulation physique est réalisée. Ce contrôleur fonctionne avec une fréquence de simulation de 2000 Hz, c'est-à-dire que les calculs physiques sont réalisés toutes les 0.5ms. Bien que ce contrôleur produise une animation en temps réel, l'ajout d'éléments autres que le personnage dans la simulation conduit à une perte de l'interactivité de l'animation. Le problème de la fréquence de simulation est une difficulté connue dans le domaine de l'animation basée sur la physique de personnage virtuel. Une partie des algorithmes sur lesquels se basent les contrôleurs récents ont pour but de rendre possible l'utilisation de fréquences de simulation faibles pour avoir plus de ressources de calcul disponibles entre deux pas de la simulation physique. Une seconde limitation du contrôleur proposé dans nos travaux précédents vient de l'utilisation d'un modèle simplifié pour calculer l'impact du fluide sur le personnage. Ce modèle simplifié représente le fluide et son impact sur le personnage par les formules générales de dynamique des fluides : la poussée d'Archimède et l'équation de résistance du fluide au mouvement. L'application directe de ces formules ne prend pas en compte le fait que le personnage déplace le fluide au cours de son



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

mouvement, négligeant ainsi les courants induits dans le fluide. De plus, l'application directe de ces formules n'est pas possible car elles considèrent que l'objet étudié ne présente aucune rotation. Par conséquent, nous avons utilisé une version simplifiée de ces dernières diminuant d'autant plus le réalisme physique du résultat. Cependant, il n'existe, à notre connaissance, aucun autre contrôleur spécialisé dans la gestion des interactions avec un fluide lors d'un mouvement bipède. Par conséquent, nous ne disposions pas de comparaison possible avec des travaux existants.

Cette thèse a pour objectif de proposer un nouveau contrôleur capable de supporter les interactions avec un fluide simulé physiquement. Notre objectif est de proposer un système d'animation, qui offre un résultat réaliste et qui soit au moins interactif, en se rapprochant autant que possible d'une animation temps réel. Pour obtenir ce nouveau contrôleur, cette thèse s'organise selon les deux objectifs suivants :

— **La conception d'un simulateur de fluide interactif.** La difficulté principale avec la simulation de fluide est que celle-ci est généralement utilisée dans le cadre de la réalisation d'animations pré calculées. Par conséquent, les systèmes existants (Blender, Maya, ...) se focalisent plus sur la qualité du résultat obtenu que la vitesse de calcul. Nous avons pour objectif de proposer un simulateur de fluide physiquement réaliste permettant d'obtenir les meilleures performances possibles.

— **La conception d'un contrôleur de personnage à basse fréquence de simulation.** Comme présenté précédemment, l'avantage des faibles fréquences de simulation est de proposer de plus grands intervalles entre deux pas de simulation. Bien qu'il existe des contrôleurs capables de supporter des fréquences de simulation très faibles (jusqu'à 20Hz), ceux-ci utilisent la totalité du temps de calcul pour réaliser des calculs complexes assurant leur fonctionnement. Notre objectif n'est pas d'obtenir un contrôleur pouvant fonctionner avec des fréquences aussi extrêmes mais de proposer un contrôleur fonctionnant à des fréquences de simulation aussi basses que possibles sans avoir à utiliser de méthodes utilisant tout le temps de calcul ainsi libéré.

Le manuscrit est structuré comme suit. Les deux premiers chapitres de ce manuscrit de thèse sont consacrée à l'état de l'art relatif à notre problématique de recherche. Dans le chapitre 2, nous présentons les différentes approches relatives au contrôle d'un personnage simulé physiquement. Le chapitre suivant (chapitre 3) est consacré aux différents modèles utilisés pour la simulation de fluide.

Nous présentons ensuite nos différentes contributions. Dans le chapitre 4, nous proposons un nouveau contrôleur de personnage basé sur la physique. En partant d'un premier modèle de contrôleur que nous avons proposé dans le cadre du stage de master, nous apportons diverses améliorations aux systèmes de contrôles. Sur ce volet, notre principal apport est la possibilité d'utilisation de fréquences de simulation faibles à travers deux contributions principales : une étude des paramètres du contrôleur et un nouveau système d'optimisation en ligne améliorant la stabilité des contacts entre le personnage et le sol. Conscient que la validité d'un modèle peut être renforcée par une implémentation efficace, le chapitre 5 présente une implémentation sur carte graphique d'un simulateur de fluide lagrangien. En plus de l'étude de diverses optimisations proposées dans les publications du domaine pour vérifier leur applicabilité pour une implémentation GPU, ce chapitre présente une nouvelle méthode de fenêtre de déplacement dynamique pour les simulations de fluide lagrangienne. Enfin, dans le chapitre 6 nous comparons les résultats obtenue par notre nouveau contrôleur à ceux du contrôleur utilisant un modèle de fluide simplifié.


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

Pour conclure cette thèse, nous présentons un bilan de notre travail en mettant en évidence ses avantages et ses limites, tout en soulevant des perspectives pour des travaux futurs.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

# 2 État de l'art : Contrôle de personnages virtuels simulés

## 2.1 Simulation physique de corps rigides

Le contrôle d'un personnage, dans le cadre d'une animation basée sur la physique, est réalisé à travers l'application d'interactions physiques telles que des moments ou des forces. L'animation d'un personnage au cours du temps repose sur un moteur physique qui, à partir de l'état actuel du personnage ainsi que des forces et des moments à appliquer, calcule la position suivante du personnage. Suivant l'application on peut avoir recours à différents modèles pour représenter les éléments du monde réel. Ces modèles peuvent concerner la forme, l'aspect visuel, les propriétés physiques, etc. Suivant le modèle adopté, différentes méthodes permettent de gérer les interactions entre ces différents éléments.

Pour le contrôle de personnages virtuels, l'approche la plus couramment utilisée est de représenter le personnage par des corps rigides articulés. Classiquement, comme rappelé par Bender et al. [BET14], la simulation de corps rigides est composée de trois étapes principales :

— **Détection des collisions :** Le système détecte les points d'intersection ou de collisions entre les différents éléments présents dans la simulation.
— **Modélisation mathématique :** À partir de ces informations, un système d'équations représentant les lois physiques nécessaires à la gestion de cette simulation est construit. Ces équations sont généralement représentées sous la forme d'un problème de complémentarité linéaire (LCP) [CPS92].
— **Résolution du système :** Le LCP obtenu est ensuite résolu à l'aide d'une méthode adaptée en fonction des restrictions sur les ressources de calculs disponibles et la précision désirée.

Un état de l'art détaillé des différentes représentations des lois physiques et des méthodes de résolution du système d'équations est présenté par Bender et al. [BET14].

Dans le cas d'un personnage bipède, seuls les pieds du personnage sont en contact avec le sol. Le poids du personnage est donc réparti sur une faible surface de contact avec le sol. Suivant la méthode de gestion des contacts choisie, les forces appliquées entre le pied et le sol seront plus ou moins stables. Celle-ci a donc un impact sur le contrôle du personnage. En pratique, deux types de méthodes sont, en général, utilisées pour la gestion des contacts entre des corps rigides :

— **Contacts sans intersection :** cette approche considère qu'il ne doit pas être possible d'observer d'intersection entre deux objets. Pour obtenir ce résultat, le moteur physique résout un système d'équations permettant de déterminer les forces à appliquer au niveau des points d'intersection pour résoudre les intersections en un seul pas de simulation. Ce type de modèle est appelé *hard model* ou *nonsmooth model*. Le principal avantage



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

est l'obtention de résultat visuellement cohérent lorsque l'on représente des matériaux durs (verres, roche, ...). Cependant, les contraintes ajoutées au système, augmentent énormément sa difficulté de résolution en raison de discontinuités provoquées par ces contraintes. Leine et al. ainsi que Bender et al. [LN04 ; BET14] font état des contacts discontinus sur plusieurs pas de temps consécutifs.

— **Contacts avec intersection :** cette approche applique une force dépendant de la profondeur de l'intersection entre les deux corps pour provoquer un déplacement diminuant la profondeur de l'intersection. Ce type de modèle est appelé *smooth model*. Une solution intuitive est de mettre en place un ressort virtuel au niveau des points de contact entre les deux corps rigides [MW88]. Récemment, Todorov et al. [Tod14] ont proposé une solution permettant d'améliorer la stabilité physique de leur simulation, en particulier des intersections entre les corps rigides simulés. Bien que la présence d'intersections entre les corps rigides détériore le réalisme visuel de la simulation obtenue, cette d'approche permet d'obtenir une simulation plus robuste et rapide à calculer.

Une approche hybride peut être employée pour bénéficier d'un résultat visuellement sans intersection sans être limité par la discontinuité des contacts. Cette approche consiste à enrichir les corps rigides d'une fine épaisseur utilisant un *smooth model* à leur surface. Cette approche est celle choisie par les moteurs physiques les plus couramment utilisés (ODE, Bullet, PhysX). Quant au moteur physique MuJoCo [TET12], il s'appuie sur une implémentation de l'algorithme proposé par Todorov et al. [Tod14] mentionné ci-dessus. MuJoCo a été utilisé dans des travaux récents pour de simuler des mouvements complexes tels que la danse en temps interactif. Dans [Han+16 ; HNS18], les auteurs soutiennent que MuJoCo leur assure une meilleure stabilité des contacts permettant l'utilisation de fréquences de simulation plus faibles. Plusieurs études comparant ces moteurs physiques ont été publiées. Celles-ci s'intéressent généralement aux différences de performance et de stabilité entre les moteurs physiques classiques (ODE, PhysX, Bullet, ...) [ETT15 ; BB07 ; GY11]. Ces travaux relèvent que Bullet et PhysX sont capables de supporter des fréquences de simulation plus faibles qu'ODE avant que la simulation physique n'échoue pour des cas simples. Dans le cas de corps articulés complexes, la stabilité de ces trois moteurs physiques est équivalente [BB07 ; GY11]. Erez et al. [ETT15] compare MuJoCo aux moteurs physiques classiques et conclu que MuJoCo permet l'utilisation de fréquences de simulation plus faibles et présente des temps d'exécution plus faibles que tous les autres moteurs physiques examinés. Les auteurs font remarquer que MuJoCo est spécialisé dans la simulation de corps rigides articulés et que les performances chutent dans le cas de mondes virtuels contenant de nombreux corps rigides indépendants. Cette étude souligne également que parmi les moteurs physiques classiques, ODE offre de meilleures performances en temps d'exécution dans la majeure partie des simulations étudiées. Ivaldi et al. [IPN14] ont réalisé une étude répertoriant les habitudes d'utilisation des moteurs physiques par la communauté scientifique du domaine de la robotique. Leurs conclusions montrent qu'ODE est le moteur physique classique le plus fréquemment utilisé même si le taux d'évaluation global par les utilisateurs n'est que moyen.

Il existe d'autres approches pour la création de moteurs physiques, comme le *position based dynamics* [BMM15], mais, à notre connaissance, ceux-ci n'ont jamais été utilisés pour l'animation de personnages virtuels basée sur la physique.

## Conclusion

Dans la table 2.1, nous présentons un rapide récapitulatif des caractéristiques des différentes méthodes de simulation physique de corps rigides.


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

| | Hard model | Moteur hybride | Smooth model |
|---|---|---|---|
| Présence d'intersection | ++ | - | - |
| Stabilité | - - | 0 | ++ |
| Temps de calcul | - | + | + |
| Disponibilité | + | ++ | - - |

**Tableau 2.1.:** tableau récapitulatif des avantages et inconvénients des différents types de méthodes de simulation. Notre échelle d'évaluation va de –, indicatif des fortes lacunes, à ++, indicatif de très hautes performances pour la caractéristique considérée.

Le niveau d'instabilité des moteurs physiques proposant un *hard model* pur rend leur utilisation difficile pour le contrôle de personnages virtuels. De cette étude, le moteur MuJoCo semble offrir de plus de stabilité. Cependant, il s'agit d'un moteur propriétaire ce qui limite d'autant plus les tentatives d'applications des contrôleurs reposant sur celui-ci. De plus, le moteur MuJoCo n'a été rendu disponible que récemment. Dans le cadre des travaux réalisés dans cette thèse, une implémentation a été initialement été réalisée pour chacun des deux moteurs ODE (*Open Dynamic Engine*) et PhysX. Ayant constaté que les résultats obtenus ne montraient pas de différences majeures, nous nous sommes orientés définitivement vers ODE. Ce choix a été motivé par le fait qu'ODE a été utilisé dans de nombreux travaux scientifiques [IPN14]. ODE a également l'avantage d'être open-source ce qui apporte plus de flexibilité lors de l'implémentation.

## 2.2 Contrôle basé sur la physique de personnages virtuels

Comme évoqué précédemment, pour pouvoir réaliser une animation basée sur la physique, le personnage virtuel doit être représenté dans le monde physique. Le personnage est composé d'un ensemble de corps rigides représentant les différents membres du personnage. Chacun est associé à une forme géométrique simple (capsule, sphère ou pavé) permettant de déterminer les points de collisions. Les parties du personnage sont reliées par des articulations pouvant avoir de un à trois degrés de libertés en rotation (figure 2.1b). Il est à noter qu'une articulation peut également être contrainte pour limiter l'amplitude maximale du moment appliqué, la vitesse de rotation relative entre les deux corps rigides qu'elle relie ou fixer des bornes à l'orientation relative entre ceux-ci. Pour animer le personnage, il est nécessaire d'appliquer à chaque pas de temps de la simulation des moments et forces sur les membres et articulations du personnage. Fournir en paramètres d'entrée à la simulation la totalité des moments à appliquer pour chaque pas de temps de simulation ne permettrait pas au personnage de s'adapter aux perturbations dues aux interactions avec l'environnement. De même, le nombre élevé de moments nécessaires (un pour chaque articulation à chaque pas de temps) rend cette approche indésirable. Il est alors nécessaire de concevoir un contrôleur qui va calculer, à la volée, les moments à appliquer en fonction de l'état du personnage et des paramètres fournis par l'utilisateur. Un contrôleur doit permettre le contrôle du personnage à partir de paramètres tels qu'un type de mouvement, une vitesse de déplacement et une direction ou par l'application de modèles tels qu'un pendule inversé. Un contrôleur est dit de bas-niveau quand il calcule les moments pour une tache spécifique (par exemple faire marcher un personnage en ligne droite à vitesse constante). Un contrôleur de haut-niveau est utilisé pour organiser de nombreux contrôleurs de bas-niveau pour obtenir une plus grande flexibilité dans le contrôle du personnage. La figure 2.1a illustre la boucle de fonctionnement d'une animation basée sur la physique.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

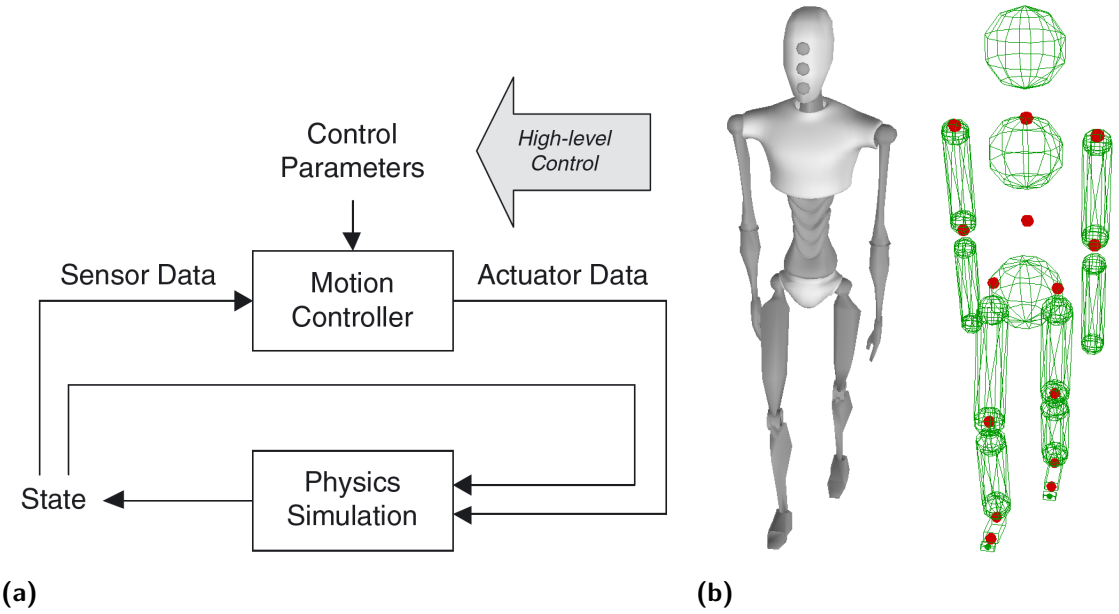



**Figure 2.1.:** (a) Boucle de fonctionnement pour le contrôle basée sur la physique d'un personnage virtuel [GP12]. (b) Structure de corps rigides et articulations composant un personnage (gauche : maillage du personnage, droite : corps rigides simulés (vert) et articulations (rouge))

Comme souligné ci-dessous, un personnage est représenté par un système de corps rigides reliés par des articulations. Ce système peut être plus ou moins enrichi suivant le type d'interactions utilisé. On retrouve deux principaux types de modèle de personnage. Le premier consiste à avoir un personnage comme présenté précédemment (c.à.d. des corps rigides reliés par des articulations). Lorsque l'on utilise ce type de personnage, trois types d'interactions peuvent être utilisées pour animer le personnage :

— **Moments articulaires :** Ce type d'interaction est le moyen le plus direct pour contrôler la pose du personnage. L'application de moments articulaires se rapproche de l'animation cinématique de personnage en contrôlant les angles au niveau des articulations via une accélération angulaire.

— **Moments ou forces externes :** Dans ce cas, le moment, ou plus souvent la force, est directement appliqué sur l'un des corps rigides composant le personnage. Ce type d'interaction permet des actions qui seraient difficiles de reproduire en appliquant un moment interne au personnage. Par exemple appliquer une force sur le centre de masse pour maintenir le personnage en équilibre. Cependant, elles peuvent mener à une perte de réalisme car elles permettent la simulation de mouvements ne pouvant pas être réalisés par un personnage réel. Ce type d'interaction est communément appelée *hand of god* [PL95].

— **Forces virtuelles :** Cette catégorie ne correspond pas à une interaction directe avec le personnage mais correspond à une solution permettant de répliquer l'impact d'une force externe sur le personnage sans la perte de réalisme associé aux forces externes. La méthode utilisée est de calculer les moments sur les diverses articulations des personnages qui auraient un résultat similaire à l'application de la force désirée [GP12]. Cette technique peut être utilisée comme méthode de contrôle. Elle est présentée en détail dans la section 2.4.2.


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

Le modèle de personnage peut être enrichi par l'ajout d'éléments représentant des muscles. Ce second type de modèle permet de calculer les moments à appliquer au niveau des articulations par la contraction des différents muscles du personnage. Cette approche vise à augmenter le réalisme du résultat en se rapprochant davantage des phénomènes liés à la physiologie des muscles. Il est cependant nécessaire de noter que l'augmentation de la complexité du personnage a pour conséquence d'augmenter la complexité de la méthode de contrôle du mouvement du personnage. Pour plus de détails sur les différents modèles, on peut se référer à la revue des méthodes publiée par Cruz et al. [Rui+16]. Les travaux réalisés dans le cadre de cette thèse utilisent un modèle de personnage contrôlé par l'application de moments articulaires et de forces virtuelles.

## 2.3 Types de contrôleur basé sur la physique

Comme nous l'avons présenté dans notre introduction, pour déterminer les forces et moments à appliquer à chaque instant, il est nécessaire d'utiliser un contrôleur. Le contrôleur répertorie des règles qui calculent les forces et moments à appliquer sur les membres et articulations du personnage en fonction de l'état du personnage et du reste de la simulation. Geijtenbeek et Pronost [GP12] proposent de classer les contrôleurs existant en trois catégories : contrôle dans l'espace des articulations, contrôle optimal, et algorithmes d'évolution et réseaux de neurones.

### 2.3.1 Contrôle dans l'espace des articulations

Cette approche provient de la théorie classique du contrôle. La figure 2.2 présente le processus global des contrôleurs reposant sur cette approche. Le contrôleur est constitué de deux groupes de composants : la génération de la pose cible et les systèmes de feedback. Le premier groupe va générer une pose cible pour le prochain pas de simulation à partir de la pose actuelle du personnage, des paramètres haut-niveau (ex : vitesse de déplacement, direction...) ainsi que de modèles physiques ou d'un mouvement de référence composé de poses clefs. La pose cible contient l'orientation et la vitesse angulaire désirée pour toutes les articulations du personnage. Cette pose est transmise au second groupe de composants qui calcule le moment à appliquer sur chaque articulation à partir de la distance entre la pose actuelle du personnage et celle décrite dans la pose cible. Le second groupe de composants rassemble des systèmes qui ajoutent des moments supplémentaires sur certaines articulations spécifiques. Ces composants servent généralement à améliorer un aspect spécifique du contrôle du personnage (ex : maintien de l'équilibre, contrôle précis de la vitesse, interaction spécifique avec l'environnement...).

La méthode la plus courante permettant de déterminer le moment à appliquer à partir de la pose cible est la Régulation Proportionnelle - Dérivée (Régulation PD).

#### Régulation PD

La régulation PD est un processus d'asservissement très répandu dans les systèmes de contrôle industriels. Elle utilise une loi linéaire selon la distance entre une consigne et une mesure afin de calculer un signal de commande. En animation physique, le signal de commande est généralement le moment articulaire $\tau$ à appliquer. La consigne et la mesure font intervenir



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

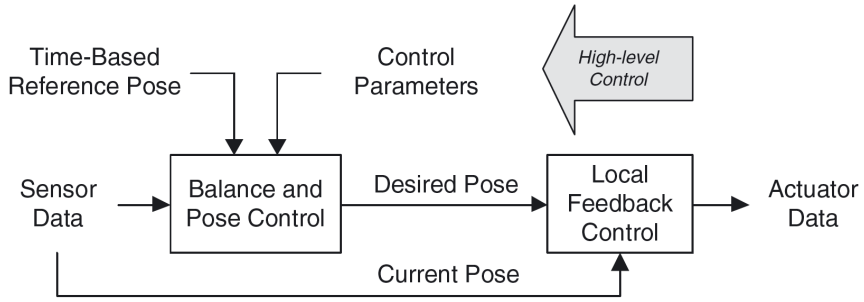



**Figure 2.2.:** schéma du fonctionnement d'un système de contrôle dans l'espace des articulations [GP12]

l'orientation courante $\theta$ de l'articulation et l'orientation désirée $\theta_d$ ainsi que la vitesse angulaire courante $\dot{\theta}$ et celle désirée $\dot{\theta_d}$ (équation 2.1).

$$\tau = k_p * (\theta_d - \theta) + k_v * (\dot{\theta_d} - \dot{\theta}) \tag{2.1}$$

On notera la présence de constantes $k_p$ et $k_v$, appelés gains du régulateur, qui permettent de contrôler la vitesse de convergence et la stabilité du système. Des gains élevés permettent de rendre la convergence plus rapide et donc d'obtenir un résultat plus réactif. Cependant, le mouvement aura un aspect robotique et le régulateur produira des oscillations autour de la cible qui peuvent mener à une simulation instable si ces oscillations divergent. À l'inverse, des gains faibles assureront une meilleure fluidité du mouvement et permettront d'éviter d'avoir un système divergent. Cependant, la convergence vers la cible sera plus lente. Habituellement, les gains sont trouvés soit par un processus d'essai-erreur, soit par une étape d'optimisation hors ligne [Pan96]. Il existe plusieurs variantes de ce régulateur.

**Régulateur PID** Le régulateur Proportionnel – Intégral – Dérivé (régulateur PID) diminue la présence d'oscillations autour de la cible en ajoutant à l'équation 2.1 un terme permettant d'intégrer la variation de l'erreur au cours du temps dans le calcul du moment (équation 2.2).

$$\tau = k_p * (\theta_d - \theta) + k_i * \int_0^t (\theta_d(x) - \theta(x))dx + k_v * (\dot{\theta_d} - \dot{\theta}) \tag{2.2}$$

Ce type de régulateur est le plus répandu dans le domaine de l'industrie. Cependant, considérer la variation de l'erreur n'est efficace que dans les cas d'une cible fixe ou d'une cible dont le mouvement ne varie pas. De ce fait, le régulateur PID n'est pas utilisé dans le domaine de l'animation où la pose cible du personnage varie au cours de la simulation.

**Régulateur PD stable** Ce régulateur, proposé par Tan et al. [JLT11], améliore la stabilité lors de l'utilisation de gains élevés. Au lieu d'utiliser l'état courant du personnage, ce régulateur prédit l'état au prochain pas de temps de la simulation en utilisant la vitesse et l'accélération angulaire de chaque articulation.

$$\tau = k_p * (\theta_d^{t+1} - (\theta^t + \Delta t \dot{\theta}^t)) + k_v * (\dot{\theta_d}^{t+1} - (\dot{\theta}^t + \Delta t \ddot{\theta}^t)) \tag{2.3}$$


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

### Génération des poses cibles

Deux approches différentes sont couramment utilisées pour générer la pose cible utilisée par le régulateur PD : la génération procédurale et le suivi d'un mouvement de référence. Il est important de noter que les deux approches ne sont pas exclusives et de nombreux travaux utilisent une combinaison des deux.

**Génération procédurale** La génération procédurale est la première méthode à avoir été explorée. Habituellement, les composants basés sur ce type d'approche utilisent des modèles physiques simples reproduisant des actions observées sur des personnages réels [Hod+95]. L'un des modèles les plus communs dans la locomotion bipède est le modèle du pendule inversé (IPM) [Kaj+01 ; CBV10]. Le but du pendule inversé, dans le cas de personnages bipèdes, est de trouver l'emplacement où le personnage doit poser son pied à la fin de chaque pas. La position du pied doit permettre la conservation de l'équilibre au cours du mouvement. Les détails du principe de l'IPM et de ses extensions sont présentés en section 2.4.1.

Jie et al. [Tan+11] ont proposé un contrôleur capable de simuler la nage de créatures aquatiques simples à partir d'une fonction sinusoïdale en profitant de la périodicité contenue du mouvement. Leur système est cependant limité à des créatures avec un nombre très faible d'articulations car il repose sur une optimisation hors ligne pour trouver les paramètres de la fonction périodique utilisée. La limitation principale des systèmes reposant uniquement sur de la génération procédurale est qu'il est impossible de contrôler le style du mouvement généré. Cela entraîne la génération de mouvements parfois peu naturels [NF02 ; HRL15].

**Suivi de mouvement de référence** Une solution pour rendre les animations produites plus réalistes est de créer un contrôleur qui a pour but de suivre un mouvement de référence donné en paramètre du système. Pour spécifier l'exemple à suivre, il est possible de s'inspirer des techniques classiques d'animation. Le mouvement de référence est décrit à travers la spécification de poses clefs décrivant l'évolution du personnage au cours de l'action à réaliser. Cette approche a plusieurs avantages. Elle permet l'utilisation de mouvements pour lesquels trouver un modèle physique est trop compliqué voire impossible. Elle offre également un contrôle du résultat beaucoup plus intuitif et rend possible l'ajout de détails dans l'animation produite. Cependant les poses clefs choisies pour une action donnée sont intimement liées aux valeurs de gains utilisées pour les régulateurs PD. En effet, l'utilisation de gains faibles, permettant d'éviter un aspect robotique dans l'animation, fait que le personnage n'atteindra jamais exactement les poses clefs spécifiées. L'une des alternatives est de légèrement exagérer le mouvement au sein de l'animation de référence. Utilisée pour le contrôle de personnages en 2D [PKF94] et étendue à un personnage 3D [LPF96], cette approche est de nos jours à la base de la quasi-totalité des contrôleurs existants.

Cependant, le simple suivi d'un mouvement de référence ne permet pas de maintenir le personnage en équilibre lorsque l'on applique des perturbations externes, particulièrement pour les contrôleurs 3D. C'est pourquoi la majeure partie des systèmes basés sur le contrôle dans l'espace des articulations utilisent le suivi d'un mouvement de référence comme base pour la pose cible et altèrent celle-ci avec l'application de modèles permettant, entre autres, au personnage de conserver son équilibre [LPF96 ; CBV10].

Une fois le moment calculé par les régulateurs PD sur chaque articulation, le second groupe de composants ajoutent des moments supplémentaires pour améliorer le contrôle du personnage. Parmi les fonctionnalités présentées dans les travaux existants, on trouve notamment :



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

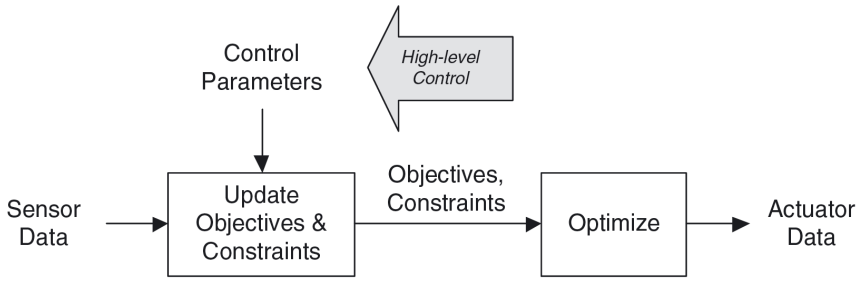



**Figure 2.3.:** schéma du fonctionnement de l'approche basée sur le contrôle optimal [GP12]

l'amélioration du suivi du mouvement de référence par calcul direct d'une partie du moment nécessaire [YLV07 ; CBV10], le contrôle de la vitesse du personnage [CBV10 ; CPB15] et l'amélioration de l'équilibre du personnage [ZP05 ; Sil+17].

Un exemple courant de système basé sur le contrôle dans l'espace des articulations est le SIMple BIped CONtroler (SIMBICON) proposé par Yin et al. [YLV07]. Le SIMBICON est un framework générique, au sein duquel il est facile d'ajouter des composants pour gérer différentes actions comme la capacité à contrôler la direction et vitesse du personnage, ou l'intégration de composants permettant de gérer un milieu spécifique [YLV07 ; CBP09 ; CBV10 ; CPB15]. Également, la simplicité du système résulte en une très faible quantité de ressources de calcul nécessaire à chaque pas de temps de simulation. Ceci rend le SIMBICON intéressant dans le cadre de la création de systèmes nécessitant une phase d'optimisation hors ligne ou des performances interactives.

Bien que simples à concevoir et à implémenter, ces contrôleurs ont le défaut d'être spécialisés pour des mouvements simples. Il est, en effet, très compliqué de concevoir un contrôleur permettant de gérer des mouvements variés avec ce type d'approche.

### 2.3.2 Contrôle optimal

Cette approche provient de la théorie du contrôle optimal et des méthodes d'optimisation. Les contrôleurs de ce type tentent de trouver les moments qui correspondent à la solution optimale d'un ensemble de contraintes définies par un modèle ou par une mise en équation des paramètres fournis par l'utilisateur.

Les contrôleurs basés sur le contrôle optimal se distinguent par l'utilisation d'une optimisation en ligne. Le but est de trouver les valeurs des moments à appliquer pour minimiser une fonction objectif tout en respectant un ensemble de contraintes. La fonction objectif permet de définir les propriétés de l'animation à obtenir (ex : vitesse, niveau d'énergie...). Cette fonction est également utilisée pour contrôler le mouvement global en évaluant la capacité du contrôleur à réaliser une action. L'ensemble de contraintes permet de limiter l'espace de recherche de l'optimisation. Les contraintes principales sont celles présentes à l'intérieur du moteur physique et qui assurent le réalisme physique de l'animation. Ces contraintes sont également utilisées pour représenter les limites physiques du personnage (ex : vitesses angulaires et moments maximaux sur les articulations). La figure 2.3 présente le processus global des contrôleurs basés sur le contrôle optimal. Geijtenbeek et Pronost [GP12] distinguent trois approches pour la conception de contrôleurs optimaux.

L'approche la plus intuitive est de concevoir une fonction évaluant chaque pas de temps indépendamment des autres et d'utiliser cette fonction pour évaluer l'animation complète.


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

On rencontre cette approche dans les premiers travaux utilisant le contrôle optimal [BN88 ; Ste92 ; SC92] mais également dans des travaux plus récents [WZ10]. La limitation de cette approche est que l'optimisation ne contient pas de capacité de prévision du fait qu'elle ne considère que le pas de temps courant ce qui limite la robustesse globale du contrôleur.

Une seconde approche est de réaliser une première optimisation hors ligne pour disposer d'un modèle de contrôle qui va permettre de guider l'optimisation en ligne. L'utilisation de l'optimisation hors ligne permet d'intégrer des éléments de prédiction dans l'optimisation en ligne sans avoir à traiter tout le mouvement pour chaque pas de temps de simulation. Cependant, cette capacité de prédiction est introduite en sacrifiant la généricité du contrôleur qui est l'un des avantages principaux du contrôle optimal. Muico et al. [Mui+09 ; MPP11] utilisent cette méthode pour améliorer la stabilité des contacts avec le sol en intégrant les forces de réaction du sol sur le pied dans les contraintes d'optimisation.

L'approche ayant permis d'obtenir les résultats les plus prometteurs est le contrôle prédictif par modèle (MPC). Comme le régulateur PID, le MPC est une méthode fréquemment utilisée en industrie. Cependant, contrairement au régulateur PID spécialisé dans la gestion de systèmes simples, les systèmes basés sur des MPC sont utilisés sur des modèles plus complexes. En effet, on trouve leur utilisation dans des systèmes ayant un plus grand nombre de variables et pour lesquels la dimension temporelle est cruciale (notamment dans l'industrie chimique). Un MPC fonctionne en optimisant non seulement le pas de temps actuel mais aussi une courte fenêtre de temps futur. Contrairement aux approches présentées précédemment, un MPC ne conserve pas les résultats de la prédiction sur la fenêtre de temps supplémentaire et répète les calculs à chaque pas de temps. Cette fenêtre de temps permet d'évaluer les interactions entre des variables indépendantes les unes des autres. Du fait que les calculs sur la fenêtre de temps supplémentaire ne sont pas conservés, un modèle réduit du système à simuler, c.à.d. du personnage dans notre domaine, est utilisé pour les pas de temps qu'elle contient. Les modèles réduits simplifient les calculs physiques nécessaires ce qui permet de diminuer les ressources de calcul requises. Parmi les travaux récents utilisant cette approche, le contrôleur proposé par Hämäläinen et al. [Häm+14 ; HRL15] offre un aperçu de l'intérêt du MPC. Grâce à la capacité de prédiction de la méthode, ce contrôleur est capable de rétablir l'équilibre du personnage même lorsque ce dernier subit des perturbations externes très importantes. De plus, le contrôleur proposé utilise peu de paramètres car il ne dépend pas d'un mouvement de référence. Han et al. [Han+16 ; HNS18] proposent d'utiliser ce contrôleur et de l'associer à un mouvement de référence pour rendre l'animation générée plus naturelle. Leur contrôleur est capable de générer des mouvements complexes tels que la danse et les arts martiaux même sous l'effet de perturbations externes élevées.

L'avantage principal des contrôleurs optimaux est qu'ils intègrent une capacité de prédiction dans le contrôle. Cela assure une animation stable. Malgré la qualité des animations produites et la possibilité de reproduire des mouvements très complexes et dynamiques, ces contrôleurs présentent plusieurs inconvénients. L'utilisation d'une optimisation en ligne requiert une large capacité de calcul si l'on veut obtenir une simulation en temps-réel. Ces contrôleurs demandent également une connaissance bien plus avancée de la simulation de corps rigides et des processus d'optimisation que ceux qui utilisent le contrôle dans l'espace des articulations ou ceux basés sur des algorithmes d'évolution. La conception d'un tel contrôleur est par conséquent complexe.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

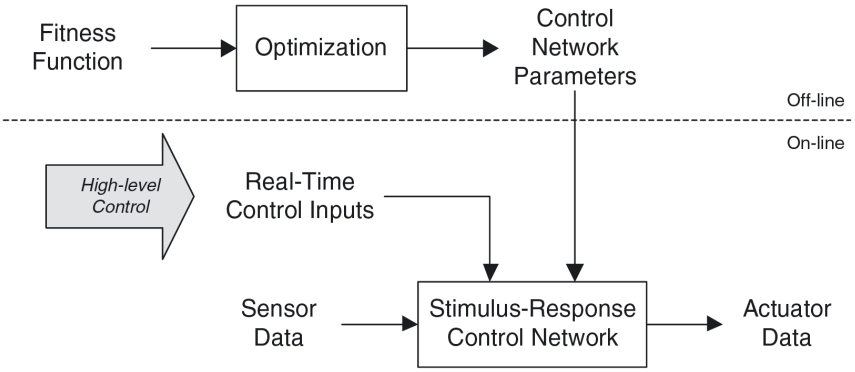



**Figure 2.4.:** schéma du fonctionnement de l'approche basée sur les algorithmes génétiques et les réseaux de neurones [GP12]

### 2.3.3 Algorithmes génétiques et réseaux de neurones

Cette approche s'inspire des méthodes d'intelligence artificielle. Elle est caractérisée par une phase d'apprentissage. Cette phase consiste en une d'optimisation hors ligne qui permet de générer un modèle de contrôle qui pourra par la suite être utilisé pour le contrôle en ligne du personnage. Ce type de contrôleur est à différencier des contrôleurs dont l'étape d'optimisation hors ligne sert à générer les paramètres de leur système. La figure 2.4 illustre le fonctionnement d'un contrôleur utilisant un réseau de neurones.

Geijtenbeek et Pronost [GP12] notent que, bien que ce type d'approche ait été exploré depuis assez longtemps [PF93], leur utilisation restait limitée jusqu'à ces dernières années. La caractéristique principale de cette approche est la capacité à créer des mouvements inattendus [Sim94]. Récemment, cette approche a été utilisée pour créer des contrôleurs à plusieurs niveaux capables de supporter la locomotion de personnages bipèdes sur des terrains complexes tel qu'un terrain accidenté [Pen+17 ; Ber+18b]. Peng et al. [Pen+18] proposent un contrôleur capable d'animer un personnage bipède humain. Ce contrôleur se sert de mouvements de référence pour guider l'optimisation vers une animation réaliste.

Bien que les approches utilisant des réseaux de neurones soient capables d'extraire des informations complexes des mouvements de référence fournis, ce n'est que récemment que des résultats concluants ont été obtenus pour des personnages complexes dans le domaine de l'animation basée physique ([Pen+17 ; Ber+18b ; Pen+18]). Il faut souligner que la phase d'apprentissage pour de tels contrôleurs nécessite de grandes ressources de calcul sur une durée de plusieurs jours. Outre cette complexité (ressources de calcul et données), il est difficile d'avoir un contrôle précis sur le résultat. Ceci rend cette approche peu appropriée pour le cas d'un scénario de simulation précis.

### 2.3.4 Bilan

On peut résumer les avantages et inconvénients de chacune des trois approches pour la conception d'un contrôleur dans le tableau 2.2.

L'avantage principal des approches basées sur les algorithmes génétiques, ou les réseaux de neurones, est leur capacité à créer des contrôleurs de haut niveau rendant possible les transitions fluides entre de nombreuses actions. Cette flexibilité du contrôleur est obtenue par une étape de pré-calcul importante ce qui a pour effet de limiter le contrôle sur le résultat


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

| | Contrôle dans l'espace des articulations | Contrôle optimal | Algorithmes génétiques |
|---|---|---|---|
| Niveau de maturité | ++ | ++ | - |
| Qualité du résultat | - | ++ | + |
| Généricité de l'approche | - - | ++ | + |
| Contrôle sur le résultat | ++ | - | - - |
| Temps de pré-calcul | + | + | - - |
| Temps de calcul durant l'animation | ++ | - | ++ |

**Tableau 2.2.:** tableau récapitulatif des avantages et inconvénients des différentes approches pour le contrôle de personnages. Les "+" indiquent que la méthode présente des avantages pour le critère en question. Par exemple, un "+" au niveau du temps de pré-calcul indique que le temps de calcul nécessaire avant la réalisation de l'animation est faible.

final. Rappelons que nous nous intéressons au contrôle basé physique d'un personnage en interaction continue avec un environnement complexe. L'obtention d'un contrôleur haut niveau capable de supporter de nombreuses actions n'est pas l'un de nos objectifs principaux. Par conséquent, nous avons jugé qu'il n'était pas pertinent d'introduire dans notre système les temps de pré-calcul nécessaires aux approches basées sur les algorithmes génétiques et nous avons décidé de ne pas nous concentrer davantage sur cette approche.

L'application du contrôle optimal apporte au contrôleur une robustesse extrême aux perturbations externes. Cette propriété est très intéressante car elle permettrait de quasiment assurer la stabilité du mouvement lors de son interaction avec des milieux aussi imprévisibles que des fluides. Cependant, bien que certaines approches MPC offrent des résultats interactifs, ces résultats sont obtenus en utilisant la totalité des ressources de calcul disponibles [Han+16 ; HNS18]. De plus, ces résultats nécessitent l'utilisation de fréquences de simulation trop faibles par rapport aux fréquences permettant la simulation d'un fluide stable. Par conséquent, il serait difficile de gérer la simulation de l'environnement ainsi que ses interactions avec le personnage tout en atteignant l'objectif d'un contrôleur temps réel.

On retient de cette étude que l'approche du contrôle dans l'espace des articulations permet d'avoir une plus grande facilité d'utilisation ainsi que des temps de calculs plus faibles tout en apportant le niveau de capacité de contrôle du personnage dont nous avons besoin. Dans notre cas, la limitation de ce type de contrôleurs à des mouvements simples n'est pas un problème majeur car nous ne cherchons pas à simuler de mouvements complexes mais uniquement de la marche. Par conséquent, l'approche du contrôle dans l'espace des articulations est la plus adaptée pour les travaux de cette thèse.

De nombreux travaux proposent des modèles composés de systèmes de feedback qui permettent d'améliorer ce type de contrôleur. Nous nous sommes intéressés au modèle le plus répandu : le *SIMple BIped CONtroler* (SIMBICON).

## 2.4 SIMBICON et extensions

Le *SIMple BIped CONtroler* (SIMBICON) est un modèle générique de contrôleur basé sur le contrôle dans l'espace des articulations. Ce modèle a été proposé par Yin et al. [YLV07].



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

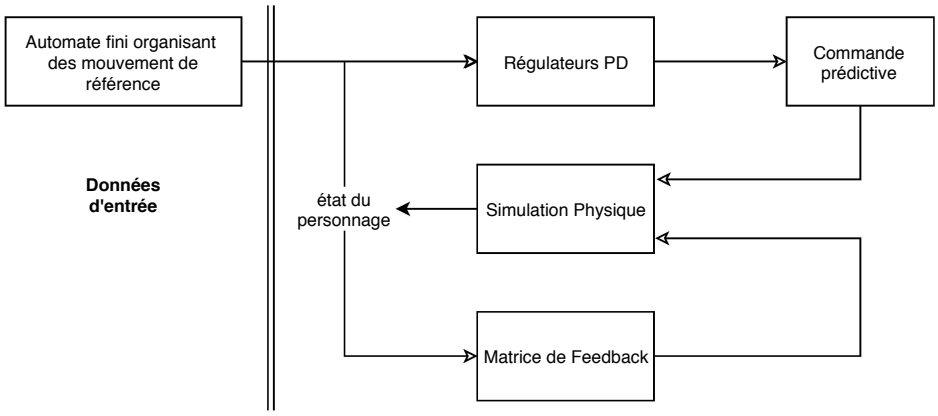



**Figure 2.5.:** schéma fonctionnel des composants du SIMBICON [YLV07]

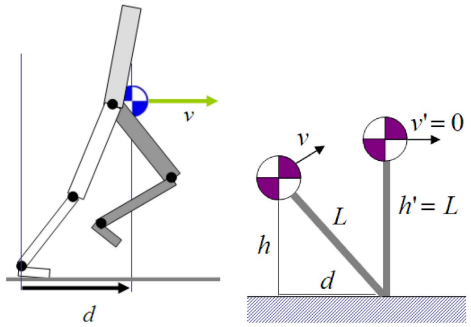



**Figure 2.6.:** (a) Représentation des poses cibles utilisées pour la matrice de feedback. (b) Schéma d'un IPM pour un personnage bipède ([CBV10])

La figure 2.5 présente une vue d'ensemble de la structure du SIMBICON. Pour générer la pose cible à transmettre aux régulateurs PD, le SIMBICON utilise une machine à états finis. Chacun des états de la machine contient un mouvement de référence. Les transitions entre les états sont effectuées à chaque fois qu'un pied touche le sol. L'utilisation de cette machine à états finis à l'avantage de donner une représentation très simple de mouvements cycliques tels que la course ou la marche. Ces mouvements sont également symétriques et donc il suffit de spécifier le mouvement de référence d'un seul pas pour pouvoir générer une animation complète. En plus des moments calculés à partir de la pose cible par les régulateurs PD, le SIMBICON utilise deux composants qui ajoutent des moments supplémentaires :

— **Matrice de feedback :** Ce système ajoute un moment supplémentaire sur la hanche et la cheville faisant partie de la jambe en phase d'appui pour contrôler l'équilibre du personnage. Le moment est calculé à partir d'une loi linéaire pour laquelle les paramètres sont déterminés par l'utilisation d'une optimisation hors ligne. Le moment supplémentaire dépend de la vitesse $v$ du centre de masse ainsi que de la distance $d$ dans le plan horizontal entre la position du centre de masse et la cheville de la jambe d'appui (figure 2.6a) : $\tau = c_d d + c_v v$, avec $c_d$ et $c_v$ des constantes obtenues dans l'étape d'optimisation hors ligne.

— **Commande prédictive :** Le but de ce système est d'appliquer directement une partie des moments selon une phase $\phi$ sans avoir à regarder la distance entre le personnage et la cible à atteindre. La phase est calculée selon $\phi = t/T_{max}$ avec $t$ le temps écoulé depuis le début du pas et $T_{max}$ le temps prévu pour l'état actuel de la machine à états finis. En


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

début de simulation, Yin et al. [YLV07] appliquent des moments principalement calculés par des régulateurs PD. Au fur et à mesure de la simulation, leur contrôleur apprend les moments à appliquer en fonction de la phase. Une fois que la période d'apprentissage est terminée, les moments appliqués sont calculés principalement à partir des valeurs apprises et seule une faible partie est calculée par les régulateurs PD. L'avantage d'avoir seulement une faible partie des gains produite par la différence entre la pose cible et la pose actuelle est qu'il devient possible d'utiliser des gains faibles pour les régulateurs PD, ce qui permet d'éviter l'apparition de mouvements robotiques comme dans le cas des régulateurs PD (section 2.3.1).

La simplicité du SIMBICON a rendu ce système populaire et de nombreux auteurs ont proposé des améliorations. Parmi ces améliorations, l'utilisation d'un modèle de pendule inversé (IPM) est l'une des solutions couramment utilisées dans le mouvement bipède.

### 2.4.1 Modèle de pendule inversé

Un modèle de pendule inversé ou IPM (*Inverted Pendulum Model*) permet de maintenir l'équilibre du personnage lors d'un déplacement bipède. Ce modèle fonctionne en calculant la position où doit être posé le pied du personnage pour que le personnage conserve son équilibre. L'IPM couramment utilisé pour la simulation de personnage bipède considère que la longueur de la jambe d'appui, définie comme la distance entre la hanche et la cheville, est constante (c.à.d. l'angle du genou est constant au cours d'un pas). Dans ce modèle, la somme de l'énergie potentielle de hauteur $E_p$ et de l'énergie cinétique $E_c$ doit être constante au cours du temps. De ce fait on peut écrire l'équation suivante :

$$E_p + E_c = E_p\text{'} + E_c\text{'} \tag{2.4}$$

$$\frac{1}{2}mv^2 + mgh = \frac{1}{2}mv\text{'}^2 + mgh\text{'} \tag{2.5}$$

Avec $v$ la vitesse du centre des masses (COM), $h$ la hauteur du COM, $m$ la masse, $g$ la valeur de la gravité et le symbole ' indique l'état de la simulation à un second instant. Si on note $d = P_{COM}\text{'} - P_{COM}$ la distance horizontale entre les positions des COM des deux instants considérés, on a $L^2 = h^2 + d^2$, $L$ étant la longueur de la jambe du personnage (figure 2.6b). Considérons le second instant tel que le COM est à la verticale du pied d'appui. Dans ce cas, $d$ correspond également à la distance (relative à la position courante du COM) à laquelle le personnage doit poser son pied et on a $h\text{'} = L$. En résolvant l'équation 2.5, on peut obtenir la formule pour la valeur de $d$ :

$$d = \sqrt{\Delta_{v^2}(\frac{\Delta_{v^2}}{4g^2} + \frac{h}{g})} \tag{2.6}$$

Avec $\Delta_{v^2} = v^2 - v\text{'}^2$. Dans le cas où l'on cherche la position d'appui pour une vitesse nulle à la verticale, on a $v\text{'} = 0$, donc $\Delta_{v^2} = v^2$.

$$d = v\sqrt{\frac{v^2}{4g^2} + \frac{h}{g}} \tag{2.7}$$



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

Un problème associé à l'IPM est qu'il considère un contact avec le sol en permanence. De ce fait, il ne peut pas être utilisé pour des mouvements de saut ou de courses. L'utilisation d'un système composé d'une masse et d'un ressort a été proposée par Blickhan et al. [Bli89] pour animer la course et les sauts.

Un défaut de l'utilisation d'un IPM est la perte en liberté de création du mouvement. En effet, l'utilisateur du contrôleur perd tout contrôle sur la trajectoire de la jambe en phase de vol. Dans nos précédents travaux [CPB15], nous limitons l'utilisation de l'IPM à la partie descendante de chaque pas, partie durant laquelle le centre de masse se rapproche du sol. Cette limitation permet de spécifier un mouvement de référence pour le début de chaque pas introduisant la possibilité de créer de nouveaux styles de marche.

Lorsque l'on simule un déplacement bipède, il est généralement préférable que le mouvement ne s'arrête pas après chaque pas, et donc il serait préférable de ne pas considérer une vitesse nulle lorsque le COM se trouve à la verticale du pied d'appui. Cependant si l'on regarde l'équation 2.6, on remarque qu'elle n'est valide que sur les intervalles suivants $\Delta_{v^2} \in [-\infty; -4hg] \cup [0; +\infty]$. Pour avoir un contrôle sur la vitesse du personnage tout en évitant cette condition, Coros et al. [CBV10] utilisent une version modifiée de l'équation 2.7 qui modifie la longueur du pas suivant la vitesse désirée $V_d$ du personnage.

$$d' = d - \alpha V_d \tag{2.8}$$

Avec $\alpha$ une constante. Le principe est qu'en faisant des pas plus courts le personnage sera déséquilibré vers l'avant et donc sa vitesse sera supérieure à zéro lors de son passage à la verticale du pied. À l'inverse, des pas plus longs permettent de ralentir le personnage. Cependant ce système se trouve confronté à deux problèmes. Le premier est que rien n'assure que le personnage atteindra bien la position demandée pour le pied à la fin du pas. De ce fait, cette modification ne permet qu'un contrôle grossier de la vitesse du personnage. Le second problème est que la valeur $\alpha$ est une valeur fixée. La conséquence est que suivant les interactions subies par le personnage, la vitesse obtenue par ce contrôle de vitesse ne sera pas celle désirée. Pour améliorer ce système et apporter une réponse à ce problème, nous avions proposé d'ajouter à la formule 2.8 une correction supplémentaire $\Delta_s$. Cette correction est recalculée à la fin de chaque pas en fonction de l'erreur entre la vitesse désirée et la vitesse observée [CPB15] :

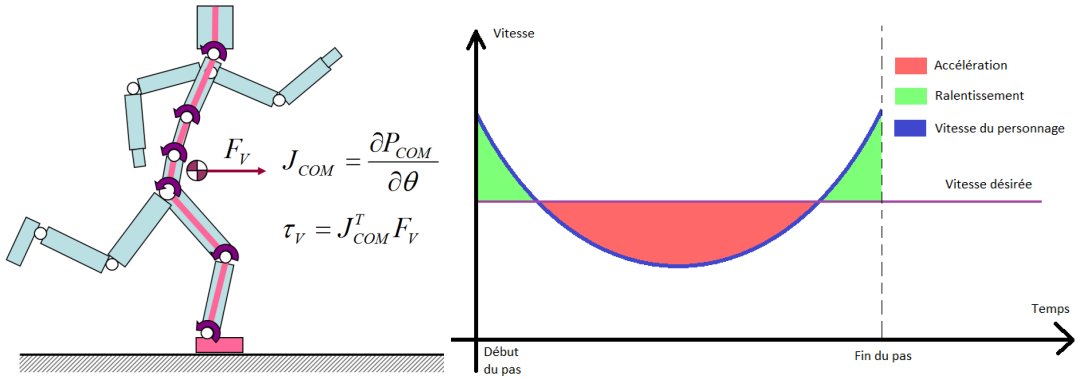
$$d' = d - \alpha V_d + \Delta_s \tag{2.9}$$

Bien que cette modification permette de ne plus être complètement dépendant de la valeur de $\alpha$ pour atteindre la vitesse désirée, la vitesse de convergence $v$ est très lente si l'on veut éviter les oscillations autour de la cible. Ce système reste également très sensible aux variations de l'environnement. Pour obtenir un contrôle précis de la vitesse désirée, d'autres systèmes de feedback ont été utilisés en coordination avec l'IPM que nous allons maintenant présenter.

### 2.4.2 Adaptation de la vitesse

Parmi les systèmes qui permettent d'obtenir un contrôle plus fin de la vitesse désirée, on trouve le système de *velocity tuning* proposé par Coros et al. [CBV10] pour leur extension du SIMBICON appelée *GENeralized BIped CONtrol* (GENBICON). Ce système fonctionne par


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés




(a) (b)

**Figure 2.7.:** (a) Chaîne de corps rigides affectée par le *velocity tuning* [CBV10]. (b) Illustration de l'accélération et du ralentissement à l'intérieur d'un pas causé par le *velocity tuning* proposé par Coros et al. [CBV10]

l'application d'une force virtuelle sur le personnage pour l'accélérer ou le ralentir. Pour rappel, une force virtuelle n'est pas directement appliquée sur le personnage. À la place, on applique des moments ayant un impact similaire à celui qu'aurait eu cette force. Une méthode commune est d'utiliser la transposée de la jacobienne du personnage. La jacobienne $J$ du personnage contient l'information indiquant l'impact de la variation de l'angle $\theta$ de chaque articulation sur le point d'application $P$ de la force à représenter $J = \frac{\partial P}{\partial \theta}$. Le moment à appliquer sur les articulations est obtenu en multipliant la transposée de la matrice obtenue par la force désirée $\tau = J^T F$. Cette technique est utilisée pour le contrôle d'un robot bipède dans les travaux de Pratt et al. [Pra+01] ou le maintien de l'équilibre d'un personnage simulé [GPS12]. Le GENBICON utilise un calcul différent pour l'axe sagittal et l'axe coronal, respectivement un régulateur D et un régulateur PD :

$$\begin{cases} F_{sag} = K_{v\_sag} * (v_{d\_sag} - v_{sag}) \\ F_{cor} = K_{v\_cor} * (v_{d\_cor} - v_{cor}) + K_p * (P_{COM\_cor}) \end{cases} \tag{2.10}$$

Avec $v_d$ la vitesse désirée. La différence entre les deux axes se justifie par le choix des auteurs de se déplacer sur une ligne droite. La force virtuelle est ensuite appliquée sur chaque articulation de la chaîne de corps rigides présentée dans la figure 2.7a. La cheville de la jambe d'appui étant à la racine de la chaîne de corps rigides affectée, ce système ajoute un moment élevé sur celle-ci, particulièrement pour des changements de vitesse importants. Ce moment élevé provoque des instabilités pouvant faire pivoter le pied en contact avec le sol. Pour limiter ce phénomène les auteurs fixent la valeur du moment ajouté à zéro si la vitesse angulaire au niveau de la cheville est trop élevée. Bien que ce système permette de faire varier de manière fine la vitesse du personnage, l'utilisation de valeurs fixes pour les gains des régulateurs calculant la force virtuelle entraîne un défaut similaire à celui présent sur l'IPM. La vitesse atteinte n'est pas la vitesse demandée si le personnage ne se trouve pas exactement dans les conditions pour lesquelles les gains ont été prévus. Également, ce système considère une vitesse cible constante durant toute la durée d'un pas du personnage. Cependant, au cours d'un pas, la vitesse observée du personnage n'est pas constante. Au cours d'un pas de marche la vitesse d'un bipède dessine une parabole car il ralentit sur la phase ascendante du pas. De ce fait en considérant une vitesse cible constante, le système proposé par Coros et



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

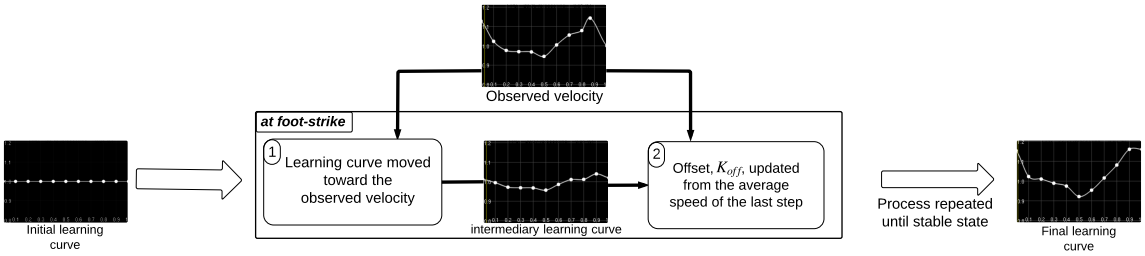



**Figure 2.8.:** Processus d'apprentissage de la courbe de vitesse cible pour le *velocity tuning* [CPB15]

al. peut faire ralentir le personnage au début du pas, puis l'accélérer au milieu et finir par le ralentir une fois de plus à la fin du pas (voir figure 2.7b).

Dans notre précédent contrôleur [CPB15] nous avons proposé une modification de ce système pour résoudre ces deux problèmes (figure 2.8). Au lieu de considérer une vitesse cible constante, nous avons utilisé une courbe de vitesses cibles qui est modifiée à la fin de chaque pas. Au cours du pas le système note l'évolution de la vitesse du personnage. À la fin du pas, la courbe de vitesse cible est modifiée pour qu'elle tende vers la vitesse observée. Au fur et à mesure du mouvement cette courbe va tendre vers la courbe de vitesse observée et donc fournira une vitesse cible adaptée à chaque instant du pas. Pour pouvoir contrôler la vitesse moyenne du personnage, nous avons associé à cette courbe de vitesse cible un offset global dont la valeur est modifiée à la fin de chaque pas selon si le personnage est trop lent ou trop rapide. Notons que pour permettre plus de flexibilité au système, celui-ci dispose de deux courbes d'apprentissage pour l'axe coronal en fonction du pied d'appui. En particulier, l'utilisation de deux courbes permet l'apprentissage dans des cas où le mouvement désiré n'est pas symétrique (ex : marche avec une vitesse coronale). Également, la jacobienne utilisée pour transformer la force virtuelle est modifiée pour séparer le haut du personnage de la partie basse. La raison est que pour modifier la position du torse il est plus naturel de pencher ce dernier plutôt que de tenter de le déplacer en modifiant l'angle de la cheville ou celui du genou. Cette modification permet de diminuer le moment ajouté sur la cheville et donc diminue la probabilité d'apparition d'instabilités.

La version du *velocity tuning* que nous avions proposée permet de contrôler exactement la vitesse du personnage après une dizaine de pas à la suite de la modification de la vitesse cible. Associé au contrôle de vitesse obtenu par l'utilisation de l'IPM, le contrôleur est capable de générer des mouvements extrêmement éloignés de la vitesse que l'on obtiendrait avec le mouvement de référence fourni au contrôleur. Notre système permet également d'atteindre les vitesses désirées même dans un milieu différent de celui correspondant au milieu du mouvement de référence. Cela permet de simuler les interactions entre le personnage et un milieu liquide. Cependant, il est possible d'observer, lors d'une marche symétrique, un apprentissage non symétrique sur les pas gauches et les pas droits. Ceci résulte en un état d'équilibre où la moyenne correspond bien à la vitesse désirée mais la vitesse au cours de pas consécutifs est différente. L'écart peut devenir tellement important que le contrôleur ne réussit plus à maintenir le personnage en équilibre.

### 2.4.3 Contrôle de haut-niveau

Une méthode pour augmenter la capacité des contrôleurs de type SIMBICON à produire de nombreuses actions au sein d'un seul contrôleur est d'utiliser une approche similaire à certains travaux utilisant des réseaux de neurones. On fournit au contrôleur de multiples


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

mouvements de référence adaptés à différentes actions ou environnements, puis on cherche à créer un contrôleur de haut-niveau capable de se servir de l'ensemble des mouvements donnés en paramètre pour obtenir un résultat stable.

Coros et al. [CBP09] utilisent cette méthode pour obtenir un contrôleur réalisant diverses tâches tout en étant résistant aux perturbations externes. L'approche utilisée est de se servir de mouvements de référence décrivant plusieurs actions différentes telles que la marche vers l'avant, la marche vers l'arrière ou des changements d'orientation. Leur contrôleur construit, lors d'une phase d'apprentissage hors ligne, une stratégie de contrôle (i.e. les conditions de transition entre les différents mouvements de référence) pour la réalisation d'une tache précise. Le contrôleur de haut-niveau obtenu présente une meilleure robustesse qu'un contrôleur se servant uniquement d'un seul des mouvements de référence. Cependant leur système requiert que chacun des mouvements de référence fourni soit capable de produire un mouvement stable même lorsqu'il est le seul mouvement donné en paramètre du contrôleur. Cette condition est difficile à obtenir car ces mouvements ne se basent pas sur l'utilisation d'un IPM. L'utilisation d'une phase d'optimisation hors ligne pour créer un contrôleur de haut-niveau est une méthode courante [LPY16].

les travaux que nous avons proposés dans [CPB15] ont recours à plusieurs mouvements de référence adaptés à l'environnement dans lequel le personnage se trouve. Cependant, les différents mouvements de référence n'ont pour but que le contrôle du style du mouvement et noter système n'en dépend pas pour réaliser différentes actions (contrôle de la vitesse ou de la direction, déplacement suivant un parcours...). Dans notre contrôleur, cette méthode est utilisée pour modifier le style du mouvement suivant le niveau du liquide en contact avec les jambes du personnage et la vitesse de ce dernier. Le système associe à chaque mouvement de référence un niveau de liquide et une vitesse sagittale. Au cours de la simulation, les poses cibles sont créées en utilisant une combinaison des poses clefs des mouvements de références associées aux conditions les plus proches de celles présentes dans la simulation. Pour obtenir un système qui fonctionne en temps réel, la combinaison est réalisée par une interpolation selon une loi d'ordre deux ce qui permet d'éviter une étape d'optimisation hors-ligne pour créer un contrôleur de haut-niveau.

### 2.4.4 Stabilité du contrôleur

Liu et al. [Liu+10] proposent le contrôleur SAMCOM (*SAMpling-based CONtrol strategy*) permettant d'animer un personnage à partir d'un mouvement de référence bruité obtenu par capture de mouvement. L'idée est qu'il est possible d'obtenir une simulation stable reproduisant le mouvement désiré en appliquant, sur l'orientation cible des articulations du personnage, de légères variations maintenues constantes pour une courte durée. L'ensemble des variations constantes à appliquer sur toutes les articulations sur une partie du mouvement est appelé un échantillon. Dans cette démarche le mouvement est divisé en fenêtres de temps de courte durée (0.1 seconde) et un échantillon différent est associé à chacune d'entre elles. Les valeurs des échantillons associés à chaque fenêtre sont obtenues en utilisant une optimisation réalisée sur les courtes fenêtres de temps traitées dans leur ordre chronologique. Pour chaque fenêtre de temps, cette optimisation génère un grand nombre d'échantillons (environ 1500), évalue le résultat obtenu et en sélectionne un certain nombre (entre 200 et 500) parmi les meilleurs résultats. Le résultat obtenu par l'optimisation pour la fenêtre de temps "n" sert sert pour initialiser l'état initial du personnage pour la fenêtre "n+1". Une fois que l'optimisation a été réalisée pour toutes les fenêtres de temps, le contrôleur conserve un seul échantillon par fenêtre de temps selon une fonction objectif évaluant le mouvement dans



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

sa totalité. Ce type d'optimisation a l'avantage d'être parallélisable car les évaluations des échantillons pour une fenêtre de temps sont indépendantes les unes des autres.

Cette technique permet de produire des animations à partir de mouvements de référence complexes tels que des acrobaties. Cependant, il possède un défaut notable : la quantité de ressources de calcul nécessaire est très importante. Pour leurs expériences, Liu et al. [Liu+10] utilisent un serveur de calcul qui dispose de 80 coeurs pour simuler 1 seconde de mouvement en 25 secondes de calcul. Également, cette technique ne réussit pas à animer des mouvements de référence longs car, si, au cours d'une fenêtre de temps, l'optimisation n'arrive pas à produire au moins un échantillon permettant d'obtenir une simulation stable, le contrôleur est forcé de redémarrer son optimisation depuis la première fenêtre de temps.

Une amélioration de ce contrôleur a été présentée par Liu et al. en 2015 [LYG15]. Trois modifications sont apportées au système. Premièrement, au lieu de générer exclusivement de manière aléatoire les échantillons, le nouveau contrôleur utilise une répartition inspirée de l'algorithme *Covariant Matrix Adaptation* (CMA-ES [Han06]). Cet algorithme est capable d'améliorer la répartition des échantillons à chaque fois qu'il doit reprendre l'optimisation pour une fenêtre de temps suite à un échec de l'optimisation d'une fenêtre de temps postérieure. Les auteurs utilisent une fenêtre glissante qui, si l'optimisation d'une fenêtre de temps échoue, permet de reprendre l'optimisation que pour les fenêtres de temps correspondant aux cinq secondes d'animation qui précédent celle pour laquelle l'optimisation a échouée (ce qui correspond à 50 fenêtres de temps). Finalement, au lieu de sélectionner les meilleurs échantillons pour chaque fenêtre pour initialiser l'optimisation de la fenêtre suivante, le contrôleur utilise un nouvel échantillon correspondant à la moyenne des meilleurs échantillons. Cela permet d'obtenir une animation finale plus fluide en réduisant le bruit introduit par l'optimisation dans l'animation finale.

Une autre modification de ce contrôleur a été proposée par Greer [Gre16] pour le cas de mouvements cycliques. Ces travaux utilisent le caractère cyclique des mouvements à produire pour pouvoir réutiliser les échantillons entre les différents cycles du mouvement. Cette technique permet, tant que les perturbations appliquées au personnage ne sont pas trop importantes, de ne pas avoir à recalculer les échantillons. Ceci permet d'obtenir un contrôleur capable de générer le mouvement en temps réel une fois le premier cycle calculé.

### 2.4.5 Support au calcul

Il est intéressant de remarquer que le système de commande prédictive utilisé dans la version originelle du SIMBICON ne correspond à aucune des fonctionnalités présentées précédemment (équilibre, contrôle de la vitesse, transition entre différents mouvements). Comme nous l'avons souligné dans la section 2.4, l'utilité du SIMBICON est de rendre possible l'utilisation de gains faibles ce qui permet d'éviter la génération d'animations à l'aspect robotique. Cependant, ce système a deux défauts majeurs : une phase d'apprentissage et l'impossibilité de modifier le mouvement une fois l'apprentissage effectué. En effet, vu que le système ne fait qu'appliquer les moments qui ont été observés par le passé, il lui est impossible de s'adapter aux changements d'environnement ou aux modifications de paramètres de haut-niveau tel que la vitesse désirée. Parmi les systèmes présentés dans le GENBICON de Coros et al. [CBV10], on trouve un système de compensation de gravité. Ce système remplit une fonctionnalité similaire à celle de la commande prédictive : calculer automatiquement une partie des gains nécessaires au déplacement du personnage. Cependant, ce système est fondamentalement différent dans le sens où il est générique à n'importe quel mouvement.


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

Le système de compensation de gravité a pour but de calculer les moments à appliquer pour compenser les poids de tous les membres du personnage. De cette manière le personnage est dans un état d'apesanteur virtuelle ce qui permet l'utilisation de gains plus faibles pour les régulateurs PD qui n'auront pas à compenser le poids du personnage. La compensation se fait sur toutes les parties du personnage excepté sur celles de la jambe d'appui. Le calcul du moment à appliquer utilise une force virtuelle de manière similaire au système de *velocity tuning*. Lors de ses travaux préliminaires, Greer [Gre16] fait remarquer qu'il est également possible de calculer les moments à appliquer sur les articulations des jambes pour lesquelles le pied est en contact avec le sol. Une jambe en contact avec le sol ne supporte pas le poids des membres de cette jambe mais elle supporte le poids de toutes les autres parties du personnage. Cela permet de pouvoir également baisser les valeurs des gains utilisés quand une jambe est en phase d'appui.

## 2.5 Contrôle de personnage virtuel à basse fréquence

Un des points de focalisation des travaux de cette thèse est la création d'un contrôleur interactif et temps réel. L'avantage principal de l'animation basée sur la physique est sa capacité à produire nativement les interactions avec l'environnement. Si le contrôleur ne permet pas d'obtenir des résultats interactifs lorsqu'il est utilisé seul, les performances obtenues lorsque le personnage est placé dans un environnement contenant d'autres éléments que ce dernier seront probablement très faibles. Ce qui limite l'application d'un tel contrôleur dans le cas de systèmes interactifs.

Une méthode simple pour diminuer les ressources consommées par le contrôleur est de diminuer la fréquence à laquelle est effectuée la simulation physique. En effet, les calculs des collisions entre les corps rigides, qui composent la majeure partie des éléments présents dans la simulation physique, ont un coût en ressources de calcul constant pour toute fréquence de simulation. Donc si l'on divise par deux la fréquence de simulation, les calculs de collisions consommeraient environ deux fois moins de ressources de calcul, car effectués deux fois moins souvent. Ce gain de performance est encore plus important pour d'autres éléments physiques tels que des fluides. Certaines simulations de fluides utilisent des algorithmes se servant des pas de simulation plus longs (donc de fréquences faibles) pour mettre en place des algorithmes plus performants (section 3.2.2).

### 2.5.1 Animation temps réel

Plusieurs contrôleurs proposés au cours de ces dernières années obtiennent des résultats temps réel. La simplicité du SIMBICON [YLV07] permet d'obtenir des performances temps réel et le GENBICON [CBV10] est capable d'animer de multiples personnages simultanément tout en restant temps réel. Cependant, ces contrôleurs ont un défaut majeur, ils ne fonctionnent que pour des fréquences de simulation élevées (1000Hz et 2000Hz). Des travaux postérieurs [GY11 ; Gre16] ont observé une limite minimale entre 750Hz et 800Hz lors de l'utilisation de contrôleurs de type SIMBICON.

D'autres contrôleurs permettent d'utiliser des fréquences de simulation plus faibles. En restant dans le cadre des contrôleurs dans l'espace des articulations, le SAMCON [Liu+10] a été utilisé par Greer [Gre16] pour créer un contrôleur fonctionnant à une fréquence de 60Hz pour des mouvements cycliques utilisant seulement 0.62ms par pas de temps de simulation. Leur contrôleur profite du caractère cyclique du mouvement pour réutiliser les mêmes échantillons



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

entre les cycles successifs du mouvement, retirant ainsi la partie du contrôleur nécessitant le plus de calculs. Liu et al. [LPY16] proposent d'utiliser une phase d'optimisation hors ligne pour générer les échantillons pour un ensemble de mouvements qui sont utilisés dans un graphe de contrôle généré par cette même optimisation. Le résultat est un contrôleur fonctionnant à une fréquence de 200Hz dix fois plus rapide que le temps réel. Si l'on regarde du côté des contrôleurs optimaux, on trouve d'autres travaux qui obtiennent des résultats temps réel. Le contrôleur proposé par Muico et al. [Mui+09 ; MPP11] obtient une simulation stable à une fréquence de 120Hz. Bien que leurs résultats décrivent des temps de calcul de 6ms par pas de temps de simulation il est important de noter que la machine utilisée pour obtenir ces résultats n'est pas spécifiée et que les ressources de calcul disponibles ont fortement augmenté depuis. Les contrôleurs bénéficiant le plus de fréquences de simulation basses sont ceux utilisant un MPC. Les travaux de Hämäläinen et al. [Häm+14 ; HRL15] utilisent des fréquences de simulation extrêmement faibles, entre 20Hz et 30Hz. Mais malgré cela les contrôleurs produits ne sont toujours pas capables de générer les animations en temps réel. Le contrôleur combinant un MPC et des poses de références présenté par Han et al. [Han+16 ; HNS18] est capable de générer des animations en temps réel pour des fréquences similaires. Cependant, Il est important de noter que si l'animation est produite en temps réel, le contrôleur utilise la totalité des ressources de calcul disponibles.

Une des raisons pour laquelle peu de travaux utilisent des fréquences de simulation faibles est que celles-ci induisent des instabilités dans la simulation physique. En effet, des pas de temps plus larges baissent le niveau de précision des calculs physiques. En particulier, les contacts entre les pieds du personnage et le sol se retrouvent moins stables car les calculs de détection de collisions et les calculs d'intensité des forces physiques sont réalisés moins fréquemment.

### 2.5.2 Traitement des contacts

Lorsque l'on utilise des fréquences de simulation plus faibles, les contacts entre le personnage et le sol sont réalisés sur moins de pas de temps de simulation. De ce fait, on observe de plus grandes variations au niveau des forces de réaction du sol. Ces variations introduisent des instabilités au sein du personnage. En particulier un des problèmes observables est l'apparition de discontinuités au niveau des contacts entre le pied et le personnage. Les divers travaux qui optent pour des fréquences de simulations faibles utilisent une solution pour pouvoir réduire ces instabilités. Il existe deux approches pour résoudre le problème des contacts : modifier le moteur physique ou créer un contrôleur compensant les instabilités.

**Modification du moteur physique** Parmi les solutions agissant directement sur le moteur physique, le contrôleur proposé par Han et al. [Han+16 ; HNS18] obtient les meilleurs résultats. La stabilité de leur système est due à l'utilisation d'un moteur physique implémentant l'algorithme de gestion des collisions de MuJoCo [Tod14]. En s'appuyant sur les travaux publiés dans [Liu+13], Liu et al. [LPY16] intègrent dans le moteur physique ODE un système d'amortissement implicite des moments. Une autre solution proposée par Jain et Liu [JL11] est d'utiliser un corps mou au niveau des pieds ce qui leur permet d'obtenir des contacts continus et stables. Cependant il ne démontre la viabilité de cette approche que pour des fréquences de simulation élevées.

**Compensation par le contrôleur** Les travaux présentant des solutions fonctionnant pour des moteurs physiques classiques (ODE, PhysX, ...) sont moins communs. Muico et al. [Mui+09] intègrent les forces de contacts directement dans leur optimisation. Bien qu'intéressante, cette solution requiert de connaître les équations utilisées pour calculer les forces appliquées et


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

est donc limitée à des moteurs physiques open source. Greer [Gre16] utilisent le système d'optimisation par échantillonage du contrôleur SAMCOM [Liu+10] pour obtenir des contacts stables. Pour accélérer la vitesse de convergence de l'optimisation, les auteurs mettent en place une optimisation hors-ligne qui permet d'améliorer la qualité des mouvements de référence. Enfin, Da Silva et al. [Sil+17] proposent d'appliquer un moment supplémentaire sur le pied en phase d'appui si le moment appliqué sur la cheville dépasse un seuil défini par l'utilisateur. Ce moment supplémentaire permet d'assurer que la totalité du pied reste en permanence en contact avec le sol. Cette technique, bien que non physiquement réaliste, car elle utilise un moment externe, permet à l'utilisateur de contrôler directement la stabilité des contacts sans nécessiter de ressources de calcul supplémentaires.

## 2.6 Bilan

Il existe de nombreuses approches possibles pour réaliser l'animation basée physique d'un personnage virtuel. La méthode de contrôle permettant actuellement d'obtenir les meilleurs résultats est l'approche qui se base sur le contrôle optimal. Notre étude montre cependant que, compte tenu de l'importante quantité de ressources de calcul nécessaires pour l'obtention d'une exécution en temps réel, ces approches n'ont pas actuellement la possibilité d'être intégrées dans un monde virtuel complexe.

Des travaux récents proposant des approches basées sur des réseaux de neurones ont réussi à produire des contrôleurs capables de gérer les mouvements pour des personnages possédant de nombreux degrés de liberté. Cependant leur exigence en ressources de calculs limite leurs portés.

La méthode la plus courante utilisée pour obtenir un contrôleur temps réel robuste est l'utilisation de pas de temps plus longs permettant l'application d'algorithmes améliorant la stabilité des contacts. Plusieurs méthodes basées sur le contrôle optimal permettent ainsi de simuler des mouvements complexes (danse, arts martiaux) tout en restant interactives. Cependant, les contrôleurs proposés dans ces travaux requièrent soit l'utilisation de moteurs physiques particuliers permettant des contacts plus stables tels que MuJoCo, soit l'utilisation de versions modifiées des moteurs physiques classiques. La non-disponibilité de tels moteurs fait que les moteurs physiques plus classiques (ODE, PhysX, ...) sont de nos jours encore plus présents dans la majeure partie des recherches du domaine de l'animation basée physique.

Les contrôleurs utilisant le contrôle dans l'espace des articulations offrent un contrôle aisé de l'animation obtenue, des temps de calcul faibles et une mise en oeuvre aisée. Cependant ceux-ci sont limités par leur incapacité à simuler des mouvements complexes et hautement dynamiques. Comme cette thèse s'intéresse aux les interactions entre le personnage et son environnement, et non à la simulation de mouvements complexes, cette limitation n'est pas une entrave majeure pour nos travaux. De nombreux contrôleurs actuels utilisent cette approche, notamment le SIMBICON [YLV07] pour lequel plusieurs extensions ont été proposées pour améliorer l'équilibre et le contrôle de la vitesse.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

# État de l'art : Simulation de fluide

# 3

Dans le domaine de l'animation, la simulation de fluide est une technique permettant d'obtenir une animation détaillée et réaliste pour des fluides allant de l'eau à des nuages de fumée. Les méthodes de simulation de fluide utilisées pour réaliser des animations reposent sur l'application des équations de fluides incompressibles de Navier-Stokes. Ces équations décrivent le comportement physique d'un fluide non compressible.

$$\nabla \cdot \vec{u} = 0 \iff \frac{D\rho}{Dt} = 0 \tag{3.1}$$

$$\frac{D\vec{u}}{Dt} + \frac{1}{\rho}\nabla p - \nu\nabla^2\vec{u} = \vec{g} + \vec{F_{ext}} \tag{3.2}$$

L'équation 3.1 exprime le caractère incompressible du fluide. $\nabla \cdot \vec{u}$ est l'opérateur divergence du champ de vitesse $\vec{u}$ et représente la variation de quantité de matière en un point du fluide. Si le fluide ne peut pas être compressé cela veut dire que la quantité de matière reste constante et donc sa variation est nulle. Ceci peut être également exprimé par le fait que la densité $\rho$ du fluide est constante. L'équation 3.2 exprime l'effet des forces externes (la gravité) et des forces internes (la viscosité et la pression) sur le fluide. Cette équation est communément appelée l'équation de bilan de la quantité de mouvement. Dans cette équation $\frac{D(.)}{Dt}$ est la dérivée matérielle et $u$, $\rho$, $p$, et $\nu$ représentent respectivement, la vitesse, la densité, la pression et la viscosité cinématique. La dérivée matérielle est une dérivée dans l'espace-temps. Elle donne la variation de la variable étudiée dans le temps en un point de l'espace et également la variation de la variable étudiée due aux changements du milieu en ce point (ex : changement de température dû au remplacement de l'air chaud par de l'air froid). La viscosité cinématique affecte l'accélération et est reliée à la viscosité dynamique par $\nu = \mu/\rho$ avec $\mu$ le coefficient de viscosité dynamique. Il est important de noter que la gravité n'est pas nécessairement la seule force externe appliquée. D'autres forces externes ($\vec{F_{ext}}$) sont couramment utilisées pour représenter des phénomènes tels que la tension de surface ainsi que les forces de contacts lorsqu'un solide est en interaction avec le liquide. Cette équation est généralement simplifiée en retirant le terme de viscosité. En effet, dans la majeure partie des cas simulés, la viscosité ne joue qu'un rôle mineur et est donc retirée de l'équation. En pratique un effet similaire à la viscosité pourra être introduit en simplement réduisant les vitesses. On obtient donc une version simplifiée des équations, 3.3 et 3.4, qui sont appelées équations d'Euler.

$$\nabla \cdot \vec{u} = 0 \iff \frac{D\rho}{Dt} = 0 \tag{3.3}$$

$$\frac{D\vec{u}}{Dt} + \frac{1}{\rho}\nabla p = \vec{g} + \vec{F_{ext}} \tag{3.4}$$

Une présentation détaillée de ces formules est donnée dans un tutoriel de Briston et Müller [BM07].



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

## 3.1 Modèles de fluide

Il existe deux représentations mathématiques, eulérienne et lagrangienne, qui mènent, avec une représentation mixte, à trois approches possibles pour la représentation d'un fluide.

**Représentation eulérienne** Lorsque l'on utilise une représentation eulérienne, l'espace de simulation est découpé selon une grille fixe et des quantités de fluides se déplacent entre les cases de cette grille. La démarche eulérienne consiste à étudier le mouvement du fluide à des endroits fixes. Une analogie simple peut être faite avec le système utilisé pour récolter les données météorologiques. Un système eulérien correspond aux différentes stations météorologiques fixes avec l'air qui se déplace entre celles-ci. Ce type de modèle présente un avantage pour la simulation de fluide : une intégration spatiale simple lors de la résolution des équations. En effet, la résolution des équations d'Euler par un système numérique implique l'intégration des différentes dérivées spatiales qu'elles contiennent. D'un point de vue technique, il est aisé de déterminer les voisins dans une grille régulière : il suffit simplement d'itérer sur les identifiants des cases. Cependant, cet avantage a un inconvénient, dans une représentation eulérienne il est plus compliqué de calculer l'accélération du fluide aux différents endroits en particulier lorsque le fluide interagit avec des objets solides. Ces deux caractéristiques permettent aux solveurs de fluide eulérien d'obtenir une très bonne précision physique mais requièrent une grande quantité de ressources de calcul. Certains travaux permettent une utilisation interactive. Parmi ceux-ci, Chentanez et Müller [CM11] proposent d'utiliser une grille avec plusieurs niveaux de détail. Les cellules éloignées de la surface du liquide sont regroupées en cellules plus grandes sur lesquelles des versions simplifiées des calculs physiques sont utilisées. Cette technique permet d'économiser de la puissance de calcul où la simulation n'a pas d'impact visuel important. Une présentation détaillée des algorithmes utilisés par les solveurs de fluide eulérien peut-être trouvée dans état de l'art proposé par Briston [Bri15].

**Représentation lagrangienne** Dans une représentation lagrangienne, le fluide doit être représenté sous forme de particules élémentaires de volume constant. Dans ce cas, on s'intéresse au mouvement de ces particules au cours du temps. Les équations d'Euler seront utilisées pour calculer l'impact de chaque particule sur son voisinage. Le principe est similaire à celui d'une piscine de boules en plastique, nous ne sommes plus attachés à l'espace de simulation mais nous sommes maintenant reliés au fluide lui-même. Pour reprendre notre analogie météorologique, cette méthode correspond à utiliser une flotte de ballons météorologiques, les points de capture se déplacent avec le milieu étudié. Le volume des particules étant constant, il n'y a pas besoin de faire de calculs de transferts de matière entre-elles. Ceci permet d'obtenir de meilleures performances que les approches eulériennes, en particulier lors de la présence d'objets dynamiques dans la simulation. Cependant, cette représentation ajoute le problème de la détermination du voisinage d'une particule, rendant ainsi le calcul des dérivées spatiales plus complexe. En particulier, le nombre de particules dans le voisinage d'une particule donnée n'étant pas constant, les calculs effectués contiennent des erreurs numériques qui dégradent la validité physique des résultats. La méthode de simulation lagrangienne principalement utilisée est le *Smoothed Particles Hydrodynamics* (SPH). Cette méthode de simulation a été originellement utilisée pour les modèles stellaires compressibles [GM77]. L'ajout du flux incompressible a été proposé par Monaghan et Humble [Mon92]. Ihmsen et al. [Ihm+14b] répertorient les différents algorithmes de simulation de fluide lagrangienne. Les principaux algorithmes récents sont présentés dans la section 3.2. L'approche lagranrienne est communément utilisée par des applications commerciales comme le moteur physique PhysX (NVIDIA


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

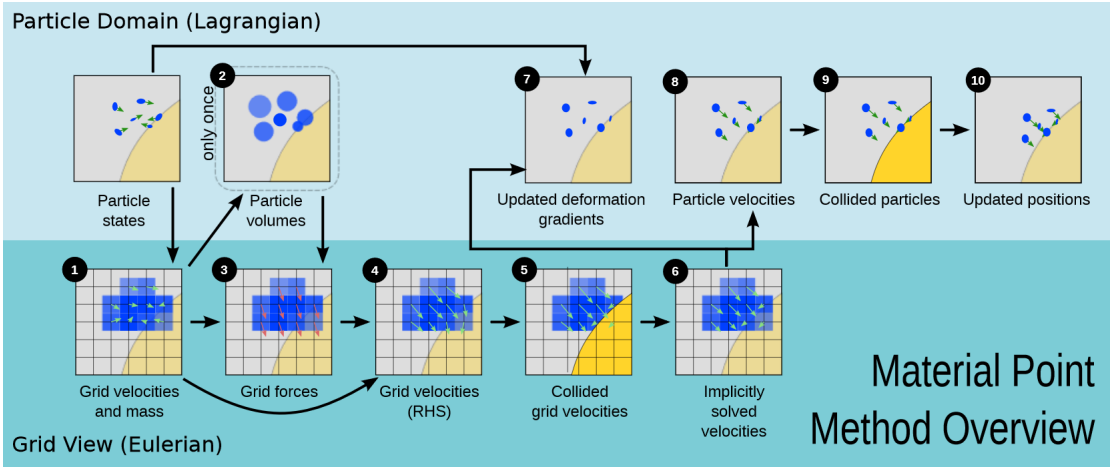



**Figure 3.1.:** Processus de simulation d'un fluide par la méthode hybride *Material Point Method* (MPM) [Sto+13]. On peut observer la séparation entre la gestion du déplacement du matériau et des collisions dans l'espace lagrangien et le calcul de la dynamique interne du fluide dans l'espace eulérien.

Flex), mais également dans des logiciels d'animation, tels que Blender (Blender-fluid) ce qui permet de proposer des simulations de fluide interactives.

**Représentations hybrides** Il ressort que le modèle eulérien est avantageux pour le calcul des valeurs de pression du fait d'un voisinage exact alors que le modèle lagrangien offre une meilleure gestion de l'étape de déplacement du matériau. Les méthodes hybrides cherchent à bénéficier des avantages des deux méthodes en adoptant les deux modèles et en passant d'une représentation à l'autre suivant l'étape de la simulation. Les deux méthodes hybrides principalement utilisées sont la méthode *Particle-In-Cell* (PIC) [HW65] et la méthode *FLuid Implicit Particles* (FLIP) [BR86]. La différence majeure entre ces deux méthodes se trouve dans les équations utilisées pour convertir le fluide entre la représentation eulérienne et lagrangienne. L'algorithme PIC permet une transition stable mais cause une dissipation importante de l'énergie dans le fluide, c-à-d que les perturbations, telles que des vagues, s'atténuent très rapidement. À l'inverse, l'algorithme FLIP préserve l'énergie du fluide mais mène à des simulations plus bruitées voire instables. Récemment, Jiang et al. ont proposé une méthode, qu'ils ont appelé *Affine Particle-In-Cell* (APIC) [Jia+15], qui utilise de nouvelles équations pour effectuer la transition ce qui permet un résultat stable et sans dissipation excessive d'énergie. Stomakhin et al. [Sto+13] utilisent un modèle hybride pour simuler des fluides particuliers (neige). Leur illustration du modèle utilisé, *Material Point Method* (MPM), offre une visualisation du processus de transition entre les représentations lagrangienne et eulérienne (figure 3.1). Si une représentation mixte offre de réels avantages, elle introduit un coût supplémentaire en calcul et mémoire. En effet, devoir maintenir deux structures de données implique de devoir stocker les données en double si l'on ne veut pas avoir à allouer la mémoire à chaque pas de temps de simulation. De plus, la transition d'une structure à l'autre a un coût en calcul.

Compte tenu des résultats obtenus et de la possibilité de simuler des fluides particuliers tels que la neige ou le sable, un modèle hybride est souvent utilisé dans les films d'animation (ex : simulation de la neige dans Frozen [Sto+13], simulation de l'océan et de la lave dans Moana [Jia+15]) ou des logiciels d'animation tels que Maya (Bifröst Fluids).



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

Pour résumer, le modèle eulérien permet d'obtenir un fluide avec un niveau de réalisme très élevé. Cependant, les temps de calculs élevés ainsi que les difficultés du modèle à supporter les interactions avec des objets dynamiques ont limité l'application de ce modèle. À l'inverse, les performances interactives obtenues avec le modèle lagrangien ainsi que sa robustesse aux objets dynamiques ont favorisé son implémentation dans des applications interactives pour lesquelles l'exactitude physique de la simulation n'a pas une importance primordiale. Enfin, les modèles hybrides qui permettent d'obtenir le niveau de réalisme du modèle eulérien ainsi que la facilité d'interaction de l'approche lagrangienne sont employés pour les applications où la qualité visuelle du résultat est prioritaire sur l'interactivité du système.

Dans notre cas, nous nous intéressons aux interactions entre un personnage en mouvement et un fluide dans un contrôleur interactif. Par conséquent, l'approche lagrangienne est la plus adaptée.

## 3.2 Smoothed Particles Hydrodynamics

Rappelons que, dans le modèle lagrangien, le fluide est représenté par un ensemble de particules élémentaires et les propriétés dynamiques de chaque particule (densité, pression) sont déterminées à partir des positions et vitesses des autres particules présentes dans la simulation. Avant de présenter les différentes variations de l'algorithme *Smoothed Particles Hydrodynamics* (SPH), il est important de savoir comment déterminer quelles sont les particules à prendre en compte quand l'on veut calculer des propriétés physiques en un point du fluide. En effet, seules les particules assez proches du point étudié ont un impact non négligeable sur le résultat. Effectuer, pour chaque particule, les calculs physiques en considérant la totalité des particules présentes dans la simulation augmenterait grandement les temps de calcul nécessaire sans apporter une amélioration notable de la simulation. Par conséquent, les algorithmes SPH utilisent une fonction de noyau qui permet de déterminer l'impact d'une particule sur les autres particules en fonction de leur distance.

### 3.2.1 Noyau

Le noyau d'une particule est une zone sphérique (circulaire en 2D) centrée sur celle-ci contenant toutes les particules ayant un impact significatif sur les calculs physiques. Ce noyau est défini par un rayon $h$ et une fonction $W$. La fonction $W$ décrit l'évolution de l'importance d'une particule en fonction de sa distance par rapport au centre du noyau. Le rayon $h$ indique la distance maximale entre les deux particules pour que leur interaction soit considérée comme significative. Les fonctions noyaux sont généralement de la forme :

$$W_{ij} = W\left(\frac{\|\vec{x_j} - \vec{x_i}\|}{h}\right) = W(q) = \frac{1}{h^d} f(q) \tag{3.5}$$

Avec $i$ et $j$ deux particules, $x_i$ et $x_j$ leur position respectives, et $d$ le nombre de dimensions (ex : 3 pour une simulation en 3D). Il est important de noter qu'il n'y a pas de consensus sur la fonction $f(q)$ à utiliser pour une simulation SPH. Chaque auteur décide d'utiliser une (ou plusieurs) fonction spécifique pour bénéficier de propriétés mathématiques adaptées à


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

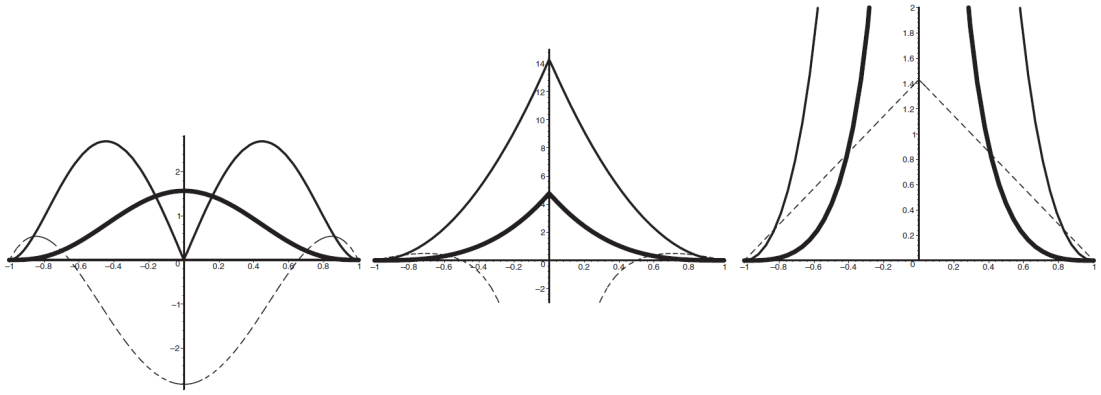


**Figure 3.2.:** Exemples de fonction noyau pour une simulation SPH utilisées par Müller et al. [MCG03]. De gauche à droite : $W_{poly6}$, $W_{spiky}$, $W_{viscosity}$. Les lignes épaisses montrent les fonctions noyau, les lignes fines leurs dérivées et les lignes en pointillés leurs Laplacien.

sa méthode de simulation. Une fonction $f(q)$ communément utilisée est la fonction *cubic spline* :

$$f(q) = \frac{8}{\pi} \begin{cases} 1 - 6q^2 + 6q^3, & 0 \leq q < 0.5 \\ 2(1-q)^3, & 0.5 \leq q < 1 \\ 0, & q \geq 1 \end{cases} \tag{3.6}$$

Selon la propriété physique calculée (densité, position...), il peut être intéressant d'utiliser différentes fonctions de noyau. La figure 3.2 illustre les trois fonctions de noyau ($W(q)$) utilisées par Müller et al. [MCG03] ainsi que leurs dérivées et leurs Laplacien. La forme généralement désirée pour le calcul de la densité est une gaussienne ce qui justifie le choix de la fonction $W_{cubic\_spline}$ ou de $W_{poly6}$. Les calculs de pression dépendent, eux, de la dérivée de la fonction noyau. Or, la dérivée d'une gaussienne tend vers zéro lorsque la valeur de $q$ tend vers zéro. Dans une simulation de fluide, cela signifie que les particules proches ont un impact sur la pression plus faible que les particules à une moyenne distance, ce qui serait absurde. Pour la pression on devrait avoir une variation de la dérivée inversement proportionelle à sa distance, propriété observée dans la fonction $W_{spiky}$. Les calculs de viscosité dépendent du Laplacien de la fonction noyau. La fonction noyau $W_{viscosity}$ est utilisée par Müller et al. pour obtenir une viscosité linéaire en fonction de la distance séparant les particules.

La fonction de noyau annule l'influence des particules trop éloignées. Cependant, elle nécessite toujours de vérifier sur la totalité des particules lesquelles se trouvent à une distance inférieure à $h$ de la particule étudiée. Pour éviter d'avoir à considérer toutes les particules, il est possible de mettre en place une structure de recherche spatiale. Une solution classique est de diviser l'espace de simulation selon une grille uniforme de taille $h$. Cette structure assure que toutes les particules se trouvant à une distance inférieure à $h$ se trouvent soit dans la même cellule que la particule étudiée, soit dans l'une des 26 cellules l'encadrant. On peut également utiliser une table de hachage spatiale. Une comparaison des performances de diverses méthodes a été présentée par Ihmsen et al. [Ihm+11].



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

### 3.2.2 Évolution des algorithmes SPH

L'algorithme 1 illustre le processus classique d'une simulation physique SPH. L'algorithme commence en cherchant les voisins de chaque particule. Puis, il calcule pour chaque particule les valeurs de la densité $\rho_i$ et de la pression $p_i$. Pour déterminer l'accélération de chaque particule, le système calcule toutes les forces appliquées. Ces forces correspondent aux forces de pression, qui dépendent du champ de pression précédemment calculé, des forces de fluide estimées (viscosité, tension de surface,...) et des forces externes (gravité). La somme de ces forces donne l'accélération de la particule et permet de calculer les nouvelles vitesses et positions.

**Algorithm 1** SPH

```
1:  while animating do
2:      for all i do
3:          find Neighborhood N_i(t)
4:      for all i do
5:          compute density ρ_i(t)
6:          compute pressure p_i(t)
7:      for all i do
8:          compute Forces F^{p,v,g,ext}(t)
9:      for all i do
10:         compute new velocity v_i(t + 1)
11:         compute new position x_i(t + 1)
```

Bien que cette approche ait permis à Müller et al. [MCG03] d'obtenir un système interactif, ceci n'est possible que pour un faible nombre de particules (entre 3000 et 5000). Le problème principal est que cette méthode introduit trop d'instabilités dans la simulation. En particulier, si des pas de temps élevés sont utilisés, la condition d'incompressibilité du fluide peut ne pas être respectée.

Solenthaler et Pajarola proposent une méthode itérative pour assurer le caractère incompressible du fluide simulé : le *Predictive-Corrective Incompressible SPH* (PCISPH) [SP09]. Cette solution présente deux différences majeures avec l'algorithme SPH de base. La contribution majeure du PCISPH est l'ajout d'un processus prédictif-correctif qui permet d'assurer que la densité du fluide reste constante. Ce processus fonctionne en calculant de manière itérative les forces de pression au lieu de tenter de les calculer en une seule fois. Elle a pour condition d'arrêt une limite maximale de l'erreur sur la densité moyenne par rapport à la densité désirée. Cette technique requiert plus de temps de calcul pour un pas de simulation donné mais augmente fortement la stabilité de la simulation en assurant le caractère incompressible du fluide. Cela permet d'augmenter la durée des pas de temps de la simulation ce qui réduit les ressources nécessaires et offre une meilleure qualité que les versions antérieures. La seconde différence est qu'elle calcule toutes les forces autres que les forces de pression et détermine des positions et vitesses intermédiaires des particules en appliquant ces forces. Les forces de pression sont ensuite calculées à partir de ces nouvelles valeurs. Cette technique est appelée *splitting* et est déjà présente dans certaines versions antérieures de simulateurs SPH [Ihm+14b]. Elle permet d'augmenter la stabilité du fluide.

Ihmsen et al. proposent en 2013 l'algorithme *Implicit Incompressible SPH* (IISPH) [Ihm+14a]. Leur système s'appuie sur une nouvelle discrétisation des équations de dynamique des fluides pour calculer les forces de pression entre les particules. Une méthode de calcul des forces de pression similaire est utilisée dans le solveur *Incompressible SPH* (ISPH) [Ihm+14b].


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

L'algorithme IISPH combine cette nouvelle discrétisation à un processus prédictif-correctif similaire à celui présent dans le PCISPH. Le résultat est un solveur qui donne de meilleures performances et une meilleure stabilité physique.

Bender et Koschier proposent d'utiliser une autre approche pour améliorer les performances du PCISPH dans leur *Divergence-Free SPH* (DFSPH) [BK15]. Leur solveur de fluide assure que la divergence du fluide reste nulle au cours de la simulation en plus d'assurer une densité constante. En pratique, le DFSPH emploie un second processus prédictif-correctif qui vérifie que la valeur de la divergence du fluide reste négligeable. Les auteurs proposent également une nouvelle discrétisation des équations de Navier-Stokes qui permet de limiter les calculs nécessaires factorisant une partie commune aux deux processus prédictifs-correctifs qui reste constante au cours d'un pas de simulation. L'algorithme du DFSPH est présenté en détail dans la section 5.1. Malgré leur utilisation de deux boucles de calcul, leur solveur obtient de meilleures performances que tous les algorithmes présentés précédemment. La raison est qu'assurer une divergence nulle améliore la stabilité du fluide et donc diminue aussi le nombre d'itérations nécessaire à l'obtention d'une densité constante. Une comparaison des deux algorithmes les plus performants, le IISPH et le DFSPH, par Barronea et al. [Bar+17] montre des résultats similaires à ceux présentés dans l'article de Bender et Koschier [BK15].

### 3.2.3 Viscosité et tension de surface

En plus d'être incompressible un fluide a de nombreuses autres propriétés telles que sa viscosité. Les discrétisations des équations de Navier-Stokes utilisées par les modèles lagrangiens ne permettent de représenter que le caractère incompressible du fluide. Pour simuler l'impact des propriétés d'un fluide ne dépendant pas des forces de pression on applique des forces externes supplémentaires sur les particules.

#### Viscosité

La viscosité d'un fluide est la propriété la plus importante après l'incompressibilité. Ajouter de la viscosité au fluide est non seulement important pour le réalisme de la simulation mais également pour sa stabilité physique. En effet, ajouter de la viscosité au fluide permet de diminuer les variations locales de vitesse et d'obtenir un fluide uniforme. Il existe plusieurs approches pour ajouter de la viscosité à un fluide. Le premier modèle de viscosité artificielle a été introduit par Lucy [Luc77]. Les équations proposées ont été utilisées dans des travaux plus récents [Mon94 ; BT07]. Cependant, l'équation utilisée est complexe et engendre un coût non négligeable en temps de calcul tout en ne permettant de reproduire que l'effet de viscosités peu élevées. Monaghan propose une autre formulation de la viscosité permettant d'éviter les intersections entre les particules dans la simulation (XSPH) [Mon89]. La méthode actuellement la plus utilisée est une variation du XSPH proposée par Schechter et Bridson [SB12] permettant d'alléger les calculs nécessaires rendant le calcul de la viscosité extrêmement rapide dans leur simulation. Des travaux récents proposent des solutions permettant de simuler des fluides avec de très hautes viscosités [Pee+15 ; BK17]. Ces solutions ont cependant la limitation de demander des ressources de calcul beaucoup plus importantes que les approches se limitant aux fluides à faible viscosité.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

#### Tension de surface

Une autre interaction entre les particules communément ajoutée à une simulation SPH est la tension de surface. La tension de surface est un phénomène qui attire entre elles les particules de fluide se trouvant en contact avec l'air. En pratique cela correspond à créer une pellicule à la surface du fluide. Ce phénomène est extrêmement important dans le cas où seule une fine couche de fluide est simulée. Deux approches principales ont été proposées. La première consiste à ajouter une force de cohésion qui attire les particules en fonction de la courbure du fluide autour de chaque particule [BT07 ; AAT13]. Cette technique permet d'obtenir un résultat correct et rapide dans des situations relativement simples (ex : liquide sur surface plane ou chute de gouttes de liquide). Ces deux approches sont habituellement utilisées dans les travaux ne se focalisant pas précisément sur le réalisme physique des phénomènes de tension de surface. La seconde solution est d'utiliser la méthode du *ghost SPH* proposée par Schechter et Bridson [SB12]. Cette technique consiste à faire apparaître des particules virtuelles au niveau de la surface libre du fluide. Ces particules virtuelles servent à représenter la présence de l'air autour du fluide simulé sans avoir à simuler le volume d'air présent dans la simulation. Le *ghost SPH* permet d'obtenir de très bons résultats même dans des situations complexes telles que l'écoulement d'une fine couche de fluide sur une sphère. Cependant, cette technique augmente grandement les temps de calcul nécessaires ce qui la rend inapplicable dans des applications interactives.

Il existe d'autres interactions qui peuvent être ajoutées pour améliorer le résultat des simulations. Parmi les travaux récents, on trouve notamment l'amélioration des turbulences par la préservation du champ de vorticité [Ben+17].

### 3.2.4 Conditions aux limites

Une dernière problématique de la simulation de fluide SPH est la gestion des interactions avec des objets solides. La méthode fréquemment utilisée pour permettre les interactions entre un solide et le fluide est de discrétiser le solide en particules. Plusieurs techniques ont été utilisées pour calculer les forces appliquées par les particules de solide sur les particules de fluide. La première technique consiste à appliquer une force de répulsion, dont l'intensité est contrôlée par un coefficient de rigidité, quand une particule de fluide s'approche d'une particule de solide [MK09]. Ihmsen et al. [Ihm+14b] proposent une revue détaillée des travaux utilisant ce type de conditions aux limites. Cette technique est extrêmement difficile à calibrer pour obtenir un résultat stable. Une seconde technique consiste à intégrer les particules de solide à l'intérieur des calculs des forces de pression. Pour cette technique il est nécessaire d'utiliser une méthode de discrétisation particulière du solide pour que les particules de solide aient des propriétés, en particulier une masse, permettant une simulation stable. La méthode initiale utilisée pour discrétiser les objets est de représenter leur surface par plusieurs couches de particules statiques de fluide [DK01]. Akinci et al. [Aki+12] proposent une nouvelle méthode pour discrétiser un solide avec seulement une seule couche de particules et de leur attribuer une masse en fonction du volume de solide qu'elles représentent. Pour éviter d'avoir à calculer les valeurs de pression au niveau des particules de solide, la plupart des travaux considèrent qu'une particule de solide a la même pression que la particule de fluide avec laquelle elle interagit. Bien que permettant d'obtenir des résultats rapides, cette approche a récemment été remise en question car elle cause des instabilités. En effet, une particule solide n'aura pas la même valeur de pression selon la particule de fluide avec laquelle elle agit (ce qui est physiquement impossible). Parmi les solutions proposées, l'utilisation d'un *moving least square* [GT17] est une méthode simple pour assurer que la direction des forces de


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

pression au niveau de bordures planes est perpendiculaire à la bordure. Band et al. [Ban+18] proposent d'effectuer directement les calculs de pression sur les particules solides de manière similaire aux particules de fluide dans un IISPH. Leur solution permet d'augmenter largement la stabilité de l'IISPH et d'obtenir un gain de performance jusqu'à un facteur de cinq pour certaines simulations.

## 3.3 SPH sur le GPU

L'un des intérêts du modèle SPH est qu'il est hautement parallélisable. En effet, chaque propriété physique peut être évaluée en simultané sur toutes les particules du fluide. De ce fait, des travaux proposent une implémentation CPU parallélisée [Ihm+14b ; BK15]. Le nombre de particules dans une simulation étant généralement très élevé (parfois plusieurs millions) il est possible de profiter du grand nombre de coeurs disponibles sur une carte graphique pour fortement diminuer le temps de calcul d'un solveur SPH. Ceci explique la présence de nombreux travaux présentant des versions spécialisées pour le calcul GPU de tous les algorithmes de simulation SPH, à l'exception du très récent DFSPH. Bien que le modèle SPH soit parallélisable de par sa définition, sa mise en pratique demande un investissement au niveau de la structure de donnée utilisée. Certaines techniques sont utilisées dans plusieurs travaux :

— **Structure de tableaux :** Pour faire le choix entre une structure de tableaux (SoA) ou un tableau de structure (AoS) il faut analyser laquelle de ces deux structures offre une meilleure utilisation du cache. Le calcul de chacune des propriétés du fluide pour une particule n'utilise qu'une partie des informations de la particule et utilise les mêmes informations des particules voisines. De ce fait, il est plus intéressant de stocker chaque information indépendamment. Il est donc plus intéressant d'utiliser une SoA pour le stockage des données [NCX15].

— **Recherche de voisins :** Il y a deux problématiques principales pour l'algorithme de recherche de voisins : le type de structure de recherche et l'indice à associer à chacune des cellules de cette structure.

  En informatique graphique il existe de nombreuses structures de données permettant d'accélérer la recherche d'un élément dans un espace donné. Trois structures sont fréquemment utilisées : les grilles uniformes, les tables de hachages et les arbres quaternaires. Ihmsen et al. [Ihm+11] comparent l'utilisation d'une grille uniforme et d'une table de hachage pour une implémentation parallèle CPU. Leurs résultats montrent que les temps obtenus sont similaires pour un nombre relativement faible de particules (<130k). Il est intéressant de noter que la grille uniforme prend plus de temps à la création mais moins à l'utilisation. Cela signifie que pour des algorithmes récents, où il y a plus d'utilisation pour chaque pas de temps, les performances de la grille uniforme seront meilleures que celles d'une table de hachage (tant que l'on dispose d'assez de mémoire GPU pour stocker la grille uniforme sans avoir à la recréer). Xia et Liang [XL16] comparent l'utilisation d'une grille uniforme et d'un arbre quaternaire et obtiennent de meilleures performances avec cette seconde approche pour un scénario d'écoulement de rivière.

  Il existe différents indices qui peuvent être employés pour identifier les cellules de la structure choisie. Le choix d'un indice spécifique peut être justifié par les propriétés spécifiques à cet indice que ne possède pas un indice linéaire classique (figure 3.3b). En particulier, l'indice de Morton (figure 3.3a) possède une meilleure cohérence spatiale,



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

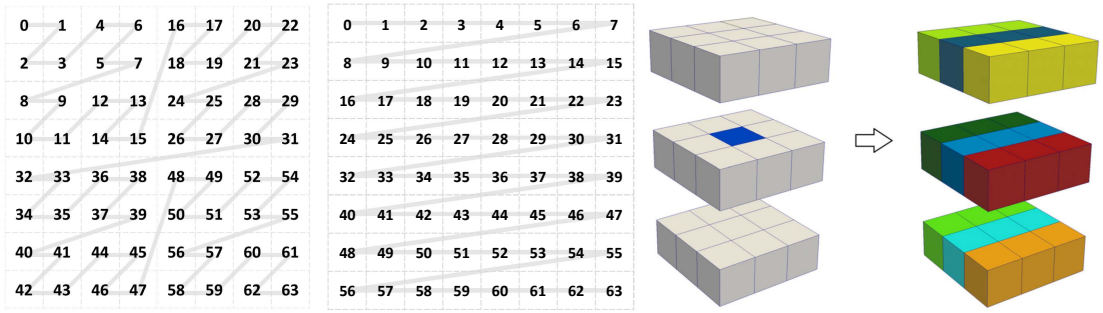



**Figure 3.3.:** (a) Représentation de l'utilisation d'un indice de Morton. (b) Représentation de l'utilisation d'un indice spatial. (c) Groupement de cellules lors de l'utilisation d'un indice spatial [DCG11]

c.à.d que des cellules proches dans l'espace possèdent des indices proches. Dans l'article d'Ihmsen et al. [Ihm+11] l'utilisation d'un indice de Morton permet de réduire de 30% le temps necessaire à la construction d'une grille uniforme par rapport à un indice linéaire. Bien que l'indice de Morton soit utilisé dans certains travaux récents [NCX15 ; XL16], Goswami et al. [GEF15] utilisent un indice linéaire pour éviter d'avoir à réaliser les calculs bit à bit couteux nécessaires à l'indice de Morton. Une des raisons de choisir un indice classique avec une grille uniforme est qu'il permet d'explorer les 27 cellules contenant une particule et son voisinage par groupe de trois cellules qui possèdent des indices consécutifs, ce qui assure que les particules contenues dans ces groupes de cellules sont alignées en mémoire (figure 3.3c) [DCG11].

Comparer les facteurs d'accélération entre une implémentation parallèle CPU et une implémentation GPU obtenus dans les différents travaux existants est difficile. En effet, suivant l'algorithme SPH utilisé ou le matériel utilisé on obtient des facteurs d'accélération différents. Goswami et al. [GEF15] rapportent des facteurs d'accélération allant de 2 à 6 suivant le nombre de particules pour l'algorithme IISPH.

## 3.4 Bilan

Nous avons vu qu'il existe deux modèles pour la simulation de fluide : eulérien ou lagrangien. L'approche eulérienne permet d'obtenir un niveau de réalisme élevé mais nécessite des calculs coûteux en temps d'exécution et rend l'interaction entre le fluide et des solides dynamiques difficile. À l'inverse, le modèle eulérien permet d'obtenir des performances interactives ainsi que des interactions robustes avec des objets dynamiques avec comme contrepartie un niveau de réalisme plus faible. Cette approche est la plus intéressante dans notre cas car elle inclut les interactions avec les solides de manière native.

La méthode de simulation lagrangienne principale est le *Smoothed Particles Hydrodynamics* (SPH). Il existe plusieurs variantes de l'algorithme SPH, notamment le PCISPH qui assure l'incompressibilité du fluide grâce à un processus prédictif-correctif et le DFSPH qui améliore la stabilité du fluide en assurant une divergence nulle.

Pour obtenir des résultats temps réel, il est nécessaire d'utiliser une implémentation GPU, qui utilise à son avantage l'aspect parallèle des algorithmes SPH. Bien qu'il existe de nombreuses implémentations GPU disponibles, aucune n'utilise l'algorithme DFSPH. Cependant, Jan


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

Bender a mis en ligne une plateforme (SPlisHSPlasH [Ben]) proposant non seulement une implémentation parallèle CPU de leur algorithme et de la plupart des algorithmes récents. Notre objectif est d'obtenir des résultats interactifs tout en assurant la stabilité de la simulation. Dans la section 5 nous proposons une implémentation sur carte graphique de l'algorithme DFSPH présent dans la plateforme SPlisHSPlasH.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

# 4 Contrôle à basse fréquence dans l'espace des articulations

Dans cette section, nous présentons les travaux relatifs au contrôle de notre personnage bipède. Comme nous l'avons présenté lors de notre étude des travaux existants, notre objectif est d'obtenir un contrôleur temps réel utilisant peu de ressources de calcul de manière à pouvoir obtenir des performances interactives lorsque la simulation contient également un liquide. Pour cette raison, nous avons choisi de concevoir un contrôleur dans l'espace des articulations similaire au modèle SIMBICON[YLV07]. Pour minimiser les ressources de calcul utilisées par la simulation du fluide, il est nécessaire, pour les méthodes de simulation actuelles, d'utiliser des pas de simulation d'environ 5ms, soit une fréquence de 200Hz. Or, les contrôleurs de type SIMBICON proposés dans les travaux existants utilisent des fréquences de simulation supérieures à 1000Hz. L'utilisation de fréquences de simulation plus faibles provoque l'apparition d'instabilités au niveau des collisions entre les différents objets présents dans la simulation physique. L'objectif principal de nos travaux est de proposer un contrôleur capable de supporter l'utilisation de fréquences de simulation variées, entre 200Hz et 1000hz. Également, nous améliorons les méthodes de contrôle de la vitesse du personnage ainsi que de sa direction pour les rendre plus efficaces et robustes.

## 4.1 Vue d'ensemble

La figure 4.1 propose une vue d'ensemble de notre contrôleur. Il étend celui présenté dans [CPB15] :

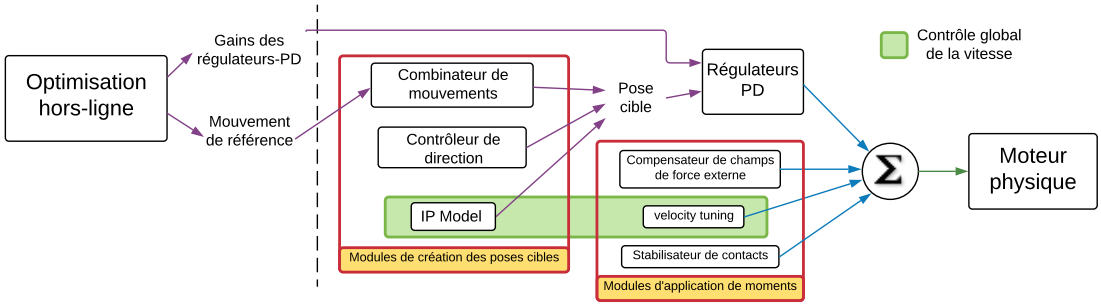



**Figure 4.1.:** Vue d'ensemble des différents composants de notre contrôleur. On distingue trois parties principales : l'optimisation hors-ligne des paramètres d'entrée (à gauche), les composants affectant la pose cible transmise aux régulateurs PD et les composants appliquant des moments additionnels. En vert, les modules contrôlés par le système du contrôle global de la vitesse.

— **Optimisation hors-ligne :** Le but de notre optimisation hors-ligne est de produire les paramètres d'entrée du contrôleur. Nous effectuons deux optimisations distinctes. La première permet d'obtenir des mouvements de référence spécifiques à des conditions données. Dans nos expériences les paramètres pris en compte sont le niveau de liquide ou la vitesse de déplacement du personnage. Cette optimisation utilise la fonction



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

d'évaluation présentée dans [CPB15]. La seconde permet de déterminer les valeurs de gains pour les régulateurs PD optimisées pour des fréquences de simulation. Cette optimisation permet l'utilisation de notre contrôleur à des fréquences de simulation plus faibles que celles utilisées par les contrôleurs similaires de la littérature. Les détails du processus d'optimisation sont présentés dans la section 4.3.2.

— **Génération de la pose cible :** Comme nous l'avons vu lors de l'étude des travaux existants, utiliser uniquement le suivi d'un mouvement de référence ne permet pas de créer un contrôleur stable. Notre contrôleur utilise un pendule inversé (IPM) permettant d'améliorer la robustesse aux perturbations. Nous utilisons également le système de combinaison de mouvements de référence présenté dans [CPB15]. Ce système permet de limiter le nombre de mouvements de référence nécessaires pour adapter le mouvement aux variations du niveau de liquide. À ces deux systèmes nous ajoutons un composant permettant de gérer de grandes variations rapides de la direction désirée du mouvement. Ce composant distribue le changement d'orientation sur une durée de temps pour permettre d'appliquer des variations de direction plus importantes que celles possibles avec le système originel (section 4.2.2).

— **Calcul direct de moments :** Dans cette catégorie se retrouvent les composants appliquant directement des moments sur les articulations du personnage. Nous utilisons une version modifiée du système de compensation de champs de force, compensant la poussée d'Archimède du liquide en plus de la gravité, que nous avons proposé dans [CPB15] (section 2.4.5). Également, nous améliorons le système de gestion de la vitesse du personnage pour obtenir une convergence plus rapide et plus stable vers la vitesse désirée. Notre intérêt pour le contrôle précis de la vitesse vient du fait que cela améliore la robustesse du contrôleur aux perturbations. Ces modifications sont présentées dans la section 4.2.1. Le dernier composant permet de diminuer les instabilités physiques qui seront introduites par l'utilisation de fréquences de simulation faibles. Nous proposons d'utiliser une optimisation en ligne permettant d'évaluer les instabilités du prochain pas de temps et d'ajouter des moments supplémentaires pour éviter leurs apparitions (section 4.3.1).

Ce chapitre est organisé de la façon suivante. Nous présentons d'abord dans la section 4.2 les modifications apportées aux systèmes permettant le contrôle de la vitesse et de la direction du mouvement. Cette section présente également le système de compensation des champs de forces externes. Nous proposons ensuite, dans la section 4.3, une présentation détaillée des différentes modifications nécessaires pour permettre au contrôleur de produire des mouvements stables lors de l'utilisation de fréquences de simulation faibles. La section 4.4 présente nos résultats expérimentaux. Nous terminons ce chapitre par un court récapitulatif des travaux réalisés (section 4.5).

## 4.2 Contrôle de la vitesse et de la direction du mouvement

### 4.2.1 Contrôle de la vitesse

Comme nous l'avons vu précédemment, un contrôle de la vitesse du personnage a été intégré au SIMBICON par Coros et al. [CBV10] via le GENBICON. Ce contrôleur utilise la combinaison d'une version modifiée de l'IPM associée à l'application d'une force virtuelle pour modifier la vitesse de déplacement du personnage. Un défaut de leur approche est dû à l'utilisation de valeurs de paramétrage statiques ce qui empêche leur contrôleur d'obtenir la vitesse


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

désirée si le personnage ne se trouve pas dans des conditions proches de celles prévues par le mouvement de référence. Dans notre contrôleur précédent [CPB15], nous avions résolu cette limitation en introduisant des processus d'apprentissage qui permettent à ces deux composants de s'adapter aux changements de la simulation. Par la suite nous avons modifié l'IPM en ajoutant un décalage de la position cible du pied modifié, à la fin de chaque pas, en fonction de la vitesse du personnage. Le composant appliquant la force virtuelle horizontale est amélioré par l'apprentissage progressif d'une courbe représentant les variations de vitesse à l'intérieur d'un pas et l'application d'un décalage global sur cette courbe pour contrôler la vitesse moyenne du personnage. Bien que permettant d'atteindre précisément la vitesse désirée, ce contrôleur présente plusieurs défauts. Le premier est l'apparition d'oscillations importantes autour de la vitesse cible avant que celle-ci ne soit atteinte. Ce problème est dû au fait qu'aucun des deux composants ne prenne en compte la possibilité que l'autre ai eu un effet qui ait évolué entre deux pas du personnage. Si la vitesse désirée du personnage est augmentée, l'IPM et le *velocity tuning* vont tous les deux tenter d'accélérer le personnage, ce qui va mener au dépassement de la vitesse cible. Ceci augmente le nombre de pas nécessaires à la stabilisation du système et mène à un échec du contrôle lorsque des modifications importantes de la vitesse cible sont demandées au contrôleur. La deuxième limitation de ce système est observable par l'apparition de petites variations de la courbe d'apprentissage et de son décalage lorsque la vitesse cible est atteinte. Ceci est dû en partie à notre supposition que le mouvement est supposé parfaitement symétrique entre deux pas consécutifs. Cependant, ces oscillations indiquent un problème plus important dans la formule utilisée pour calculer l'évolution des points de la courbe d'apprentissage à la fin de chaque pas. Cette évolution est calculé par la formule suivante :

$$V^{t+1}_{cible\_i} = V^{t}_{cible\_i} + \Delta_V$$

$$\Delta_V = \begin{cases} SIGN(Err_i) * \Delta_{Err\_avg}, & \text{if } abs(Err_i) > \Delta_{Err\_avg} \\ SIGN(Err_i) * variation\_limit, & \text{if } abs(Err_i) > variation\_limit \\ 0.25 * Err_i, & \text{sinon} \end{cases} \tag{4.1}$$

Avec $Err_i$ l'erreur entre la vitesse observée et la vitesse de la courbe d'apprentissage au point i telle que $Err_i = (V^t_{observee\_i} - V^t_{cible\_i})$, $variation\_limit$ la variation maximale autorisée pour un pas de simulation et $Err_{avg}$ la moyenne des valeurs absolues des erreurs des différents points de la courbe. Les conditions sur la valeur de $\Delta_V$ sont appliquées dans l'ordre dans lequel elles sont écrites dans le système. Comme on peut le remarquer, si les vitesses observées sont en moyenne plus hautes que celles contenues dans la courbe d'apprentissage, cela va mener à une augmentation de la moyenne de cette dernière. Cela veut dire que ce système ne sépare pas l'apprentissage des variations de vitesse à l'intérieur d'un pas et le contrôle de la vitesse moyenne. Ceci provoque des interférences similaires à celles observées entre l'IPM et le *velocity tuning*.

Pour résoudre ces deux limitations nous proposons de modifier le système de *velocity tuning* et d'ajouter un contrôle global de la vitesse organisant les interactions entre ce système et le décalage appris par l'IPM.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

### velocity tuning

La figure 4.2 présente le cycle de fonctionnement global du système de *velocity tuning*. La détection des perturbations importantes est faite de manière similaire à notre contrôleur précédent [CPB15]. Si la moyenne des erreurs observées entre les vitesses de la courbe d'apprentissage et les vitesses observées, après le décalage dont nous discutons à la fin de cette section, est supérieure à un seuil, alors nous considérons que le personnage a été perturbé. Dans ce cas, nous ne modifions pas la courbe d'apprentissage ainsi que le décalage global.

Il est important de commencer par modifier le système de *velocity tuning* pour découpler l'apprentissage des variations internes à un pas et le contrôle de la vitesse moyenne pour stabiliser le processus d'apprentissage. Pour rappel, le but de la courbe d'apprentissage est de s'assurer que la force virtuelle reste cohérente au cours d'un pas. En pratique, cela revient à chercher à obtenir une force virtuelle constante si le personnage n'est pas perturbé. Le contrôle de la vitesse moyenne étant effectué par l'utilisation d'un décalage global sur cette courbe, il est nécessaire de minimiser l'impact des modifications de la courbe d'apprentissage sur la vitesse globale du personnage. Pour ce faire, il est envisageable de maintenir une valeur moyenne de la courbe constante entre chaque phase d'apprentissage. Une solution est de translater la courbe de vitesse observée de manière à ce que sa moyenne soit égale à celle de la courbe d'apprentissage.

$$V_{obs\_recal\_i} = V_{obs\_i} + (V_{obs\_avg} - V_{apprentissage\_avg})$$

Avec $V_{obs\_recal\_i}$ les vitesses après translation, $V_{obs\_i}$ les vitesses observées dans la au cours du pas du personnage, $V_{obs\_avg}$ la moyenne des vitesses observées et $V_{apprentissage\_avg}$ la moyenne de la courbe d'apprentissage. Cependant, le nombre de pas de temps de simulation n'est pas forcément égal entre chaque pas du personnage. Cette irrégularité des pas pourrait être amplifiée par les interactions avec l'environnement. La solution choisie pour nos travaux consiste à translater la courbe de vitesse observée de façon à ce que sa médiane soit la même que celle de la courbe d'apprentissage.

$$V_{obs\_recal\_i} = V_{obs\_i} + (V_{obs\_median} - V_{apprentissage\_median})$$

Avec $V_{obs\_median}$ et $V_{apprentissage\_median}$ les valeurs médianes des vitesses observées et de la courbe d'apprentissage En plus de résoudre le problème de durée des pas du personnage, la médiane rend le recalage des deux courbes résistant à des perturbations ponctuelles du personnage. La stabilité apportée par cette modification nous a permis de retirer les seuils de variation maximale des points de la courbe d'apprentissage présent dans notre ancien système d'apprentissage. La variation appliquée correspond à $50\%$ de l'erreur entre les deux courbes au niveau de chaque point. Ceci permet d'accélérer grandement la vitesse de convergence vers une courbe stable ainsi que la convergence de la vitesse du personnage vers la vitesse cible. La formule exprimant la variation de chaque point est la suivante :

$$V^{t+1}_{apprentissage\_i} = V^{t}_{apprentissage\_i} + 0.5 * (V^{t}_{obs\_recal\_i} - V^{t}_{apprentissage\_i})$$

Avec $t$ l'identifiant du pas que le personnage vient de terminer. En plus de cette correction nous avons également ajouté deux autres modifications au système de *velocity tuning*. La première est l'ajout d'un seuil retirant la phase d'apprentissage de la courbe d'évolution dans le cas où la différence entre cette dernière et la courbe de vitesse observée est très faible. Ce système est illustré dans la partie *Limiteur d'oscillations* de la figure 4.2. Le but est de


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

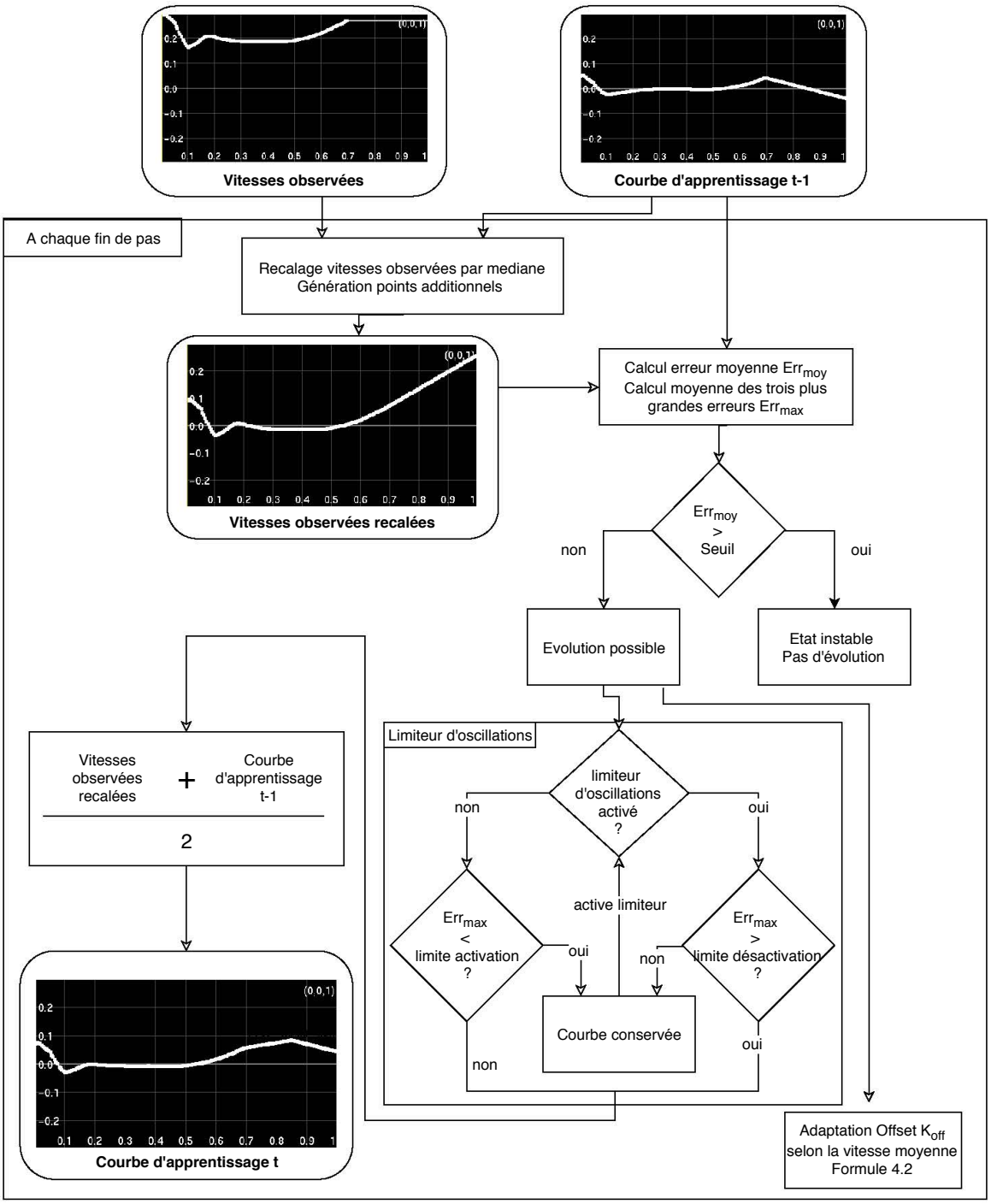



**Figure 4.2.:** Diagramme de fonctionnement du système de *velocity tuning*



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

supprimer les petites oscillations dues aux légères différences de durée de vitesse entre deux pas consécutifs lorsque l'on est dans un état stable. La valeur comparée au seuil est la moyenne des trois plus grandes erreurs entre les deux courbes après le décalage des vitesses observées. En pratique, nous utilisons deux seuils : le premier est utilisé pour savoir si ce système doit être activé à partir de ce pas et le second, plus élevé, est utilisé pour décider si l'on doit reprendre l'apprentissage si le système était actif au cours du pas précédent. La deuxième modification est apportée à la formule utilisée pour l'apprentissage du décalage global. La formule a été modifiée pour intégrer à chaque pas une partie de l'évolution apportée lors du pas précédent. Le but de cette modification est d'empêcher l'apparition d'un mouvement où le décalage oscillerait entre deux valeurs autour de la valeur cible. Ce genre de phénomène est une des causes possibles de l'apparition d'une variation cyclique du temps pris pour effectuer chaque pas suivant la jambe d'appui. La formule utilisée est :

$$\Delta K_{off}^{t} = K1 * \Delta K_{off}^{t-1} + K2 * (V_D - V_{avg}) \tag{4.2}$$

Avec $V_D$ la vitesse désirée et $V_{avg}$ la vitesse moyenne observée. Les coefficients $K1$ et $K2$ utilisés sont respectivement $0.3$ et $0.7$ pour l'axe sagittal et les deux sont fixés à $0.5$ sur l'axe coronal. L'impact de la valeur du pas de temps précédent est plus élevé pour l'axe coronal car cet axe utilise des courbes d'apprentissage séparées suivant le pied qui est en phase d'appui pour permettre l'apprentissage de mouvements non symétriques sur l'axe coronal. Par conséquent, le problème de l'apprentissage de deux décalages différents est plus fréquent sur l'axe coronal. La dernière modification porte sur les points de la fin de la courbe d'apprentissage. Comme évoqué précédemment, ces points peuvent ne pas avoir de vitesse observée associée si un pas du personnage est plus court que la durée attendue. Notre courbe d'apprentissage dispose de 100 points d'échantillonnage répartis de manière homogène sur la durée attendue du pas. Avec ce niveau de précision nous avons observé un maximum de trois points de la courbe pour lesquels nous ne disposons pas toujours d'une vitesse observée pour un mouvement non perturbé. Pour leur attribuer une vitesse observée, nous utilisons une extrapolation linéaire des dernières vitesses observées pour déterminer les valeurs additionnelles. Pour nos expériences, les coefficients de la fonction d'extrapolation linéaire sont déterminés avec une extrapolation des moindres carrés des 4 vitesses observées qui précédent celle du dernier pas de simulation. La raison pour laquelle nous ne considérons pas la dernière vitesse observée est qu'elle est observée pour un pas de temps ou un nouveau pied est rentré en contact avec le sol et par conséquent elle correspond à un cas particulier.

### Contrôle global de la vitesse

La figure 4.3 présente le schéma général du processus de contrôle global de la vitesse. La partie supérieure droite (condition 1) de la figure 4.3 illustre le fait que notre système de contrôle de la vitesse n'est activé que si le personnage n'a pas été perturbé par un élément de la simulation.

Comme nous avons vu précédemment, le système de *velocity tuning* permet un contrôle précis de la vitesse autour de celle obtenue naturellement avec le mouvement de référence. Pour pouvoir obtenir des vitesses éloignées de la vitesse naturelle, il est nécessaire d'utiliser un système supplémentaire comme celui d'altération d'IPM utilisé par notre contrôleur précédent [CPB15]. Cependant, si l'apprentissage des deux systèmes est actif simultanément, cela peut entraîner des oscillations autour de la vitesse désirée. Pour résoudre ce problème, il est nécessaire de réguler l'utilisation des phases d'apprentissage pour minimiser le phénomène.


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

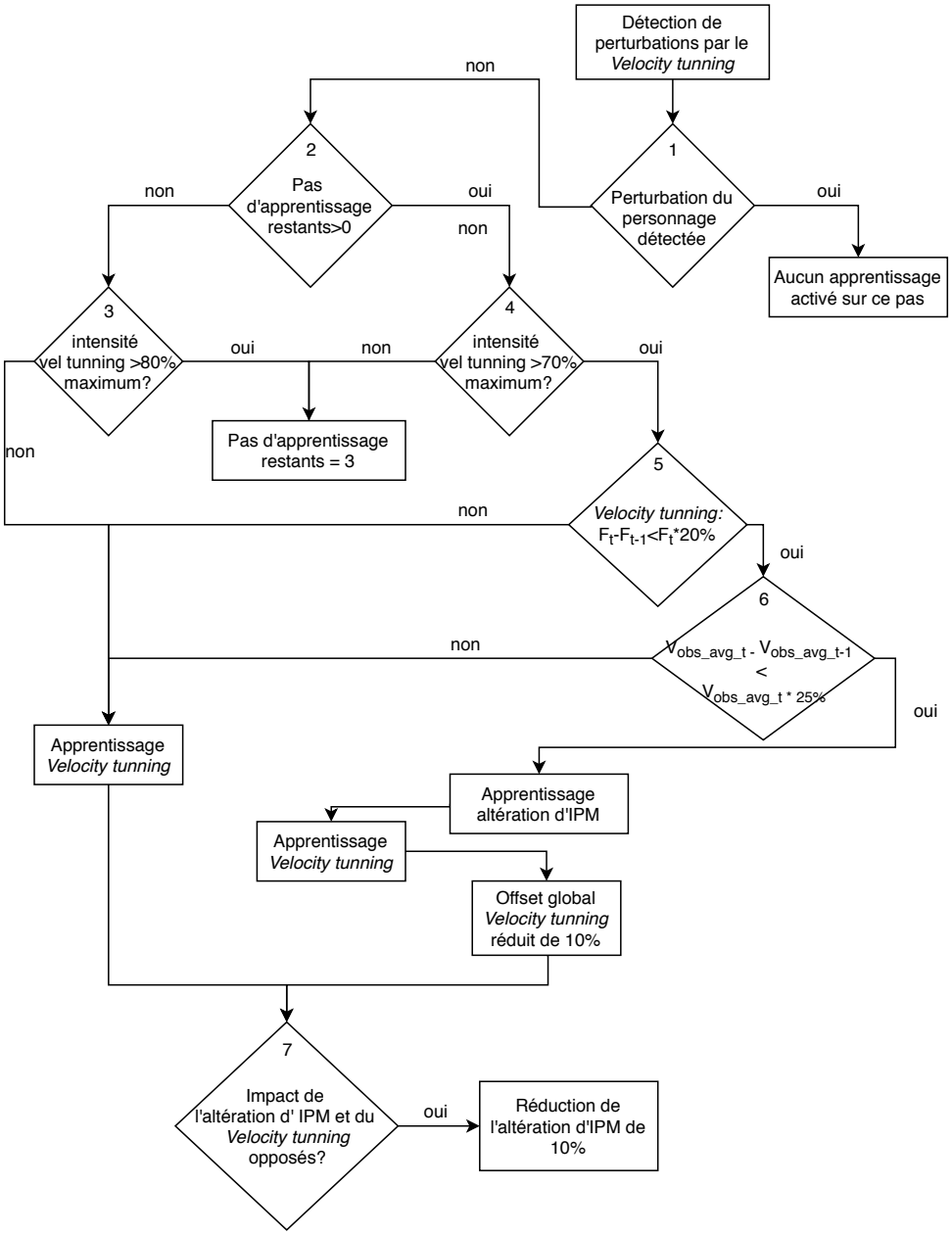



**Figure 4.3.:** Diagramme présentant l'organisation des différents composants permettant de contrôler la vitesse du personnage. Ce processus est exécuté à la fin de chacun des pas du personnage. La détection de perturbation est effectuée en examinant la variation moyenne de la courbe d'apprentissage du *velocity tuning* comme présenté dans le diagramme 4.2. L'étape d'apprentissage du *velocity tuning* à la fin du processus est identique si l'apprentissage de l'altération d'IPM est actif ou non.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

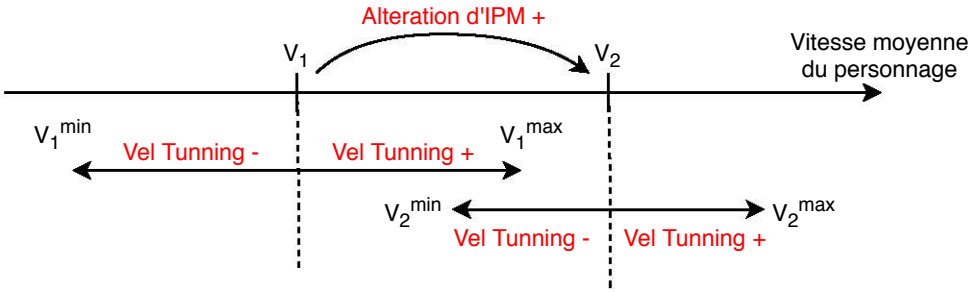



**Figure 4.4.:** Illustration de valeurs de vitesse atteignables par l'utilisation de chacun des systèmes d'apprentissage. Pour simplifier l'exemple, seule une valeur positive d'altération d'IPM est représentée.

Vu que la modification de la valeur de l'altération d'IPM déforme le mouvement pour modifier la vitesse du personnage, il est intéressant d'essayer d'obtenir la vitesse cible sans avoir recours à ce système pour rester aussi fidèle que possible au mouvement de référence. Nous avons donc choisi d'utiliser au maximum l'apprentissage du *velocity tuning* et d'activer celui de l'altération d'IPM seulement si le premier ne permet pas d'atteindre à la vitesse désirée.

La figure 4.4 représente l'impact de chacun des systèmes d'apprentissage. Si l'on considère que le mouvement de référence possède une vitesse naturelle $v_1$ alors le *velocity tuning* permet d'obtenir des vitesses contenues dans l'intervalle $[v_1^{min}, v_1^{max}]$. Si la vitesse désirée $v_d$ se trouve à l'extérieur de cet intervalle, il est nécessaire de déformer le mouvement en appliquant une valeur d'altération d'IPM différente afin d'atteindre une vitesse $v_2$ pour laquelle la vitesse désirée se trouve à l'intérieur de l'intervalle permit par le *velocity tuning* $v_d \in [v_2^{min}, v_2^{max}]$. Pour éviter d'avoir à modifier la valeur de l'altération d'IPM lors d'un futur changement de vitesse désirée, il est intéressant de faire en sorte que $v_2 = v_d$. Ceci permettrait d'obtenir la vitesse désirée sans application de force virtuelle par le *velocity tuning*, permettant ainsi une plus grande marge de manœuvre par le futur. Cependant, obtenir la valeur d'altération d'IPM permettant d'obtenir ce résultat est problématique. Pour illustrer le problème, nous allons supposer que $v_d > v_1^{max}$. Vu que l'apprentissage du *velocity tuning* est prioritaire, le contrôleur va en premier atteindre $v_1^{max}$ en appliquant la force maximale possible. L'apprentissage d'IPM est ensuite activé permettant ainsi d'atteindre la vitesse $v_d$. Cependant, vue que la vitesse observée n'a jamais dépassé la vitesse désirée, cela veut dire que la force appliquée par le *velocity tuning* est toujours la force maximale, c.à.d que la vitesse désirée correspond à la vitesse maximale possible à l'intérieur de l'intervale permi par le *velocity tuning* $v_d = v_2^{max}$. Cela veut dire que même une légère augmentation de la vitesse cible déclencherait à nouveau l'apprentissage de l'altération d'IPM. Réussir à faire apprendre à l'altération d'IPM un décalage permettant $v_d = v_2$ est complexe. En effet, pour faire évoluer la valeur de l'altération, nous utilisons l'erreur sur la vitesse moyenne du personnage. Une fois que le système atteint l'état où $v_d = v_2^{max}$, le contrôleur a atteint la vitesse cible et il n'y a plus d'erreur entre la vitesse cible et la vitesse observée permettant d'adapter davantage la valeur de l'altération d'IPM. Le problème dont nous venons de discuter vient principalement du fait qu'au cours de l'apprentissage de l'altération d'IPM, la force virtuelle appliquée par le *velocity tuning* limite l'apprentissage de l'altération d'IPM. La solution naïve serait de ne pas appliquer la force virtuelle du *velocity tuning* au cours de la phase d'apprentissage de l'altération d'IPM. Cependant, cela provoquerait des changements brusques de vitesse menant à une réduction importante de la capacité du personnage à conserver son équilibre à l'instant de l'activation de l'apprentissage de l'altération d'IPM. Pour obtenir un système stable il est nécessaire que la force virtuelle du *velocity tuning* diminue progressivement tant que l'apprentissage de


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

l'altération d'IPM est activé. Pour obtenir ce résultat nous pouvons proposer deux solutions. La première serait de faire en sorte que l'altération d'IPM vise une vitesse plus élevée que la vitesse cible. De cette manière l'apprentissage du *velocity tuning* diminuerait naturellement la valeur de la force virtuelle pour corriger la vitesse permettant ainsi d'obtenir le résultat désiré. La seconde solution est de forcer une réduction de la force virtuelle appliquée par le *velocity tuning*. Ceci peut être obtenu en réduisant la valeur du décalage global du *velocity tuning*. La première solution maintient donc une vitesse supérieure à la vitesse cible durant la phase d'apprentissage alors que la seconde solution maintient une vitesse inférieure. La seconde méthode nous a apparu plus robuste car l'intervalle des vitesses observées au cours de la phase d'apprentissage sera moins grand. En effet, la vitesse intermédiaire nécessaire à son fonctionnement est plus proche de la vitesse naturelle du mouvement de référence augmentant ainsi les chances qu'elle soit atteignable par notre système. En pratique nous utilisons une réduction de $10\%$ de la valeur du décalage global du *velocity tuning* à la fin de chaque pas au cours de la phase d'apprentissage de l'IPM.

Enfin, la question qui reste posée est à quel moment doit s'activer l'apprentissage de l'altération d'IPM et à quel moment doit-on l'arêter. Pour décider du moment d'activation on définit une intensité moyenne maximale pour la force virtuelle applicable par le système de *velocity tuning*. Si l'intensité de la force virtuelle moyenne au cours d'un pas est supérieure à $80\%$ de cette intensité maximale, alors l'apprentissage de l'altération d'IPM est activé. Nous avons choisi de le désactiver après un nombre de pas. Cette solution a pour inconvénient de limiter la capacité à atteindre la valeur optimale d'altération d'IPM. Cependant, étant donné que l'état du contrôleur avec les deux systèmes d'apprentissage de la vitesse actifs n'est pas un état stable, il est préférable de quitter cet état rapidement, quitte à ne pas atteindre la valeur optimale. Nos expériences ont montré qu'une durée d'activation de $3$ pas est suffisante. Tant que la force appliquée par le *velocity tuning* reste proche de l'intensité maximale le compteur de pas est maintenu à 3. Dans nos expériences, vu que nous réduisons de $10\%$ la valeur du décalage à chaque fin de pas, tant que l'intensité moyenne reste supérieure à $70\%$ le compteur de pas ne diminue pas. Ce processus d'activation et de désactivation de l'altération d'IPM est illustré dans la partie centrale (conditions 2,3 et 4) de la figure 4.3.

L'apprentissage de l'altération d'IPM peut être désactivé ponctuellement sur un pas du personnage selon l'état observé dans la simulation. Nous avons ajouté à notre système deux règles pouvant avoir cet effet. Le but de ces règles est de laisser le temps au système de se stabiliser lorsque la variation de la valeur de l'altération d'IPM entre deux pas provoque une variation très élevée de la vitesse du personnage supérieure à $25\%$. La première désactive l'apprentissage de l'altération d'IPM si une forte variation de la vitesse entre le pas actuel et le pas précédent est détectée. La seconde règle décide de la désactivation de l'apprentissage en regardant l'évolution de la force virtuelle moyenne appliquée par le *velocity tuning* par rapport au pas précédent. Si celle-ci a varié de plus de $25\%$ de l'amplitude maximale autorisée alors il n'y a pas de modification de la valeur de l'altération d'IPM. Ces deux règles sont illustrées par les deux conditions (conditions 5 et 6) dans la partie inférieure droite de la figure 4.3.

Enfin, nous avons une dernière règle traitant le cas où l'altération d'IPM et la force virtuelle du *velocity tuning* ont des effets opposés sur la vitesse du personnage. Ce cas est facile à détecter en comparant simplement le signe de l'altération d'IPM à celui de la force virtuelle moyenne. Dans ce cas, nous diminuons de $10\%$ la valeur de l'altération d'IPM. Cette règle n'a pas d'impact sur la vitesse du personnage, elle permet simplement d'éviter d'appliquer des forces virtuelles et altérations du mouvement naturel du personnage superflues. Cette dernière règle (condition 7) est visible en fin de processus sur la figure 4.3.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

### 4.2.2 Contrôle de la direction

Le but du système présenté dans cette section est de rendre possibles des changements de direction brusques et élevés. Pour obtenir des changements de direction importants avec l'un des contrôleurs de type SIMBICON existants il est nécessaire d'utiliser des mouvements de référence spécifiques ou d'utiliser un moteur physique bien plus stable que les moteurs classiquement utilisés. La raison est que l'orientation du déplacement est simplement contrôlée en modifiant l'orientation cible du bassin pour qu'elle corresponde à la direction du mouvement. Cependant, si une variation de direction assez élevée (40 degrés et plus) est demandée au contrôleur, celui-ci va produire des moments trop importants sur la hanche de la jambe de support qui mèneront à la chute du personnage.

Pour rendre possible des changements de direction plus élevés, il est nécessaire de mettre en place une solution qui réduit l'amplitude des moments appliqués ou au minimum l'amplitude des variations des moments au cours du temps. Pour ce faire, notre contrôleur utilise deux stratégies complémentaires. La première est d'effectuer le changement entre l'ancienne direction et la nouvelle sur un intervalle de temps au lieu de l'effectuer sur un seul pas de temps. En pratique, nous considérons une période d'environ $50\%$ du temps moyen nécessaire à un pas du personnage. Ceci permet de répartir le changement de direction sur deux pas. Également, il est possible d'améliorer la stabilité du personnage en ajoutant quelques contraintes sur cette période de transition. En particulier, il est préférable de ne pas modifier la direction cible du personnage durant le début d'un pas. En effet, durant cette période le pied peut ne pas être correctement ancré au sol ce qui diminue les moments maximaux applicables sans mener à une perte d'équilibre. Nous avons choisi de limiter notre contrôleur pour qu'il ne modifie pas la direction demandée au bassin durant les premiers $15\%$ de la durée moyenne d'un pas. La seconde stratégie provient d'une réflexion sur ce qui peut être considéré comme un changement d'orientation brusque. Une réponse immédiate serait de considérer brusque un virage de grande amplitude effectué en un seul pas. Cependant, si l'on observe le mouvement humain, les changements de direction importants sont effectués sur deux pas. Par exemple, pour effectuer un virage à $180°$, il est nécessaire de faire un premier pas dans l'ancienne direction au cours duquel le bassin commence à pivoter, puis un second pas finissant le changement de direction. Notons tout de même que ceci est une observation que nous avons réalisé nous-même et que nous n'avons pas trouvé d'étude scientifique le mesurant précisément. Nous avons intégré une stratégie inspirée de ces observations dans notre contrôleur. Sur un premier pas, notre contrôleur vise une direction intermédiaire $\alpha_{inter}$ permettant d'effectuer une proportion du changement de direction. Puis un second pas permet d'atteindre la direction finale. Ce second pas considère comme direction de départ pour la période de transition la direction effectivement atteinte à la fin du premier pas et non pas la direction intermédiaire désirée. Ceci permet de respecter la règle précédemment établie interdisant les changements d'orientation au cours du début du pas. Il est parfois possible que le personnage n'atteigne pas la direction désirée à la fin du second pas et un troisième pas est alors nécessaire pour atteindre la direction. Cependant, nous avons observé que les changements de direction effectués au cours de ce troisième pas sont faibles même pour des changements de directions proches de $90°$. La figure 4.5 présente un exemple d'évolution de la direction cible et de la direction observée au cours d'un changement de direction important.

Une autre amélioration que nous avons apportée au contrôleur du contrôle permet de contrôler l'orientation du pied de la jambe en phase de vol. Étant donné que les chevilles du personnage ne disposent pas de degré de liberté dans l'axe de la jambe, le seul moyen de contrôler l'orientation du pied est de modifier l'orientation de la hanche. Dans le GENBICON [CBV10] et par extension notre précédent contrôleur [CPB15], l'angle de la hanche dans l'axe permettant


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

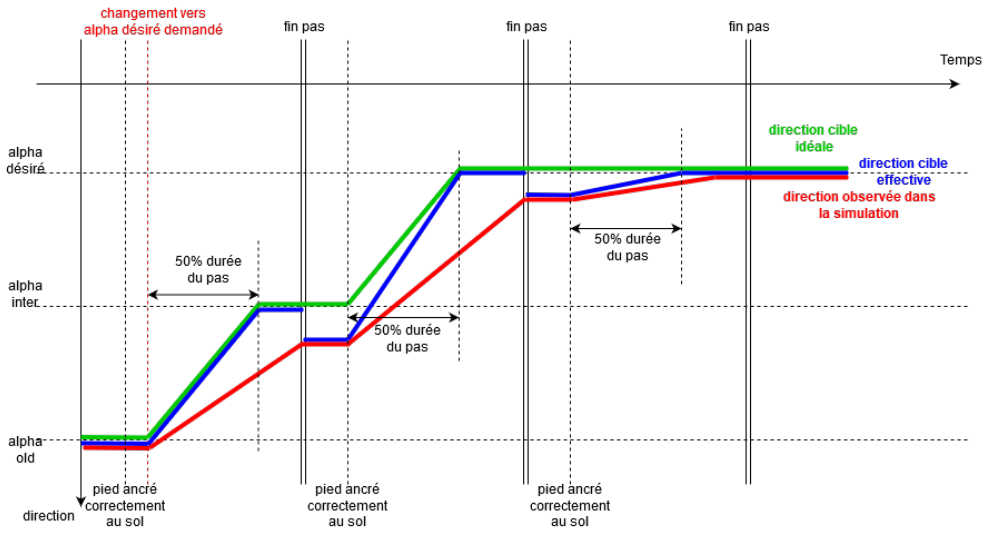



**Figure 4.5.:** Représentation de l'évolution de la direction cible idéale (verte), la direction cible effective (bleue) et la direction observée dans la simulation (rouge) lorsque l'on change la direction désirée de $alpha_{old}$ vers $alpha_{desire}$. Au lieu de passer la direction cible instantanément à $alpha_{desire}$, la direction cible idéale utilise plusieurs pas du personnage où une orientation intermédiaire $alpha_{inter}$ est définie comme cible pour le premier pas. Cette direction cible idéale prend aussi en compte les contraintes proposées (changement de direction sur un maximum de $50\%$ du cycle de marche, interdiction de changer l'orientation pendant les premiers $15\%$ d'un pas). Afin d'éviter l'application de moments trop élevés, la direction cible effective est fixée à l'orientation observée en début de chaque pas.

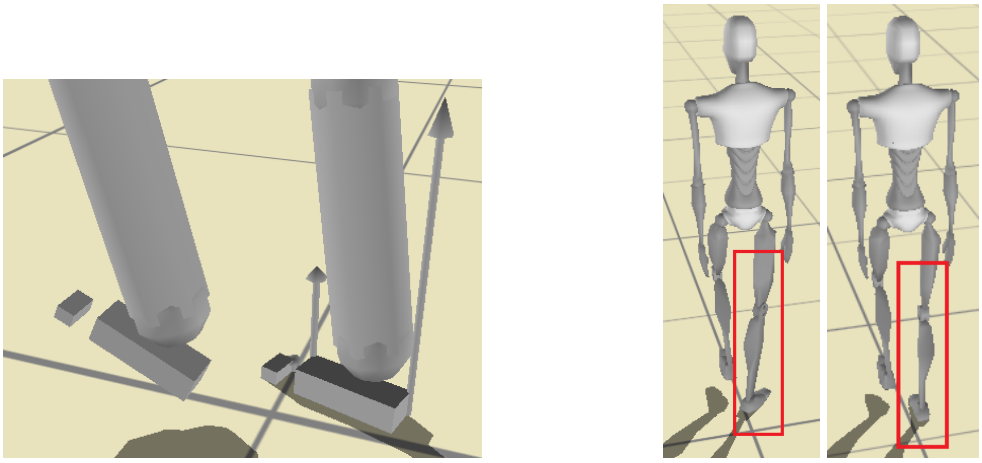
de contrôler l'orientation du pied est simplement fixé à zéro après que la position du pied soit obtenue par l'utilisation de l'IPM. Comme les calculs de l'IPM se basent sur l'orientation observée du bassin, il en résulte un problème. Si une perturbation cause un changement d'orientation du bassin, le personnage va orienter le pied en phase de vol pour suivre cette perturbation et non pas la direction désirée du mouvement. Pour résoudre ce problème, nous calculons l'orientation cible de la hanche de la jambe en phase de vol pour aligner la direction cible du pied avec la direction cible du bassin. On peut noter un fait intéressant, avec cette technique, le pied devrait toujours atteindre l'orientation cible avant le bassin. Ceci est dû en partie au fait qu'il est plus précis d'appliquer un changement d'orientation sur la jambe en phase de vol que sur le bassin, mais également du fait que le bassin est le support de cette jambe.

## 4.3 Contrôle à basse fréquence

L'un des objectifs de cette thèse est la création d'un contrôleur interactif et temps-réel où le personnage est en interaction avec un fluide simulé. Comme nous l'avons souligné dans notre étude des travaux existants, les contrôleurs de type SIMBICON permettent des performances compatibles avec cet objectif. Cependant, la principale méthode utilisée pour diminuer le temps de calcul nécessaire aux simulateurs de fluides est d'utiliser des grands pas de simulation pour utiliser des algorithmes plus performants. En particulier, l'algorithme DFSPH, l'algorithme permettant d'obtenir les meilleurs temps d'exécution lors de l'écriture de ce manuscrit, nécessite le minimum de temps d'exécution pour des pas de temps entre 3ms et 5ms (i.e. entre 200Hz et 333Hz). Cependant, les travaux existant sur les contrôleurs de type SIMBICON sont incapables de produire une simulation stable en dessous d'une fréquence de



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés



(a) (b)

**Figure 4.6.:** (a) Exemple de situation où seule une partie des coins de la primitive de collision (boite) représentant le pied en phase de support est en contact, et des forces sont appliquées, avec le sol. Dans cette simulation, le pied est posé à plat sur le sol, et les quatre coins inférieurs de la boite sont effectivement en intersection avec le sol. Les flèches verticales correspondent à la composante normale des forces de réaction du sol, *Ground Reaction Forces* (GRF). (b) Exemple de situation où des contacts instables ont mené à la rotation de la jambe de support (rectangle rouge). Gauche : état observé ; droite : état correct.

800Hz [GY11 ; Gre16]. La principale raison est l'introduction d'instabilités physiques dues aux imprécisions numériques générées par de grands pas de temps. La solution préconisée dans les travaux les plus récents est d'utiliser un nouvel algorithme de calcul des forces de collisions présent notamment dans le moteur physique MuJoCo. Cependant nous avons choisi de ne pas limiter notre contrôleur à ce moteur physique propriétaire qui n'était pas encore disponible au début de cette thèse. Il existe également une seconde raison à la limite d'une simulation à 800Hz observée dans la littérature : les données d'entrée ne sont pas adaptées. En effet, comme nous le verrons dans cette section, il est nécessaire d'utiliser des valeurs de gains pour les régulateurs PD différentes selon la fréquence de simulation.

Dans cette section nous présentons les modifications apportées à notre contrôleur permettant l'utilisation de fréquences de simulation basses. La section 4.3.1 présente les différentes méthodes expérimentées visant à diminuer l'impact des instabilités physiques introduites par les fréquences de simulation basses (i.e. inférieures à 500Hz). Nous y présentons principalement une nouvelle méthode qui utilise une optimisation en ligne pour ajouter des moments supplémentaires sur la jambe en phase d'appui qui permettent de compenser les instabilités. Nous développons dans la section 4.3.2 une étude des valeurs de gains des régulateurs PD et leur évolution en fonction de la fréquence de simulation. Nous explicitons par la suite l'étape d'optimisation hors ligne permettant de déterminer les valeurs de gains adaptées à chaque fréquence pour un mouvement de référence donné.

### 4.3.1 Gestion des contacts instables

Comme expliqué précédemment, le principal problème lié à l'utilisation de fréquences de simulation basses est l'apparition d'instabilités au niveau des contacts dues à la plus faible fréquence des tests de collision. Bien que l'on n'observe pas d'accumulation de ces instabilités menant à un échec de la simulation, leur présence contribue à détériorer les contacts entre le personnage et le sol. Elles causent des tremblements, en particulier entre le pied et le sol,


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

pouvant entraîner une absence de points de contact entre ces deux éléments au moment où l'on devrait en observer (figure 4.6a). Le problème est que le contrôleur suppose que le pied de support est en permanence en appui avec le sol et que les forces appliquées par le moteur physique empêche tout déplacement du pied. Le manque de contacts continus mène généralement à une rotation de la jambe d'appui due aux moments appliqués sur la hanche qui ont pour but de contrôler l'orientation du bassin (figure 4.6b). Au début de cette thèse le moteur physique MuJoCo n'était pas encore disponible et aucun autre moteur physique ne mettait en place une solution permettant de résoudre ce problème du côté du moteur physique. Nous avons étudié deux solutions, chacune visant à résoudre le problème des contacts par l'ajout de forces externes ou par la modification du modèle physique.

**Ancrage manuel du pied de support** Étant donné que le problème menant à un échec de l'animation est la rotation de la jambe en phase de support ou bien un glissement du pied, la solution la plus simple serait de fixer l'orientation et la position de ce dernier. Pour cela, il aurait été possible de fixer le pied en le transformant en objet statique dès qu'il arrive en contact avec le sol. Une telle solution ne correspond pas à un cas physiquement réaliste. En théorie, toute rotation de la jambe de support ou glissement du pied devrait être empêchée par la génération des forces de friction par le moteur physique. Leur apparition nous montre que le moteur physique n'est pas capable de générer les forces de friction nécessaires du fait des tremblements des points de contact. Notre première approche consiste donc à calculer des forces de friction pour empêcher le pied de bouger. Pour ce faire, nous enregistrons la position d'apparition des points de contact quand chacun des quatre sommets de la primitive de collision touche le sol pour la première fois au cours d'un pas. A chaque fois qu'un nouveau point de contact apparaît nous adaptons également les positions de ceux déjà enregistrés pour qu'ils correspondent bien à la surface inférieure du pied. L'adaptation des points existants est nécessaire car il est possible d'observer des légers déplacements entre le pied et le sol. Pour calculer les forces à appliquer sur chacun des sommets, nous utilisons des régulateurs PD dans le plan horizontal. Ce système possède le principal avantage d'être extrêmement léger en calculs.

Lors de nos expérimentations, notre ancrage manuel nous a permis d'améliorer la robustesse du contrôleur en particulier lorsque l'on soumettait le personnage à des perturbations externes ou bien que l'on demandait des changements de direction. Cependant, il présente plusieurs problèmes physiques et pratiques. D'un point de vue physique, cette approche revient à créer un matériau avec un coefficient de friction infini en dessous du pied ce qui n'est pas réaliste. Ce manque de réalisme reste cependant sur le plan théorique car normalement le moteur physique devrait lui-même générer des forces de friction générant le même résultat. Le problème principal est que ce système est basé sur l'utilisation de régulateurs PD. De ce fait, il est nécessaire d'en déterminer les gains ce qui est extrêmement complexe. En effet, contrairement aux gains utilisés pour le contrôle global du personnage, ce système requiert des gains extrêmement précis. Des gains trop élevés ne vont pas simplement mener à un aspect robotique mais ils vont mener à de brusques oscillations de la position du pied et donc à des vibrations de la jambe d'appui ce qui mène à un échec du contrôle. À l'inverse, des gains faibles vont mener à un déplacement visible du pied ainsi qu'à une correction lente ce qui ne donne pas un résultat satisfaisant. C'est pour ces raisons que nous avons choisi de ne pas continuer à explorer cette solution.

**Utilisation d'un corps mou** Cette solution s'inspire de la méthode proposée par Jain et Liu [JL11]. Leur solution aux instabilités observées au niveau des contacts est de placer un corps mou en dessous du pied du personnage. Cette solution permet d'obtenir une excellente



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

stabilité des forces de contact générées par le contrôleur. Cependant, leurs travaux se basent sur une fréquence de simulation élevée (2000Hz) et l'applicabilité d'un tel système à des fréquences de simulations faibles n'est pas démontrée. Également, les auteurs utilisent un moteur physique développé pour la réalisation de leurs travaux à la place d'utiliser un moteur physique classique. Pour vérifier l'applicabilité de cette solution nous avons mis en place un système similaire dans le moteur ODE. Le corps mou utilisé est un modèle simple. On trouve deux couches de points sur le pied discrétisant de manière régulière le dessus et le dessous du pied. Les points se trouvant sur la face supérieure du pied sont des points fixes qui servent de point d'ancrage du corps mou. Les points se trouvant sur la surface inférieure du pied sont des points mobiles qui nous permettront de détecter les collisions. Pour simuler l'aspect mou du corps simulé deux types de forces sont appliquées sur les points mobiles.

— **Correction de la position.** Cette force permet de pénaliser l'erreur entre la position actuelle d'un point mobile $x_i$ par rapport à sa position de repos $x_{i\_ini}$. La force appliquée simule l'effet d'un ressort de raideur $k_v$ :

$$F_1 = k_v * (x_i - x_{i\_ini}) \tag{4.3}$$

— **Déformation des arêtes.** Cette seconde force vise à réduire les déformations des arêtes reliant chacun des points à ces points adjacents. Cette force se base également sur le modèle d'un ressort de raideur $k_e$ :

$$F_2 = k_e * \sum((x_i - x_j) - (x_{i\_ini} - x_{j\_ini})) \tag{4.4}$$

En reformulant cette équation, on remarque que cette formule correspond simplement à $n$ fois la déformation par rapport à la position au repos moins la somme des déformations des points adjacents, avec $n$ le nombre de points adjacents :

$$F_2 = k_e * (n * (x_i - x_{i\_ini}) - \sum(x_j - x_{j\_ini})) \tag{4.5}$$

Pour intégrer le corps mou dans le moteur ODE, il suffit de se servir des points mobiles pour détecter les collisions au lieu de se servir des sommets de la primitive représentant le pied. De ce fait, en plus des deux forces permettant la simulation du corps mou, on applique sur les points en contact avec le sol la force de réaction du sol calculée par le moteur physique. Cette intégration permet de faire en sorte que le moteur physique s'occupe lui-même de l'application des différentes forces de réaction du sol obtenue au niveau des différents points de contact sur la primitive représentant le pied.

Nous ne sommes cependant pas arrivés à reproduire des résultats similaires à ceux présentés dans la publication de Jain et Liu. Nous avons bien obtenu une meilleure cohérence temporelle des forces de contact entre le pied et le sol. Cependant, il nous a été impossible de déterminer des valeurs de raideur qui permettent d'éviter la présence de fortes intersections entre le pied et le sol au début de chaque pas tout en obtenant un résultat stable sur la totalité de la phase de support. Le problème est que les valeurs de $k_v$ et $k_e$ dépendent probablement de la fréquence de simulation et il est donc extrêmement complexe d'arriver à faire fonctionner un système aussi sensible pour un contrôleur qui doit être stable pour de nombreuses fréquences de simulation. Il est possible que cette approche permette d'obtenir les résultats désirés si on utilise un modèle de corps mou plus complexe, par exemple en utilisant d'autres formules que des ressorts pour les forces internes. Il serait également intéressant d'étudier si l'ajout de plusieurs couches de corps mobiles permet de stabiliser la simulation. Cependant, de telles modifications entraîneraient probablement un fort coût en calcul.


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

Plutôt que de continuer à rechercher une méthode permettant de corriger les imperfections du moteur physique nous nous sommes orientés vers une solution où le contrôle du personnage s'adapte aux instabilités. Ce choix permet de rendre notre contrôleur générique à tout moteur physique.

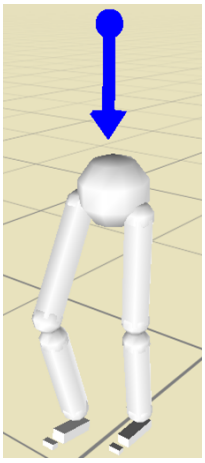
### Stabilisateur de contacts

Notre stabilisateur de contacts ajoute des moments sur diverses articulations du personnage afin d'empêcher l'apparition d'instabilités entre les pieds et le sol. Le but est d'obtenir un état où les forces de réaction du sol sur chacun des sommets du pied sont non négligeables lorsque le pied est supposé posé. Lors de nos expérimentations nous avons remarqué que la rotation de la jambe en phase d'appui n'est observée que dans le cas où un maximum de deux points de contact existent entre le pied d'appui et le sol. L'idée est que si l'on a au moins trois points de contact entre le pied et le sol, les instabilités physiques n'auraient qu'un impact négligeable sur le contrôle du personnage. D'autre part, comme nous l'avons évoqué précédemment, le contrôleur proposé par Greer [Gre16] est capable de produire des animations stables pour des fréquences de simulation très faibles et ceci sans avoir de composant spécifique à la correction des instabilités physiques. Leur contrôleur utilise une méthode basée sur un échantillonnage. Celui-ci tente de réaliser la simulation avec une multitude de mouvements de référence générés en y ajoutant de légères variations et conserve le mouvement permettant d'obtenir le meilleur résultat. Ces légères variations sont les échantillons. Ce type de contrôleur est stable même pour des fréquences de simulation faibles car il rejette simplement tous les échantillons pour lesquels les instabilités physiques ont pour conséquence un échec de la simulation. Plus précisément, leur contrôleur découpe le mouvement en plusieurs parties et il recherche l'échantillon le plus adapté sur chacune de ces parties. Ce type de contrôleur est extrêmement coûteux et son utilisation demande d'évaluer des milliers d'échantillons sur chacune des sous-parties. Pour obtenir des performances temps réel, Greer met en place un processus qui permet de réutiliser les mêmes échantillons entre chaque répétition d'un mouvement cyclique. De ce fait, une fois les échantillons déterminés lors du premier cycle, l'échantillonnage n'est plus à répéter. Si cette solution permet de diminuer les calculs réalisés, elle ne permet pas de produire un contrôleur agile et robuste aux perturbations externes. En effet, les échantillons sont spécifiques aux conditions dans lesquelles se trouvait le personnage au premier cycle. De ce fait, si l'environnement de simulation change, ou si une perturbation assez importante est appliquée, ou si une action différente est demandée (ex : un changement de direction), les échantillons utilisés ne seront plus capables de produire un mouvement stable et le contrôleur perd ses performances temps réel.

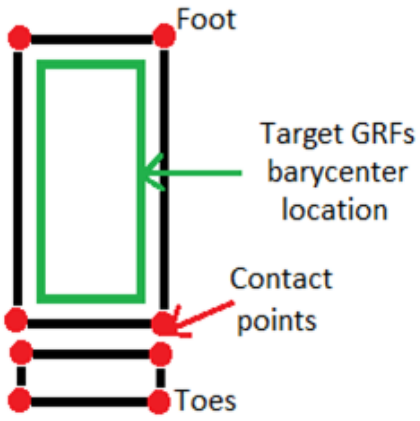
Nous nous sommes inspirés de l'approche basée sur l'échantillonnage pour concevoir un système qui évite les situations où les instabilités physiques provoquent un échec de la simulation. Dans notre approche, nous simplifions de nombreuses parties du processus d'échantillonnage car notre système se concentre sur les contacts avec le sol. En particulier, nous ne maintenons pas nos échantillons sur une fenêtre de temps. Nous calculons un échantillon différent pour chaque pas de temps de simulation. Il y a deux raisons pour ce choix. La première est que les instabilités physiques ne durent pas dans le temps. Entre deux pas de simulation consécutifs il est possible d'observer deux instabilités opposées. Par exemple, le moteur physique ne génère des forces que sur l'un des côtés du pied et le coté affecté par ces forces change à chaque pas de simulation. La seconde est que nous étudions un phénomène dans une zone du personnage où les corps rigides ne se déplacent que très



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés



(a)




(b)

**Figure 4.7.:** (a) Exemple d'application de notre modèle réduit pour lequel toutes les parties du personnage situées au-dessus du bassin ont été simplifiées. La flèche rouge représente le poids des parties simplifiées et est appliquée sur le bassin dans cet exemple. (b) Représentation de la zone objectif pour le barycentre des GRFs dans notre système.

peu entre les pas de temps, de ce fait l'aspect temporel de l'évaluation des échantillons ne nous est pas informatif.

**Les échantillons** Une différence notable avec les travaux récents basés sur de l'échantillonnage [Liu+10 ; LYG15 ; Gre16] est qu'au lieu de définir les échantillons par des variations appliquées au mouvement de référence, nous les définissons comme des moments supplémentaires à appliquer aux articulations. Cela permet de rendre le système indépendant des données d'entrée du contrôleur, en particulier indépendant des valeurs des gains. Soulignons que notre but n'est pas de contrôler le personnage entier mais d'éviter l'apparition des instabilités au niveau des contacts avec le sol. De ce fait nous pouvons limiter nos échantillons aux articulations pour lesquelles l'ajout d'un moment additionnel à de fortes implications sur les contacts entre le pied et le sol. Nos échantillons se limitent donc aux articulations de la jambe en phase d'appui. En effet, l'ajout de moments sur les autres articulations du personnage ne permet pas de produire de variations significatives au niveau des contacts. Parmi les articulations de la jambe d'appui, nous avons exclu l'articulation entre le pied et les orteils car elle ne permet de modifier que la distribution des forces entre le pied et les orteils et non pas leur impact global sur le personnage. Nous avons également exclu la hanche car, lors de la phase d'appui, elle est utilisée pour contrôler le bassin. Nos échantillons sont donc composés de la cheville (2 DOF) et du genou (1 DOF) de la jambe en phase d'appui. Nous obtenons un espace de recherche à trois dimensions au lieu d'un espace à 26 dimensions si l'on considérerait le personnage entier.

**Simplification du modèle du personnage** De manière similaire à la simplification des échantillons, il est possible de simplifier le modèle utilisé pour l'évaluation des échantillons afin de réduire les temps de calcul. Dans notre modèle simplifié, nous retirons certaines parties du personnage et les remplaçons par une force représentant l'influence de leur poids sur les corps rigides restants. Pour minimiser l'introduction d'erreurs, nous conservons la jambe en phase d'appui intacte. Nous avons étudié divers modèles réduits constitués de différentes combinaisons de parties simplifiées du personnage (voir section 4.4.2). La figure 4.7a donne


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

un exemple du modèle réduit où toutes les parties situées au-dessus du bassin ont été simplifiées.

**Évaluation des échantillons** Pour évaluer la qualité des contacts, nous évaluons la répartition des forces de réaction du sol, *Ground Reaction Forces* (GRF), sur le pied en phase d'appui. Nous ne considérons que la composante normale des GRF et ne prenons pas en compte les forces dues à la friction. Ce choix s'appuie sur trois raisons. Premièrement, ce que nous cherchons à évaluer est l'intensité des points de contact entre le pied et le sol. La composante normale des GRFs dépendant directement de la profondeur de l'intersection entre le pied et le sol, elle nous permet parfaitement d'évaluer l'intensité d'un contact. De plus, la composante normale est bien plus stable sur plusieurs pas consécutifs que les forces de friction. Enfin, la formule permettant de calculer les forces de friction utilise la valeur de la composante normale pour fixer son maximum (cône de friction). Si le pied glisse cela veut dire que les forces de frictions appliquées étaient maximales et qu'elles n'ont pas été suffisantes. Dans un tel cas, qui est le cas que nous voulons corriger, l'amplitude de la force de friction est donc liée à celle de la force normale. Par conséquent, évaluer uniquement la force normale est suffisant pour obtenir une bonne représentation de l'état global des contacts.

Notre objectif final est d'assurer qu'une partie non négligeable des GRFs est appliquée sur chacun des côtés du pied et non pas forcément de trouver une solution où les GRFs sont réparties équitablement sur le pied. Une autre formulation de cet objectif serait de se dire que nous cherchons des solutions pour lesquelles le barycentre des GRFs est éloigné des bordures du pied (voir figure 4.7b). Pour nos expériences, notre objectif est de maintenir le barycentre à l'intérieur d'une zone $15\%$ plus petite que le pied.

Nous définissons les rapports $r_{left/right} = F_{left/right}/F_{all}$ et $r_{front/back} = F_{front/back}/F_{all}$ où $F_{right/left}$ et $F_{front/back}$ sont les sommes des GRFs sur chacun des côtés du pavé représentant le pied en phase d'appui et $F_{all}$ est la somme totale des GRFs.

Notre fonction d'évaluation des échantillons est la suivante :

$$f_{eval} = f_{quality} * 10 + f_{distance} \tag{4.6}$$

$$f_{quality} = \begin{cases} 10^{15}/10^{F_{all}/10+1} & F_{all} < 100 \\ (max(l_{cor} - r_{left}, 0))^2 + & \\ \quad (max(l_{cor} - r_{right}, 0))^2 + & sinon \\ \quad (max(l_{sag} - r_{front}, 0))^2 + & \\ \quad (max(l_{sag} - r_{back}, 0))^2 & \end{cases} \tag{4.7}$$

$$\begin{aligned} f_{distance} = max(max(l_{cor}, r_{left\_init}) - r_{left}, 0)^2 + \\ max(max(l_{cor}, r_{right\_init}) - r_{right}, 0)^2 + \\ max(max(l_{sag}, r_{front\_init}) - r_{front}, 0)^2 + \\ max(max(l_{sag}, r_{back\_init}) - r_{back}, 0)^2 \end{aligned} \tag{4.8}$$

Distribution des GRFs : $f_{quality}$ représente la qualité de la distribution des GRFs. $l_{sag}$ et $l_{cor}$ sont des paramètres permettant de contrôler le niveau de restriction imposé sur la distribution, respectivement la limite sagittale et la limite coronale. Des valeurs faibles mènent à une



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

convergence plus rapide et, à l'opposé, des valeurs élevées mènent à une distribution plus équitable et donc probablement plus stable. Dans nos tests, nous avons fixé la valeur de ces deux variables à $0.15$, ce qui correspond à une zone cible $15\%$ plus petite que le pied.

Distance : Le but de la fonction $f_{distance}$ est d'évaluer la distance entre la distribution des GRFs obtenue sans la méthode d'échantillonnage ($r_{cor\_init}$ et $r_{sag\_init}$) et celle obtenue lorsque l'échantillon à évaluer est utilisé. L'objectif qu'exprime cette fonction est de promouvoir les échantillons qui résultent en une distribution aussi proche que possible de la distribution naturelle de la simulation. Nous limitons cependant $r_{left/right\_init}$ et $r_{front/back\_init}$ à respectivement $l_{sag}$ et $l_{cor}$ pour éviter que cette fonction ne mène à préférer un échantillon ne résultant pas à un barycentre des GRFs situé dans la zone cible.

Ces deux fonctions ont le même ordre de grandeur. Nous multiplions le résultat de $f_{quality}$ par un facteur de 10 pour que le système favorise une bonne qualité à une bonne distance du résultat original. Ce choix est dû au fait que la fonction $f_{distance}$ ne sert qu'à différencier entre les échantillons permettant une simulation stable (Eq. 4.6). Nous utilisons l'algorithme *Covariance Matrix Adaptation* (CMA) [Han06] pour générer les échantillons. Dès qu'un échantillon obtient une évaluation telle que $f_{eval} < 1$, il est accepté et l'optimisation se termine. Habituellement, un échantillon est accepté dans les deux premières itérations du CMA. La section 4.4.2 donne les résultats des expérimentations portant sur la gestion des contacts.

### 4.3.2 Étude des gains des régulateurs PD

Habituellement, les contrôleurs basés sur la physique utilisant des régulateurs PD obtiennent les valeurs nécessaires pour les gains à travers une phase d'optimisation hors ligne. Cependant, cette phase d'optimisation n'est pas focalisée uniquement sur la recherche optimale des gains eux-mêmes. En plus de chercher les valeurs des gains, cette phase d'optimisation modifie, par exemple, les poses clefs définissant le mouvement de référence. Généralement, la fonction objectif utilisée cherche à trouver un mouvement de référence permettant de respecter des conditions données ou bien elle a pour but d'améliorer un mouvement de référence obtenu par capture de mouvement, généralement pour assurer une meilleure stabilité, une meilleure efficacité ou le respect des collisions physiques [Gre16]. Bien que ce type d'optimisation permette généralement d'obtenir des valeurs de gains fonctionnelles, elle ne spécifie pas de propriété spécialisée sur les gains. Par exemple, comme nous l'avons discuté dans la section 2.3.1, l'utilisation de gains faibles est généralement favorisée car elle permet d'éviter la génération de mouvement paraissant robotiques. Également, si la plage de gains permettant d'obtenir une simulation stable est très faible, il est possible que l'optimisation échoue (car rien ne l'y guide) alors qu'une solution existe.

Si l'on regarde la formule du régulateur PD (équation 2.1), on peut se rendre compte que si l'on utilise les mêmes valeurs de gains pour des fréquences différentes, le résultat obtenu sera différent. Pour visualiser ce problème on peut s'imaginer le cas suivant. On suppose une simulation pour une fréquence élevée avec des valeurs de gains tels que l'on observe des oscillations harmoniques autour de la cible. Si on diminue la fréquence de la simulation, l'erreur par rapport à la valeur cible est toujours la même et donc le contrôleur applique un moment de la même intensité. Cependant, les pas de temps étant maintenant plus grands, ce moment sera appliqué durant plus de temps ce qui augmentera l'erreur observée au prochain pas de temps. De cet exemple, on peut facilement s'imaginer la raison expliquant pourquoi les travaux existants utilisant des contrôleurs de type SIMBICON observent un échec de


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

la simulation en dessous de 800Hz. Les valeurs de gains utilisés dans leurs tests ne sont probablement plus dans les plages valides pour les fréquences de simulation au-dessus de 800Hz. Cet exemple nous permet de formuler les hypothèses suivantes :

— les plages de gains diminuent avec la fréquence

— les gains diminuent avec la fréquence

Pour pouvoir étudier l'évolution des plages de gains possibles en fonction de la fréquence, nous avons séparé l'optimisation permettant d'obtenir le mouvement de référence et l'optimisation permettant d'obtenir les valeurs de gains. L'optimisation sur les gains est effectuée en second et son but est de déterminer les gains minimaux et maximaux pour chaque articulation pour diverses fréquences.

### Fonction d'évaluation

La fonction d'évaluation utilisée au cours de notre optimisation des gains est la suivante :

$$f_{eval} = f_{gains} + \frac{\sum_{t<k}(f_{tete} + f_{mains})}{k * frequence} + f_{vitesse} + f_{equilibre} \tag{4.9}$$

avec $k$ la durée de l'optimisation en secondes et $frequence$ la fréquence de simulation en Hertz. Cette fonction d'évaluation est composée de trois parties. La première permet d'évaluer les valeurs des gains :

— *évaluation des gains* ($f_{gains}$). Cette fonction est différente si l'optimisation doit rechercher les gains minimaux ou maximaux et permet de guider les gains vers des valeurs faibles ou des valeurs élevées. Pour éviter que l'optimisation ne favorise l'une des articulations, nous utilisons une somme où la valeur de chaque gain est normalisée par sa valeur initiale. Également, pour rendre ce processus indépendant de la valeur initiale, nous mettons à jour la valeur $A_i$ utilisée pour la normalisation toutes les $150$ itérations de l'optimisation.

$$f_{gains} = \begin{cases} f_{gains\_min} = \sum_{t=0}^{k}(\frac{K_p}{A_{p\_i}} + \frac{K_d}{A_{d\_i}}) & \text{recherche du minimum} \\ f_{gains\_max} = C_i - f_{gains\_min} & \text{recherche du maximum} \end{cases} \tag{4.10}$$

$C_i$ est une constante plus grande que la somme et est mise à jour en même temps que les valeurs de normalisation pour que le résultat de la fonction soit continu au cours de l'optimisation.

La seconde partie de la fonction d'évaluation permet d'assurer que le mouvement obtenu reste similaire à celui utilisant les gains initiaux avec la fréquence initiale. Pour évaluer la différence nous comparons la position des effecteurs du système articulé (c.à.d. les mains et la tête) :

— *évaluation des mains* ($f_{mains}$). Pour pouvoir évaluer la qualité de la position des mains, nous utilisons la cinématique inverse pour produire un mouvement de référence avec une position constante pour les mains. Cette position est fixée relativement à la position du bassin ce qui permet de pénaliser toute variation sur les angles des articulations entre le bassin et les mains, la position du torse étant assuré par la seconde fonction.

$$f_{mains} = \sum_{t} \left\| pos_{mains\_t} - pos_{mains\_cible} \right\|^2 \tag{4.11}$$



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

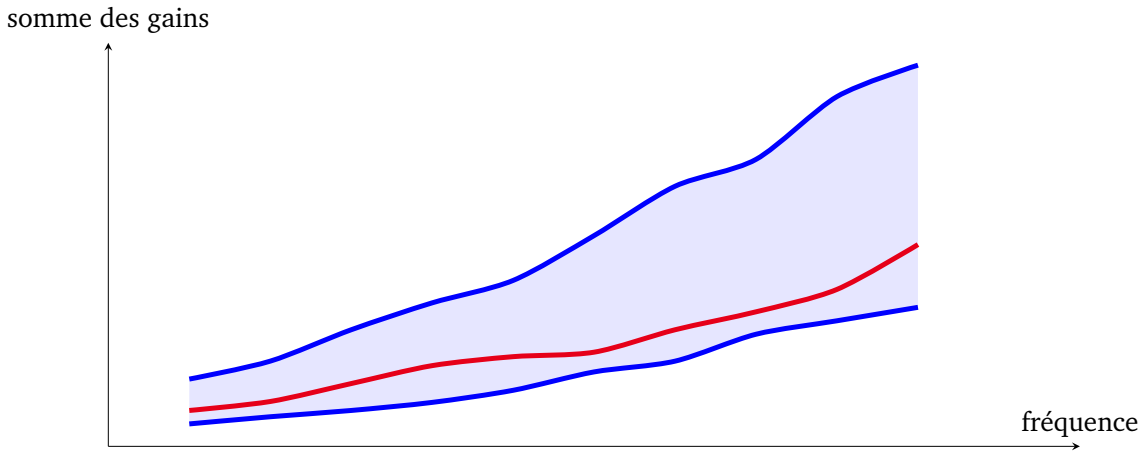



**Figure 4.8.:** Représentation de l'évolution supposée de la somme des gains en fonction de la fréquence. Les courbes bleues représentent l'évolution des ensembles minimum et maximum et la courbe rouge représente l'évolution supposée de l'ensemble de gains optimal. Cette figure est une représentation de nos hypothèses et ne vise pas à correspondre à des valeurs effectivement observées.

— *évaluation de la tête* ($f_{tete}$). Cette fonction permet de pénaliser les solutions pour lesquelles l'élévation de la tête par rapport au sol ne reste pas proche de celle observée dans la simulation initiale. Nous vérifions également que la position de la tête relative au bassin reste similaire. De plus, cette fonction assure que la rotation du cou reste proche de celle originalement observée, car elle influence la position de la tête. Le facteur $K$ est une constante permettant de faire en sorte que chacun des deux facteurs évalués ait le même ordre de grandeur.

$$f_{tete} = \sum_{t} \left\| pos_{tete\_t} - pos_{tete\_reference} \right\|^2 + K * (\theta_{cou\_t} - \theta_{cou\_reference})^2 \qquad (4.12)$$

Il est important de noter que, pour ces deux composantes, nous ne cherchons à obtenir qu'un résultat similaire au résultat initial et non pas exactement le même. En pratique, nous ne pénalisons une solution que si elle sort d'un intervalle centré sur la valeur de référence. Cela permet de laisser une marge de liberté à l'optimisation pour éviter qu'elle reste bloquée dans des minimaux locaux.

La troisième partie contient les fonctions permettant de délimiter l'espace de recherche. Les fonctions que nous utilisons sont similaires à celles proposées dans l'optimisation des poses clefs de notre contrôleur précédent [CPB15]. $f_{vitesse}$ permet de rejeter les solutions pour lesquelles la vitesse moyenne observée sur les trois derniers pas du personnage s'éloigne de plus de $5\%$ de la vitesse désirée. $f_{equilibre}$ vérifie que le personnage est bien dans un état stable au cours de toute la simulation en vérifiant que le système de *velocity tuning* n'a pas détecté de perte d'équilibre.

### Processus d'optimisation

En plus d'utiliser une fonction d'évaluation spécifique aux gains, nous avons mis en place un processus permettant de déterminer les plages de gains valides pour une fréquence donnée. Ce processus utilise comme valeurs initiales un ensemble de gains produisant un résultat stable à une fréquence plus élevée. Avant de présenter l'algorithme, il est nécessaire d'expliquer notre


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

raisonnement. Comme nous l'avons présenté précédemment nous supposons que pour des fréquences de simulation faibles (1) la plage des gains valides est plus restreinte et (2) les valeurs des gains sont probablement plus faibles. À ces deux hypothèses nous pouvons en ajouter une dernière : (3) l'évolution des valeurs optimales des gains avec la fréquence est continue. C'est-à-dire que pour deux fréquences proches les ensembles de gains optimaux seront également proches. La figure 4.8 représente notre vision de l'évolution de la somme des gains valides en fonction de la fréquence. Dans cette vision, on peut voir qu'il est imaginable que deux fréquences éloignées n'aient aucun ensemble de gains commun. C'est pourquoi notre processus d'optimisation utilise des fréquences intermédiaires pour obtenir les ensembles de gains minimaux et maximaux à partir d'un ensemble fonctionnant pour une fréquence proche. En pratique, nous cherchons les ensembles de gains minimaux $S_{min}$ et maximaux $S_{max}$ pour la fréquence de départ, puis nous créons un ensemble intermédiaire $S_i$ proche de l'ensemble minimal $S_i = S_{min} * 0.8 + S_{max} * 0.2$. Cet ensemble intermédiaire est utilisé pour démarrer l'optimisation pour une seconde fréquence plus faible et le processus est répété jusqu'à ce que l'on arrive à la fréquence voulue. Le processus d'optimisation est illustré dans l'algorithme 2.

**Algorithm 2** Procédure d'optimisation hors-ligne des gains

**Require:** un ensemble de gains $S_{ini}$ capable de produire un mouvement stable pour une fréquence $F_{ini}$

**procedure** LOWERFREQUENCY($S_{ini}$,$F_{ini}$)
  ($S_i$,$F_i$) ← ($S_{ini}$,$F_{ini}$)
  **do**
    ($S_{min}$,$S_{max}$,*is_valid*) ← OPTIMIZEFREQUENCY($S_i$,$F_i$)
    $S_i$ ← $S_{min}$*0.8+$S_{max}$*0.2
    $F_i$ ← $F_{new}$ with $F_{new} < F_i$
  **while** *is_valid*

**function** OPTIMIZEFREQUENCY($S_{start}$,F)
  ($S_{temp}$,*is_valid*) ← ($S_{start}$,true)
  **do**
    $S_{min}$ ← $S_{temp}$
    ($S_{temp}$,*is_valid*) ← run_cma($S_{min}$,F,min)
  **while** $\|S_{temp} - S_{min}\| < \eta$ && *is_valid*
  **if** *is_valid* **then**
    **do**
      $S_{max}$ ← $S_{temp}$
      ($S_{temp}$,*is_valid*) ← run_cma($S_{max}$,F,max)
    **while** $\|S_{temp} - S_{max}\| < \eta$ && *is_valid*
  **return** ($S_{min}$,$S_{max}$,*is_valid*)

La fonction *run_cma* exécute une optimisation hors ligne utilisant l'algorithme CMA-ES et la fonction d'évaluation présentée précédemment. Le premier paramètre est l'ensemble de gains à utiliser comme point de départ, le deuxième est la fréquence de simulation et le dernier paramètre indique si l'optimisation recherche l'ensemble de gains minimaux ou maximaux. Elle s'arrête au bout de 150 itérations ou lorsqu'il n'y a plus d'évolution sur les valeurs des gains. La limite de 150 itérations est due à la nécessité de mettre à jour les valeurs de normalisation utilisées pour l'évaluation de la somme des gains. En plus de l'ensemble de gains obtenu, la fonction retourne, à la fin de son exécution, un booléen indiquant si elle a réussi à trouver un ensemble de gains produisant un mouvement stable. Si, pour une fréquence donnée, le processus d'optimisation n'arrive pas à trouver un ensemble de gains



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

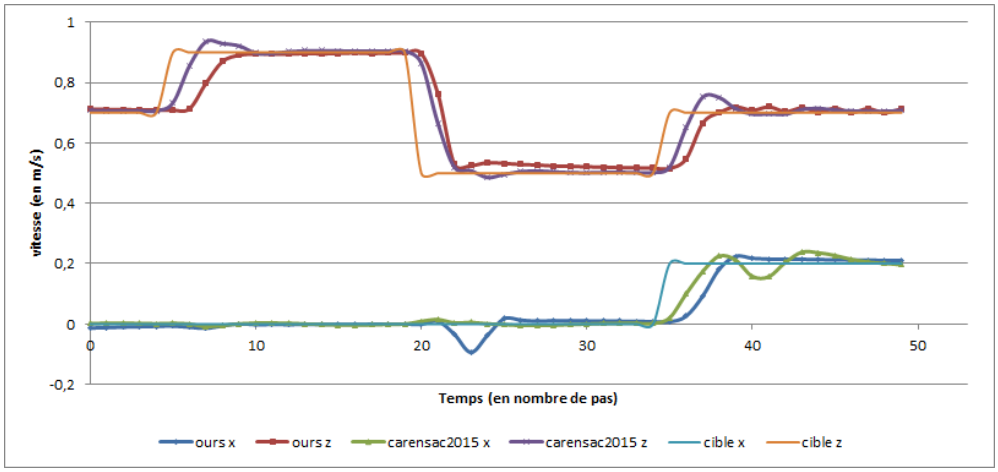



**Figure 4.9.:** Vitesses coronale (axe x) et sagittale (axe z) moyennes observées au cours du temps lorsqu'un changement de vitesse désirée est demandé (lignes verticales noires. Dans l'ordre, les vitesses cibles sont $(0; 0.7)ms^{-1}$, $(0; 0.9)ms^{-1}$, $(0; 0.4)ms^{-1}$ et $(0, 2; 0.7)ms^{-1}$.

valides pour le minimum ou le maximum, le processus s'arête même si la fréquence cible n'a pas été atteinte.

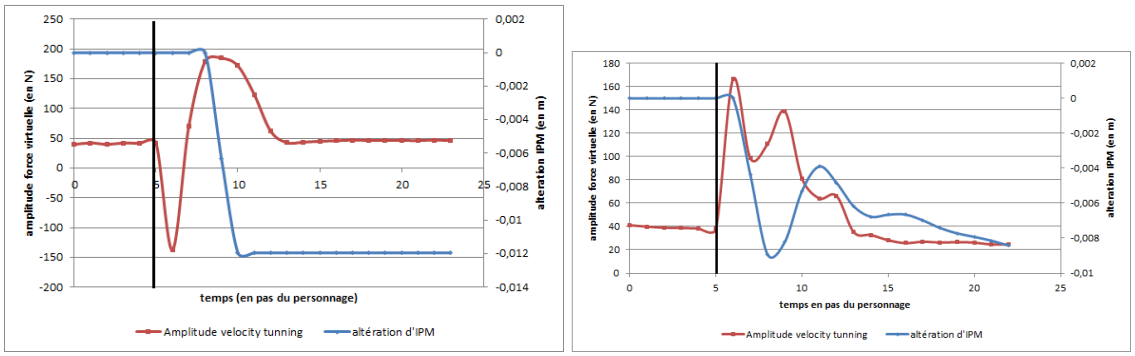
## 4.4 Résultats

Cette section s'organise en trois parties. Dans un premier temps nous comparons les résultats de notre contrôleur avec ceux obtenus par notre contrôleur précédent [CPB15]. Dans un second temps, nous présentons les résultats relatifs à notre étude des gains ainsi que ceux associés au *stabilisateur de contacts*. Enfin nous comparons les résultats obtenus par notre contrôleur avec une fréquence élevée et une fréquence faible.

### 4.4.1 Contrôle à haute fréquence

Nous réalisons la comparaison des deux systèmes avec des paramètres aussi proches que possibles. Le mouvement de référence utilisé au cours de cette comparaison est un mouvement de marche similaire à celui utilisé par Yin et al. [YLV07], avec les orientations cibles des articulations de la jambe en phase de vol contrôlées par l'IPM. Chaque contrôleur a son propre ensemble de gains afin d'assurer un contrôle robuste. La différence majeure entre les deux ensembles est que le contrôleur de cette thèse utilise des gains plus faibles pour les articulations se trouvant dans la jambe en phase d'appui, du fait de l'extension de la compensation de la gravité aux articulations la composant. La fréquence de simulation est fixée à 2000 Hz, fréquence utilisée dans les travaux utilisant la SIMBICON dont notre publication de 2015. Chacun des tests proposés dans cette section est précédé d'une période de stabilisation de plusieurs pas du personnage de manière à assurer que les contrôleurs soient à un état stable. Dans cet état la vitesse du personnage est de (0 ;0,7) (vitesse coronale nulle et vitesse sagittale de $0, 7ms^{-1}$)


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés




(a) (b)

**Figure 4.10.:** Évolutions de l'amplitude de la force virtuelle appliquée par le *velocity tuning* et de la valeur de l'altération d'IPM en fonction du temps à la suite d'une modification de la vitesse désirée. La ligne verticale indique le moment du changement de vitesse cible de $(0; 0.7)ms^{-1}$ à $(0; 0.3)ms^{-1}$. La figure (a) présente l'évolution observée dans le contrôleur proposé dans cette thèse et la figure (b) présente le résultat obtenu par le contrôleur [CPB15].

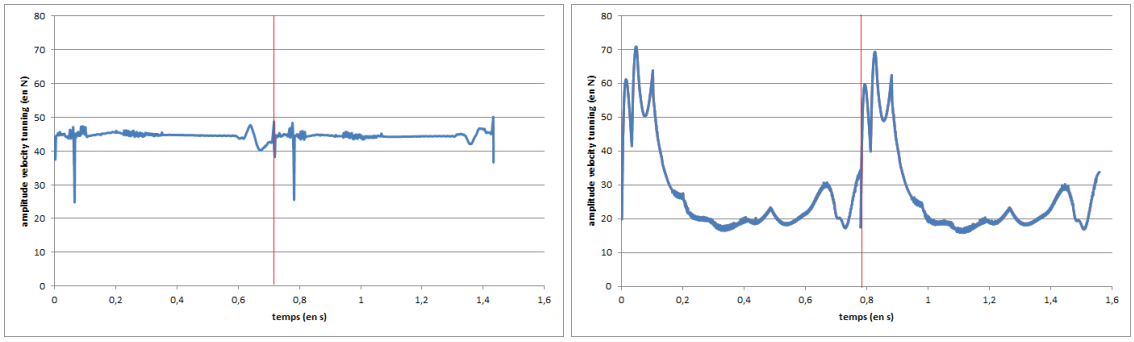
### Contrôle de la vitesse

En premier, nous proposons de comparer la capacité des deux contrôleurs à suivre la vitesse désirée lorsque celle-ci est modifiée (figure 4.9). Pour effectuer cette comparaison nous faisons varier la vitesse désirée du personnage. Bien que les deux contrôleurs atteignent exactement la vitesse désirée, nous pouvons remarquer que notre contrôleur atteint un état stable plus rapidement (en générale trois pas). Nous pouvons facilement remarquer que le contrôleur de 2015 a tendance à dépasser la vitesse cible ce qui cause l'apparition d'oscillations, en particulier lorsque la vitesse cible coronale est modifiée. L'absence de ce phénomène dans le contrôleur proposé dans cette thèse est principalement due à notre gestion globale des systèmes de contrôle de la vitesse de personnage. Cependant, cette gestion globale semble également avoir comme effet un temps de réaction plus grand. Enfin, nous pouvons remarquer que lors du second changement de vitesse cible une perturbation sur le maintien de la vitesse coronale apparaît dans notre contrôleur. Ceci est dû au changement important et brusque de la taille des pas entre $(0; 0.9)ms^{-1}$ et $(0; 0.4)ms^{-1}$ et au fait que nous spécifions la hauteur du pied de la jambe en phase de vol relativement à la hanche et non pas relativement au sol. Le but de cette modification était d'obtenir un résultat plus naturel en cas de perturbation brusque. Cependant, il a pour effet de rendre le premier pas suivant une réduction importante de la vitesse désirée très long car la position verticale de la hanche est moins proche du sol à la fin d'un pas lorsque la vitesse désirée est plus faible. Cette durée plus longue du pas va activer soudainement de nombreux points jusqu'à présent inactifs des courbes d'apprentissage du *velocity tuning*. Or, comme nous l'avons présenté, les points inactifs ont une valeur observée estimée par extrapolation linéaire des derniers points actifs. Il est donc probable que leur valeur soit erronée pendant les pas suivant un changement de vitesse du personnage. Ceci va provoquer une perturbation de la force virtuelle appliquée sur la fin du pas ce qui entraine la perturbation observée.

Nous pouvons vérifier l'efficacité de notre contrôle global de la vitesse en regardant l'évolution de l'amplitude de la force virtuelle appliquée par le *velocity tuning* et l'évolution de la valeur de l'altération d'IPM (figures 4.10a et 4.10b). La figure 4.10b permet de bien visualiser le problème d'interférence entre les deux systèmes présents dans l'ancien contrôleur affectant la



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés




(a) (b)

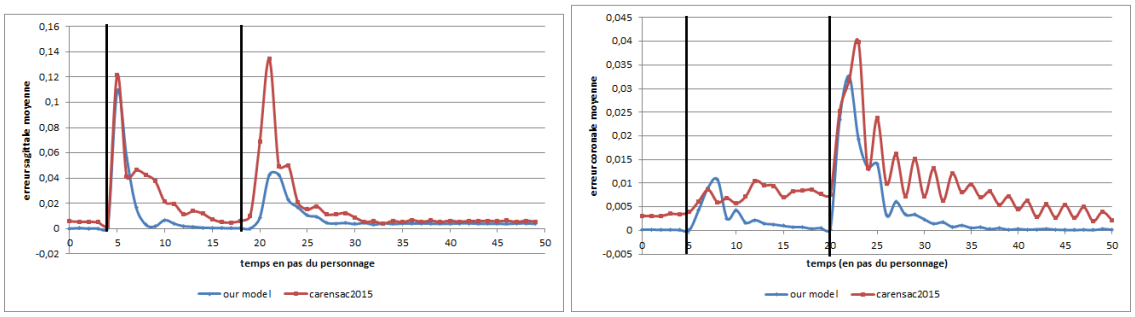
**Figure 4.11.:** Évolution de l'amplitude de la force virtuelle appliquée par le *velocity tuning* au cours de deux pas du personnage. La figure (a) présente l'évolution observée dans le contrôleur proposé dans cette thèse et la figure (b) présente le résultat obtenu par le contrôleur [CPB15]. La ligne verticale noire représente le début d'un nouveau pas du personnage.

vitesse du personnage par l'apparition d'oscillations limitant la vitesse de convergence vers un état stable. . Dans la figure 4.10a nous pouvons voir que ce phénomène a été complètement éliminé dans le contrôleur présenté dans cette thèse. Ceci permet une complète stabilisation du système au bout de 7 pas du personnage alors que ceux-ci sont encore dans une phase d'évolution au bout de 12 pas pour le contrôleur de 2015.

Une propriété importante que nous avons apportée dans notre nouveau contrôleur est une meilleure dissociation des deux composants internes du *velocity tuning* : la courbe d'apprentissage et le décalage global. Pour rappel, le but principal de l'utilisation du système d'apprentissage de la courbe d'évolution de la vitesse est de faire en sorte que la force virtuelle appliquée reste cohérente au cours d'un pas. En particulier, il est important qu'en l'absence de perturbation, la force virtuelle reste stable, c.à.d relativement constante, au cours d'un pas. Les figures 4.11b et 4.11a présentent l'évolution de l'amplitude de la force virtuelle au cours de deux pas consécutifs lorsque chaque contrôleur a convergé vers un état stable. On peut voir que malgré l'utilisation d'une courbe d'apprentissage, de très fortes variations de la force virtuelle sont présentes au cours d'un pas dans le contrôleur de 2015. En particulier, la force appliquée est très forte au début du pas. Ceci est probablement dû au fait qu'à l'état stable ce contrôleur utilise une force virtuelle opposée au sens du mouvement. Une amplitude élevée au début du pas indique que le *velocity tuning* a tendance à ralentir le personnage au début du pas. Or, l'ajout de la courbe d'apprentissage avait pour but de retirer ce phénomène. Dans le cas de notre contrôleur nous pouvons voir que la force virtuelle appliquée est bien plus stable, indiquant un meilleur apprentissage de l'évolution de la vitesse du personnage au cours du pas. Nous pouvons tout de même noter un pic sur une courte durée légèrement après le début du pas. Ceci correspond au moment où l'avant du pied entre en contact avec le sol. Cet instant et le moment où le talon entre en contact avec le sol sont les deux moments où les contacts varient le plus ce qui peut expliquer les instabilités dans l'amplitude de la force virtuelle appliquée.

Enfin nous pouvons comparer la capacité du *velocity tuning* à apprendre l'évolution de la vitesse dans chaque contrôleur. Pour ce faire, nous regardons l'erreur observée entre la courbe d'apprentissage et la courbe de vitesse observée à chaque pas de simulation avant la phase d'apprentissage. Pour obtenir une métrique d'erreur commune aux deux contrôleurs nous avons dû recaler les deux courbes dans le contrôleur de 2015 avant de calculer l'erreur.


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés




(a) (b)

**Figure 4.12.:** Erreur observée entre la courbe d'apprentissage du *velocity tuning* et la courbe de vitesses observées en fonction du temps. La figure (a) compare l'erreur observée sur la courbe sagittale entre les deux contrôleurs et la figure (b) représente l'erreur sur l'axe coronal. Les lignes verticales noires représentent des changements de vitesse cible. Les trois vitesses cibles utilisées sont dans l'ordre $(0; 0.7)ms^{-1}$ à $(0; 0.3)ms^{-1}$ et $(0, 2; 0.7)ms^{-1}$

Les résultats sont présentés dans les figures 4.12a et 4.12b. Nous pouvons voir que notre contrôleur converge plus rapidement vers une erreur négligeable entre les deux courbes. De plus la phase de convergence est beaucoup plus stable avec notre contrôleur comme l'indique l'absence d'oscillation sur la courbe d'erreur sagittale contrairement au contrôleur de 2015. Ces oscillations sont beaucoup plus visibles sur l'axe coronal du fait de la nécessité d'utilisation de deux courbes d'apprentissage suivant la jambe en phase de vol. Malgré l'importante amélioration de la stabilité de la convergence dans notre contrôleur nous pouvons tout de même observer de légères oscillations sur l'erreur coronale. Également nous pouvons remarquer que la convergence sur l'axe coronal est plus lente, probablement du fait de l'utilisation de deux courbes.

La courbe d'erreur coronale nous permet d'observer un phénomène étrange. Bien que la première modification de la vitesse cible ne change pas la vitesse cible coronale, nous pouvons voir des variations de l'erreur. Cela est dû au fait que le changement de vitesse sagittale est tellement important (de $0.7ms^{-1}$ à $0.3ms^{-1}$) qu'il a introduit des perturbations sur la vitesse coronale observée dans la simulation. Ceci nous permet de remarquer que l'apprentissage de notre contrôleur présente une meilleure capacité à résister à ce genre de perturbations.

### Résistance aux perturbations externes

Pour évaluer plus globalement la résistance aux perturbations de notre contrôleur, nous avons regardé l'évolution de la vitesse du personnage lors de l'application de forces externes. Pour réaliser cette expérimentation nous avons appliqué des forces sur le bassin du personnage sur une durée de 0.3 secondes. La figure 4.13 présente une comparaison des résultats obtenus lors de l'application de deux forces d'une intensité d'environ $140N$. Sur cette figure nous pouvons voir que notre contrôleur possède une bien meilleure robustesse aux perturbations externes que le contrôleur précédent. En effet, nous pouvons voir que notre contrôleur est capable de retourner à la vitesse désirée sans créer d'oscillations autour de celle-ci, contrairement au contrôleur de 2015.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

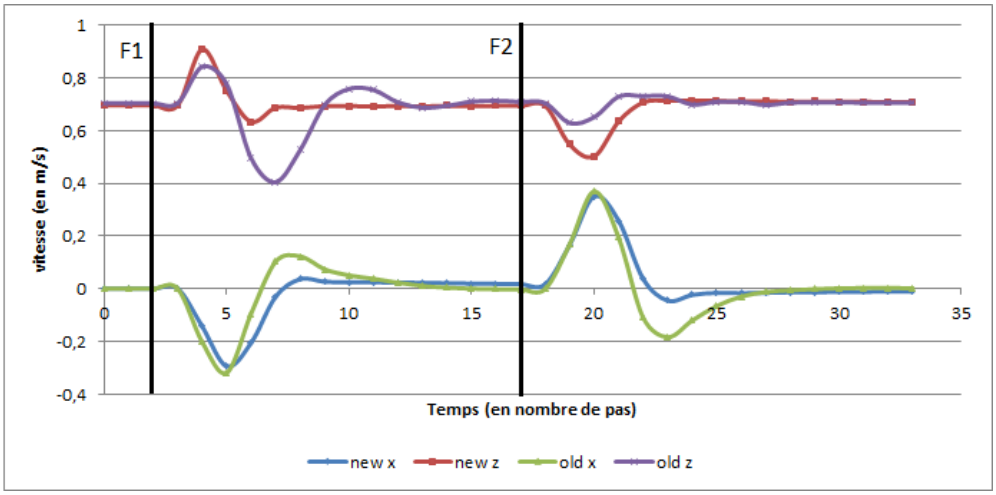


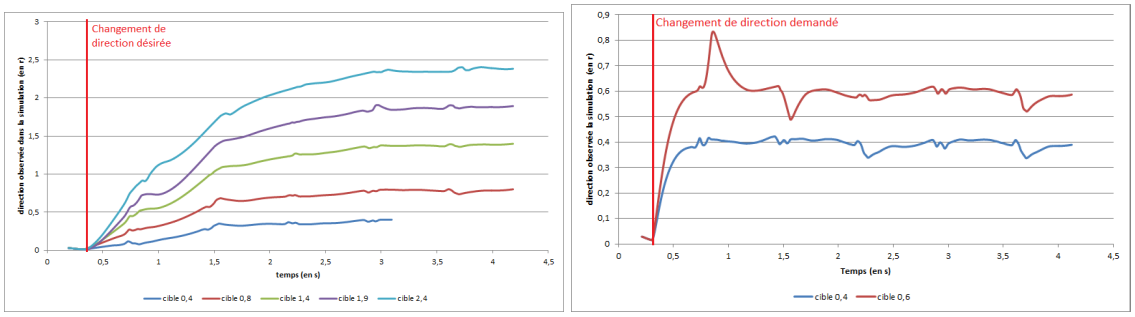

**Figure 4.13.:** Vitesses coronale (axe x) et sagittale (axe z) moyennes observées au cours du temps lors de l'application de forces externes sur le bassin du personnage. Les lignes verticales noires indiquent le moment où chacune des forces est appliquée. Ces forces ont pour intensité $140N$ et sont maintenues durant $0.3s$. Les directions des forces sont (-1,0,1) et (1,0,1) l'axe $z$ étant l'axe sagittal.

### Contrôle de la direction

Le dernier composant à évaluer est notre gestion des changements de direction désirée. Comparer la gestion de l'orientation présentée dans cette thèse à celle du contrôleur dans [CPB15] n'est pas forcément pertinent. En effet, comme nous l'avons vu, notre contrôleur est bien plus robuste aux perturbations. Cela lui permet donc d'avoir une meilleure capacité à supporter les changements de direction car la période de transition entre les deux directions est équivalente à un état perturbé. Donc pour cette évaluation nous utilisons deux variations de notre contrôleur, l'une utilisant notre gestion de la direction et l'autre appliquant des changements instantanés de manière similaire aux travaux précédents [YLV07 ; CBV10 ; CPB15]. Pour chacune de ces deux configurations nous modifions la direction désirée une fois que le contrôleur a atteint un état stable et que le personnage est vers le milieu d'un pas, moment où nous sommes assurés que le pied est correctement ancré au sol. Les résultats de cette expérience sont représenté dans les figures 4.14a et 4.14b. Pour les changements de direction élevés la durée pour un pas est différente de la durée prévue pour le mouvement de marche simulé. Il est possible de déterminer la fin de chaque pas du personnage en regardant la présence des perturbations qu'elle provoque et en considérant qu'un pas dure environ 0.75s.

La figure 4.14b montre le résultat obtenu avec la modification instantanée. Cette expérience ne contient que deux changements de direction ; 0.4 radians et 0.6 radians, car dès 0.8 radians cette technique ne permet pas de produire une simulation stable. Pour les changements de direction plus importants nous avons observé une rotation de la jambe d'appui du fait de l'intensité très importante des moments appliqués sur la hanche de la jambe d'appui pour réussir à produire le changement d'orientation désiré ce qui entraine la chute du personnage. Par comparaison, la figure 4.14a illustre le fait que notre système de modification nous permet d'effectuer des changements de direction jusqu'à 2.4 radians, environ $140°$. Nous pouvons tout de même remarquer que cette technique n'est pas parfaite. Seuls les changements de direction relativement faibles ont réellement atteint leur objectif au bout des deux pas prévus par notre système (la fin du second pas se situe à environ 1.5 secondes). Pour les changements d'orientation supérieurs le personnage a effectué environ $80\%$ du changement d'orientation


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés




**(a)** **(b)**

**Figure 4.14.:** Évolution au cours du temps de la direction du personnage observée dans la simulation lorsque la direction désirée est modifiée. La figure (a) présente les résultats obtenus avec l'utilisation de notre système de modification de la direction cible et la figure (b) représente les résultats obtenus avec un changement de direction cible instantané utilisé dans les systèmes antérieurs ([YLV07 ; CPB15]). La ligne verticale noire représente l'instant où la direction cible est modifiée. Chaque pas du personnage dure environ 0.75s et leur fin est visible au moment où les perturbations induites.

total après deux pas. Pour le changement de direction de 1.4 radians la partie manquante correspond à seulement 0.3 radians ce qui est assez faible pour ne pas perturber le résultat visuel. Pour 1.9 radians et 2.4 radians le troisième pas nécessaire est bel et bien visible. Cependant, nous pouvons remarquer un phénomène intéressant. À la fin du second pas ces deux cas ont permis d'atteindre respectivement 1.5 et 1.9 radians ce qui indique qu'il est en effet possible d'effectuer des changements d'orientation de ces amplitudes en deux pas. Pour ce faire il faudrait simplement spécifier une cible plus importante que la cible que nous souhaitons effectivement atteindre ou utiliser plus de $50\%$ de la direction désirée pour la cible intermédiaire. Nous avons cependant choisi de ne pas implémenter un tel système car cela ne nous semble pas être l'approche à aborder car elle ne permettrait pas de gérer les changements d'orientation supérieurs à 2.4 radians.

### 4.4.2 Stabilisateur de contacts

Cette section présente l'impact des différentes variations du modèle réduit, évoqué en section 4.3.1, sur la qualité des contacts ainsi que les ressources de calcul nécessaires. Pour comparer les modèles réduits, nous simulons une marche de 200 pas. Nous faisons varier la vitesse coronale cible entre $0m.s^{-1}$, $0.2m.s^{-1}$ et $-0.2m.s^{-1}$ tous les 5 pas pour introduire des perturbations.

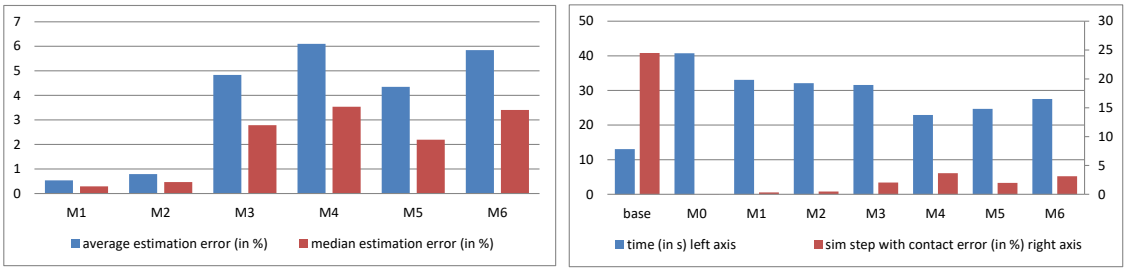
Cette comparaison est effectuée sur 6 modèles réduits chacun décrivant une différente combinaison de parties simplifiées du personnage. Les parties simplifiées pour chaque modèle sont :

- **modèle** $M1$ **:** bras
- **modèle** $M2$ **:** bras et tête
- **modèle** $M3$ **:** bras, tête et torse
- **modèle** $M4$ **:** bras, tête, torse et jambe en phase de vol
- **modèle** $M5$ **:** bras, tête et jambe en phase de vol

Il est important de noter que pour les modèles $M4$ et $M5$, la jambe qui sera la jambe en phase de vol au cours du pas est également simplifiée au début du pas lorsque le personnage est



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés




**(a)** **(b)**

**Figure 4.15.:** (a) Erreur sur la force au niveau des contacts par chaque modèle réduit (équation 4.13) ; (b) rouge : pourcentage de la simulation présentant des contacts instables, bleu : temps total de simulation pour 200 pas du personnage. 'base' représente les résultats obtenus par le contrôleur lorsque le stabilisateur de contacts est désactivé.

dans une phase de double support. Nous avons également testé un modèle dynamique ($M6$) où tout corps rigide ne se trouvant pas dans une jambe en contact avec le sol est simplifié. En pratique, ce modèle correspond à $M3$ durant la phase de double support au début du pas puis à $M4$ lorsqu'une seule jambe touche le sol. Un modèle $M0$ utilisant le personnage complet sert de base pour la comparaison.

Nous commençons par évaluer l'erreur introduite dans la simulation par l'utilisation du modèle réduit. Cette erreur est définie comme la différence moyenne des contacts entre le modèle réduit et le modèle complet ($M0$). Pour chaque pas de temps, nous calculons la somme des différences en valeur absolue de la composante normale des GRF sur chaque sommet du pied et des orteils entre le modèle réduit et le modèle complet. Cette valeur est exprimée en pourcentage de la somme des GRF obtenues avec $M0$.

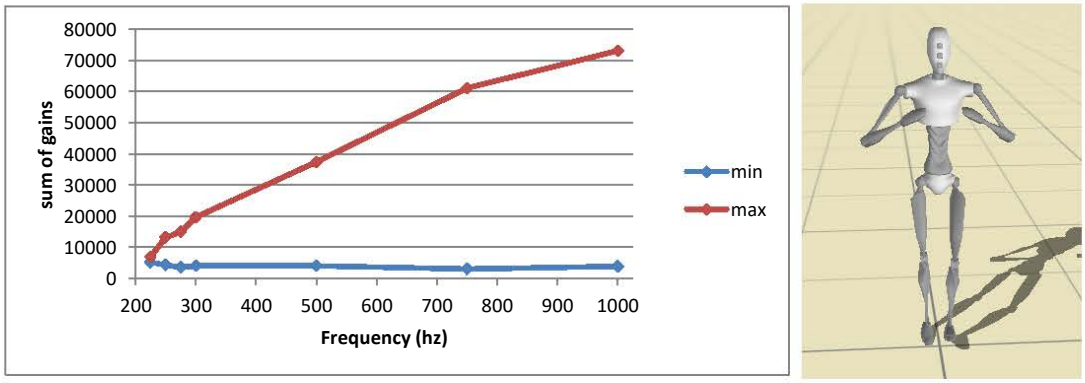
$$Err = \frac{\sum_i abs(F_{0i} - F_{ki})}{\sum_i F_{0i}} \tag{4.13}$$

avec $k$ une configuration et $i$ chacun des points de contact.

Comme nous pouvons le voir dans la figure 4.15a, l'erreur introduite par la simplification des bras et de la tête est négligeable (inférieure à $0.5\%$ des forces totales). Les résultats de $M4$ et $M6$ nous montrent que la conservation de la seconde jambe lors de la phase de double support ne permet pratiquement pas de nous rapprocher des contacts sur la jambe d'appui observés avec le modèle complet. Enfin, la comparaison de $M3$ et $M5$ montre que la simplification de la jambe en phase de vol introduit moins d'erreur dans les contacts que la simplification du torse. Bien que contre intuitif, ce résultat est probablement dû la masse importante du torse causant un effet d'inertie absent après simplification. Ce résultat est particulièrement intéressant car le modèle $M5$ contient moins de corps rigides que le modèle $M3$ et donc permet de meilleures performances.

Notre seconde évaluation consiste à étudier l'impact du modèle réduit sur les contacts obtenus ainsi que le temps de calcul pour les 200 pas de simulation. Nous définissons les contacts comme présentant une erreur lorsque la distribution observée des GRF n'est pas incluse dans les bornes définies précédemment en section 4.3.1 (c.à.d. lorsque le barycentre des GRF ne se trouve pas dans la zone cible de la figure 4.7b). Nous comparons nos résultats avec ceux obtenus lorsque le stabilisateur de contacts est désactivé. Comme nous pouvons le voir dans la figure 4.15b, le nombre de pas de simulation présentant une erreur dans les contacts est grandement réduit par notre système pour tous les modèles ($M1$ à $M6$). Conformément à


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés




(a) (b)

**Figure 4.16.:** La figure (a) présente Représentation de la somme des gains $\sum K_p + K_d * 10$ correspondant aux ensembles de gains minimums (bleu) et maximums (rouge) pour chaque fréquence. La figure (b) présente la position cible utilisée pour les mains utilisée au cours de l'optimisation des gains.

nos attentes, on remarque que $M3$ et $M5$ présentent un niveau de stabilité similaire, avec $M5$ nécessitant un temps de calcul plus faible (environ $30\%$), confirmant la plus grande importance du torse par rapport à celle de la jambe en phase de vol. En comparant les résultats pour $M4$ et $M5$, on remarque que l'ajout du torse permet une réduction du nombre de pas de temps présentant un déséquilibre par un facteur de deux pour une augmentation de moins de $7\%$ du temps de calcul. Enfin, on remarque que la simplification des bras et de la tête ($M1$ et $M2$) permet un gain de temps d'environ $20\%$ par rapport au modèle complet $M0$ tout en corrigeant presque totalement les erreurs.

Nous avons donc considéré deux modèles réduits dans la suite de nos expérimentations :

— $M2$ **:** bras et tête, pour les cas où l'on désire une réduction maximale des instabilités au niveau des contacts tout en évitant les calculs superflus.

— $M5$ **:** bras, tête et jambe en phase de vol, si l'on désire une solution permettant d'obtenir un équilibre entre ressources de calcul et correction des instabilités.

### 4.4.3 Etude des gains des régulateurs PD

Pour tester nos hypothèses sur l'évolution des gains des régulateurs PD en fonction de la fréquence de simulation nous avons utilisé le processus d'optimisation présenté dans la section 4.3.2. Le point initial de l'optimisation est composé des valeurs utilisées dans l'article du SIMBICON [YLV07] pour un mouvement de marche droite. Nous avons placé la position cible des mains légèrement en avant du torse (figure 4.16b). Pour introduire des perturbations sur le personnage au cours de l'optimisation nous introduisons des modifications de la vitesse désirée coronale ainsi que des changements de direction cible pour nous assurer de la robustesse des ensembles de gains obtenus.

L'évolution globale des gains en fonction de la fréquence que nous avons obtenue est donnée dans la figure 4.16a. Nous pouvons voir que, conformément à nos hypothèses, la taille de l'intervalle des gains ainsi que leurs valeurs diminuent avec la fréquence de simulation. Notre processus d'optimisation a été capable de trouver des gains produisant un résultat similaire à celui obtenu pour la fréquence de simulation initiale pour des fréquences jusqu'à 225 Hz. Il est cependant important de noter que pour les fréquences de simulation en dessous de 300 Hz



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

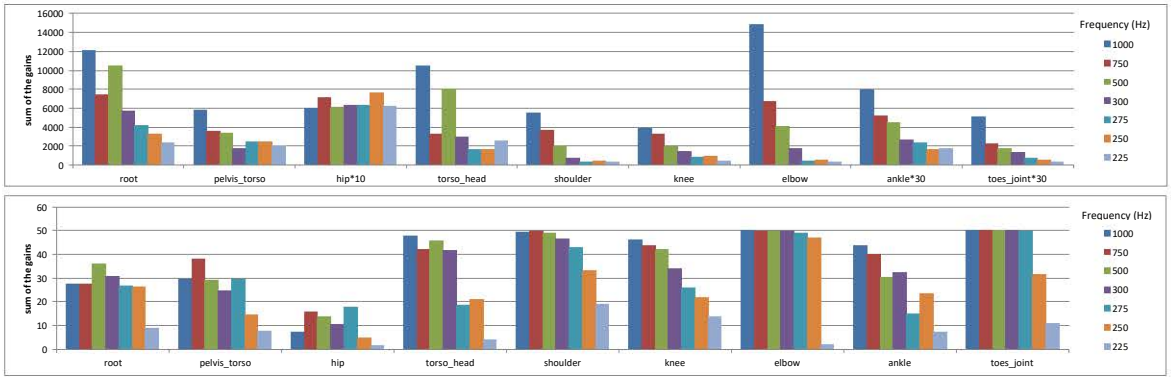



**Figure 4.17.:** Haut : Évolution de la moyenne de la somme des gains $K_p + K_d * 10$ en fonction de la fréquence pour chaque articulation. Pour que les valeurs de toutes les articulations soient lisibles sur un histogramme commun les valeurs pour la hanche ont été multipliées par 10 et celles pour les chevilles et orteil par 30 ; Bas : évolution de l'intervalle de la somme des gains $K_p + K_d * 10$ entre les valeurs de l'ensemble de gains minimaux et maximaux (en $\%$ de la valeur moyenne)

.

le contrôleur perd grandement de sa robustesse aux perturbations externes. Ces contrôleurs moins stables peuvent tout de même être utilisés pour obtenir un aperçu rapide du résultat lors de la modification de poses clefs.

La figure 4.17 présente les résultats par articulation. Nous pouvons tout d'abord noter sur le graphe du haut, présentant l'évolution de la valeur moyenne des gains en fonction de la fréquence, que cette réduction est un phénomène observable sur presque toutes les articulations. On remarque que les valeurs pour 750 Hz pour le bassin (*root*) et le cou (*torso_head*) semblent être dues à un minima local. Également, les valeurs des gains pour la hanche (*hip*) sont très faibles pour l'une des articulations centrales du personnage. Elles sont multipliées par un facteur de 10 sur le diagramme pour être correctement visibles. De plus, on remarque que le gain pour cette articulation est quasiment constant et évolue peu avec la fréquence de simulation. Avant de discuter de ce phénomène étudions l'évolution de la taille de l'intervalle des gains en fonction de la fréquence (figure 4.17 (bas)). Le comportement général que nous observons se découpe en deux phases. Pour les fréquences de simulation au-dessus d'une certaine fréquence seuil, la taille de l'intervalle est grande et constante. En dessous de cette fréquence, la taille de l'intervalle va soit diminuer lentement ou brusquement. Cette fréquence seuil n'est pas la même pour toutes les articulations. Pour la majeure partie des articulations cette fréquence de simulation est entre 250 Hz et 300 Hz et est associée à une chute brusque de la taille de l'intervalle. Pour le genou et la cheville elle se situe à respectivement 500 Hz et 750Hz et est associée à une diminution plus continue. Encore une fois la hanche semble avoir un comportement particulier. On peut remarquer une diminution presque continue de la taille de l'intervalle en fonction de la fréquence. Ceci nous laisse supposer que pour la hanche, les valeurs de fréquence étudiées ne nous laissent observer que la seconde phase et que la fréquence seuil est bien plus grande que 1000Hz étant donné que pour 750Hz la taille de l'intervalle est déjà extrêmement faible (inférieure à $20\%$ de la valeur moyenne). On peut noter un autre phénomène intéressant, toutes les articulations pour lesquelles la valeur seuil se distingue de celle globalement observée se trouvent dans la jambe. Hors au cours du mouvement ces articulations sont les seules qui auront à supporter deux rôles. Lorsque la jambe est en phase de support, ces articulations doivent supporter le personnage et lorsqu'elle est en phase de vol elles doivent permettre de contrôler la position


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

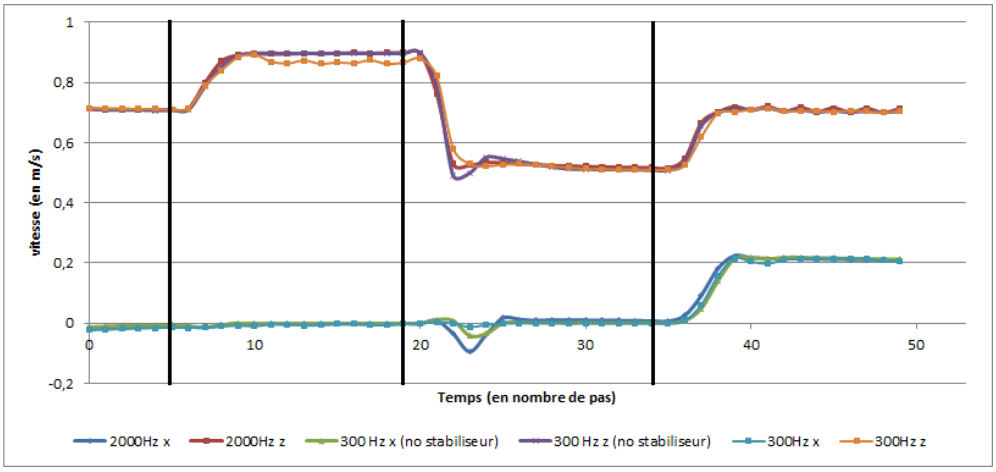



**Figure 4.18.:** Vitesses coronale (axe x) et sagittale (axe z) moyennes observées au cours du temps lorsqu'un changement de vitesse désirée est demandé au contrôleur. Les lignes verticales noires indiquent le moment où le changement de vitesse désirée est spécifié. Dans l'ordre les vitesses cibles sont $(0; 0.7)ms^{-1}$, $(0; 0.9)ms^{-1}$, $(0; 0.4)ms^{-1}$ et $(0, 2; 0.7)ms^{-1}$. Nous avons également affiché le résultat obtenu lorsque le stabilisateur de contact est désactivé (pour pouvoir évaluer son impact sur le contrôle de la vitesse).

des différentes parties de la jambe. Ceci explique probablement pourquoi les valeurs des gains de la hanche sont si faibles. Étant donné que selon si la jambe est en phase d'appui ou en phase de vol les composants du contrôleur appliquant des moments sur celle-ci sont tous différents, il est nécessaire d'avoir des valeurs de gains faibles pour obtenir un résultat stable dans les deux cas. La raison pour laquelle les gains de la cheville et des orteils sont si faibles est probablement dû au fait que ce sont les parties du personnage en contact avec le sol et que la masse des corps rigides du personnage qu'elles contrôlent durant la phase de vol est très faibles.

### 4.4.4 Contrôle à basse fréquence

Nous comparons les résultats obtenus pour une fréquence de 2000Hz (section 4.4.1) aux résultats obtenus pour une fréquence de simulation de 300Hz. Notre contrôleur devrait nous permettre d'obtenir les mêmes résultats indépendamment de la fréquence de simulation.

Nous pouvons commencer par examiner si notre contrôleur produit des résultats similaires pour les deux fréquences de simulation lorsque la vitesse désirée est modifiée. Cette comparaison est présentée dans la figure 4.18. Nous visualisons également pour la fréquence faible, le résultat obtenu lorsque le stabilisateur de contacts est désactivé. L'intérêt de cette comparaison est que, vu que ce système ajoute des moments additionnels dans la jambe de support, il est probable qu'il perturbe le *velocity tuning* et donc la capacité globale du contrôleur à suivre la vitesse désirée. La première remarque que nous pouvons faire est que notre contrôleur produit un résultat similaire pour les deux fréquences de simulation. Nous pouvons tout de même noter deux phénomènes intéressants. Le premier est que le stabilisateur de contact perturbe bel et bien le contrôle de la vitesse du personnage. En l'utilisant à 300 Hz la vitesse de $0, 9ms^{-1}$ n'est pas atteinte de manière stable. Cependant, il permet de complètement corriger la perturbation que nous observons lors de la forte réduction de la vitesse désirée sans son utilisation (second changement de vitesse).



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

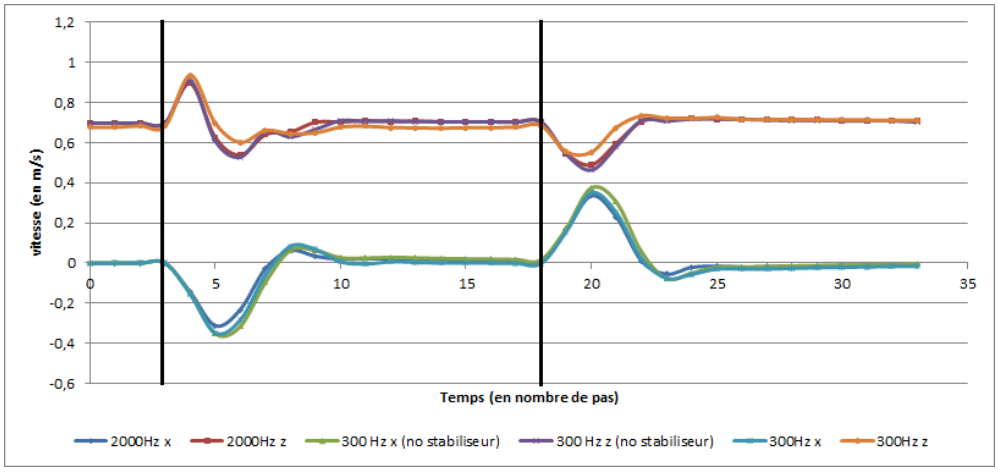
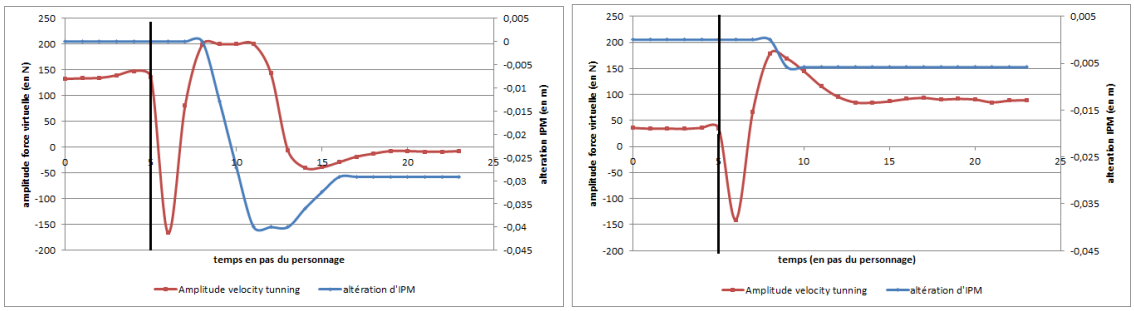



**Figure 4.19.:** Vitesses coronale (axe x) et sagittale (axe z) moyennes observées au cours du temps lors de l'application de forces externes sur le bassin du personnage. Les lignes verticales noires indiquent le moment où chacune des forces est appliquée. Ces forces ont pour intensité $140N$ et sont maintenues durant $0.3s$




**(a)** **(b)**

**Figure 4.20.:** Évolution de l'amplitude de la force virtuelle appliquée par le *velocity tuning* et la valeur de l'altération d'IPM en fonction du temps à la suite d'une modification de la vitesse désirée à une fréquence de 300Hz. La ligne verticale indique le moment du changement de vitesse cible de $(0; 0.7)ms^{-1}$ à $(0; 0.3)ms^{-1}$. La figure (a) présente l'évolution observée avec le stabilisateur de contacts et la figure (b) présente le résultat obtenu lorsque qu'il est désactivé.

La figure 4.19 illustre les résultats obtenus lorsque nous reproduisons l'application des forces externes de 140N utilisées lors de la comparaison avec le contrôleur de [CPB15]. Les trois configurations obtiennent des résultats similaires. Nous pouvons noter que l'utilisation du stabilisateur de contact permet une légère amélioration de la robustesse sans causer de perturbation dans le contrôle de la vitesse ce qui renforce notre idée que les perturbations qu'il introduit sont liées aux systèmes d'apprentissage de l'altération d'IPM et du *velocity tuning*.

Les figures 4.20a et 4.20b nous permettent de visualiser en détail la période d'apprentissage lors d'une forte réduction de la vitesse cible (de $0.7ms^{-1}$ à $0.3ms^{-1}$). Ces figures confirment bien les résultats précédents : le stabilisateur de contact introduit de fortes interférences dans les systèmes d'apprentissage permettant de respecter les changements de vitesse désirée. Il double quasiment le nombre de pas nécessaires à la stabilisation des deux systèmes et provoque l'utilisation de valeurs beaucoup plus importantes pour l'altération d'IPM. Il est cependant important de noter que le système a réellement convergé au $13^{eme}$ pas. Si nous


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

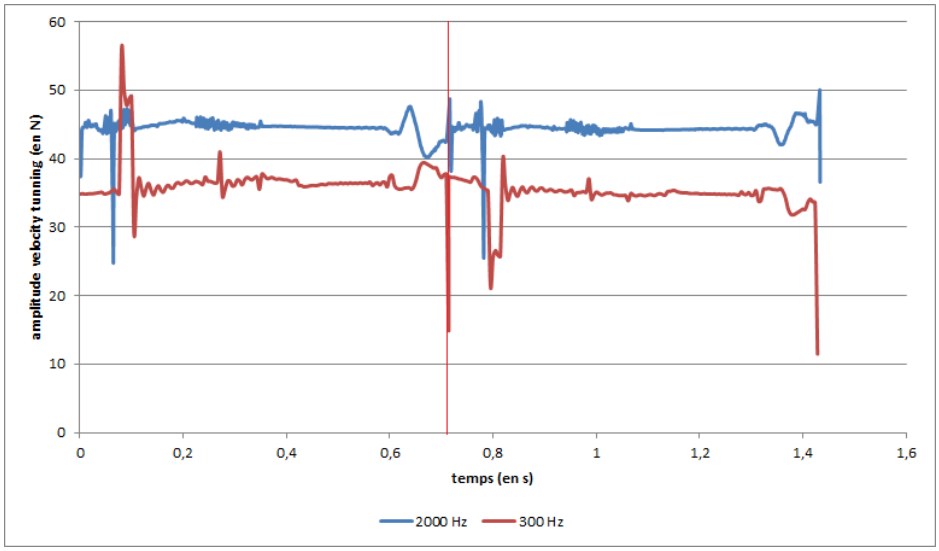



**Figure 4.21.:** Évolution de l'amplitude de la force virtuelle appliquée par le *velocity tuning* au cours de deux pas du personnage. La ligne verticale noire représente le début d'un nouveau pas du personnage.

revenons à notre comparaison avec les résultats obtenus pour une fréquence de simulation élevée (figure 4.10a), nous pouvons voir que les résultats sont très similaires dans le cas n'utilisant pas le stabilisateur de contacts (figure 4.20b). Cette absence de différence nous montre que notre contrôleur fonctionne aussi bien à 300Hz que 1000Hz.

En ce qui concerne la stabilité de la valeur de la force virtuelle appliquée par le *velocity tuning* à l'état stable, nous pouvons voir qu'il n'y a pas de différence majeure entre les fréquences de simulation (figure 4.21). Nous pouvons tout de même voir que les perturbations correspondant au début de chaque pas et à l'instant où l'avant du pied touche le sol sont plus marquées pour la fréquence de simulation faible.

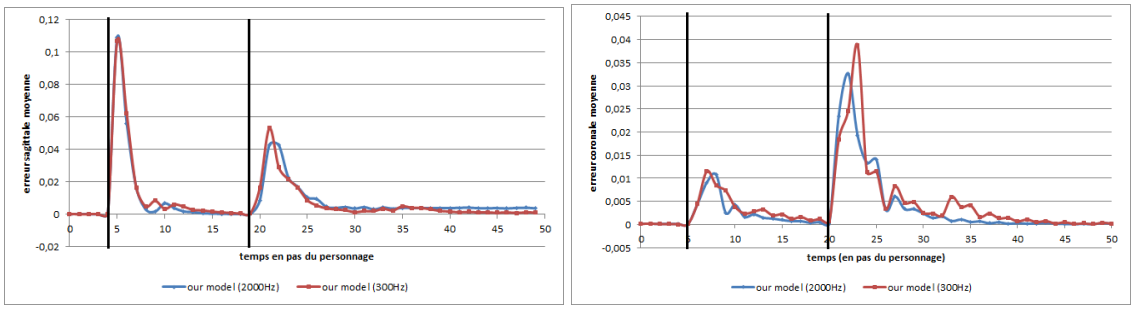
Les figures 4.22a et 4.22b donnent un aperçu de l'erreur entre les vitesses observées et les courbes d'apprentissage pour chaque fréquence de simulation. Comme nous pouvons le voir, la convergence sagittale est quasiment identique. En ce qui concerne l'axe coronal nous pouvons observer une faible réduction de la capacité à converger. Ceci peut probablement être expliqué par la difficulté supérieure à contrôler la convergence des deux courbes coronales simultanément. Cette difficulté est amplifiée par l'échantillonnage plus faible causé par la réduction du nombre de pas de temps de simulation dû à la fréquence de simulation plus faible.

Enfin nous pouvons évaluer les résultats obtenus par notre système de contrôle de la direction. Les résultats pour la fréquence élevée sont visibles dans la figure 4.14a et la figure 4.23 illustre nos résultats pour une fréquence de 300Hz. Comme nous pouvons le voir les résultats sont quasiment identiques. La seule différence est que la convergence vers la direction désirée est très légèrement plus lente pour les changements importants (supérieur à 1.9 radians).

Nous proposons enfin une dernière expérimentation pour illustrer de manière pratique l'avantage de l'utilisation d'une fréquence de simulation faible. En plus de simuler le personnage, nous ajoutons à la simulation une boite contenant des cubes simulés physiquement. Cette boite est mise en rotation de manière à générer une charge de calcul constante dans la simulation qui représente l'intégration du personnage dans un monde virtuel comprenant



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés




**(a)** **(b)**

**Figure 4.22.:** Erreur observée entre la courbe d'apprentissage du *velocity tuning* et la courbe de vitesse observée en fonction du temps pour les fréquences de simulation 2000Hz et 300Hz. La figure (a) montre l'erreur observée sur la courbe sagittale et la figure (b) représente l'erreur sur l'axe coronal. Les lignes verticales noires représentent des changements de vitesse cible. Les trois vitesses cibles utilisées sont dans l'ordre $(0; 0.7)ms^{-1}$ à $(0; 0.3)ms^{-1}$ et $(0.2; 0.7)ms^{-1}$

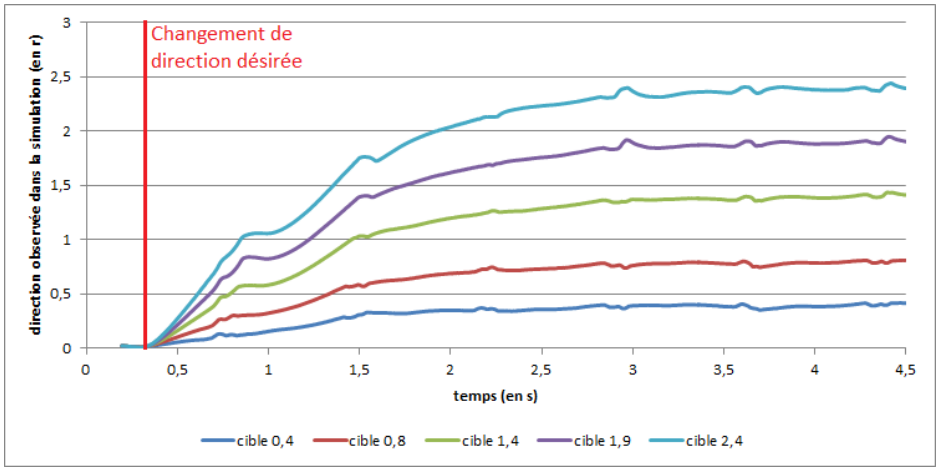



**Figure 4.23.:** Évolution au cours du temps de la direction du personnage observée dans la simulation lorsque la direction désirée est modifiée pour une fréquence de simulation de 300 Hz. La ligne verticale noire représente l'instant où la direction cible est modifiée. Chaque pas du personnage dure environ 0.75s et leur fin est visible au moment où les perturbations induites.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

un relativement petit nombre d'éléments dynamiques simulés. Pour cette expérience nous utilisons 30 cubes dynamiques produisant une moyenne de 40 collisions physiques par pas de temps de simulation. Avec une fréquence de simulation de 1000 Hz nous obtenons une moyenne de 16.34 images par seconde contre 29.56 avec une fréquence de simulation de 300 Hz. Cela veut dire que malgré l'utilisation coûteuse en temps de calcul du stabilisateur de contacts nous obtenons un facteur deux sur les performances.

### 4.4.5 Discussions

Nous avons vu dans la section 4.4.2 que le temps de calcul nécessaire à l'utilisation du stabilisateur de contacts est assez important. Même avec l'utilisation de la configuration présentant le meilleur équilibre de stabilité et temps de calcul ($M5$) on remarque que le temps total est deux fois plus élevé que celui nécessaire à l'exécution du contrôleur et de la simulation avec le composant désactivé. Ce temps de calcul élevé est explicable par le fait que l'implémentation utilisée pour obtenir ces résultats n'est pas une implémentation parallélisée. Nous pouvons estimer le gain de performance d'une implémentation parallèle de ce système car la quasi-totalité du temps de calcul est pris par les évaluations de l'état futur. En particulier, nous pourrions utiliser l'initialisation du CMA pour récolter notre point de comparaison pour l'évaluation des échantillons et nous pourrions paralléliser les évaluations des échantillons au sein de chaque itération du CMA (8 échantillons dans notre cas). Pour les cas où le système requiert une ou deux itérations du CMA, ce qui est le cas le plus courant (la moyenne est $1.4$), nous pourrions observer un gain de performances de respectivement $78\%$ et $80\%$ ( $78\%$ sur la première itération et $87.5\%$ sur les itérations suivantes). Cette parallélisation rendrait notre stabilisateur de contacts bien plus performant. Nous avons développé une implémentation parallélisée de ce composant. Cependant, il s'est avéré impossible d'exécuter en parallèle les pas de temps de plusieurs mondes virtuels créés avec le moteur physique ODE. Il existe une option étant supposé rendre ce type d'action possible mais nous ne sommes pas arrivés à la faire fonctionner.

Notre stabilisateur de contact présente une capacité limitée à prévenir les instabilités physiques. Nous avons remarqué au cours de nos expériences, qu'il n'était pas capable de trouver de solution stable dans certaines configurations. En particulier, l'une de ces configurations est lorsque l'état futur naturel (sans échantillon) de la simulation ne présente qu'un seul point de contact entre le sol et le pied alors que nous sommes dans une phase où les quatre sommets du pied devraient être en contact (avec le sol). Or, cette configuration est l'une des plus importantes à corriger car avec un seul point de contact rien ne peut empêcher une rotation de la jambe en phase d'appui. Deux modifications pourraient être apportées au système pour tenter de pallier cette limite. La première est l'ajout de points d'initialisation supplémentaires spécifiques aux cas extrêmes. Nous avons exploré cette piste, mais elle s'est avérée être très compliquée. En effet, lorsque les perturbations sont très élevées, il est parfois nécessaire d'appliquer des moments très élevés pour atteindre des contacts corrects. Hors comme nous l'avons vu les moments habituellement appliqués par ce système perturbent déjà énormément le contrôle de la vitesse du personnage. La seconde solution serait de réinstaurer une fenêtre de prédiction. Cependant cette approche augmenterait énormément le temps de calcul nécessaire, ce qui, associé à l'impossibilité pratique d'évaluer des échantillons en simultané avec ODE, ne permettrait pas d'obtenir des performances interatives.

Notre optimisation hors ligne permettant de déterminer les gains à utiliser pour chaque fréquence présente également une limite. Bien que nous soyons arrivés à déterminer des bornes minimales et maximales pour chaque variable, trouver l'ensemble de gains produisant



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

le résultat optimal est très complexe. Il ne s'est pas avéré possible de le déterminer par optimisation du fait des très nombreux minimas locaux. Nous avons également examiné la possibilité d'utiliser des gains différents pour les articulations des jambes suivant si elles sont en phase de vol ou en phase d'appui. La séparation des gains des jambes se justifie par le fait que toutes les articulations pour lesquelles les gains commencent à diminuer à partir d'une fréquence de simulation supérieure à la moyenne se trouvent dans les jambes. L'utilisation de deux gains en fonction de l'état de la jambe permettrait d'utiliser des gains spécialisés ce qui devrait permettre de trouver des ensembles de gains permettant d'obtenir des simulations stables à des fréquences de simulation plus faibles que la limite actuelle de notre système. Cependant, notre processus d'optimisation ne nous a pas permis de produire des résultats répétables entre plusieurs répétitions du processus. De plus, les résultats obtenus ne présentaient pas d'évolution ordonnée des gains en fonction de la fréquence. Plus de travaux dans cette direction seraient nécessaires pour déterminer la validité de l'idée de la séparation des gains.

## 4.5 Bilan

Dans ce chapitre nous avons présenté notre contrôleur. Celui-ci se présente comme une extension du modèle SIMBICON. Il contient notamment des versions améliorées des composants de contrôle de la vitesse utilisés dans notre contrôleur précédent [CPB15]. Nous avons ajouté à celui-ci une gestion de la direction désirée du mouvement permettant une plus grande liberté de manipulation du personnage.

Nous avons également ajouté à notre contrôleur un nouveau composant de stabilisation des contacts permettant de diminuer les instabilités introduites au niveau des pieds. Ce composant est entièrement indépendant du moteur physique utilisé.

Nous avons réalisé une étude montrant l'intrication des valeurs des gains des régulateurs PD à la fréquence de simulation utilisée. Cette étude a abouti à un processus d'optimisation hors ligne permettant de déterminer les intervalles de valeurs possibles pour chacun de ces gains.

Enfin nous avons montré que notre contrôleur est capable de produire un mouvement de marche stable et précis pour des fréquences de simulation faibles et élevées. Notre contrôleur permet d'obtenir une simulation temps réel et interactive pour toute fréquence de simulation allant de 300 Hz à 2000 Hz.


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

# 5 Simulation lagrangienne de fluide sur GPU

Comme nous l'avons vu dans la section 3.2.2, il existe de nombreux algorithmes pour réaliser une simulation de fluide lagrangienne. Nous avons choisi d'utiliser l'algorithme *divergence-free SPH* (DFSPH) proposé par Bender et Koschier [BK15]. Ce choix est motivé d'une part par les meilleures performances en temps d'exécution et d'autre part par la meilleure stabilité physique du fluide. Ce gain de stabilité, dû au respect d'une divergence nulle, réduit les instabilités causées par les vibrations des particules dans un fluide au repos (voir figure 5.1).

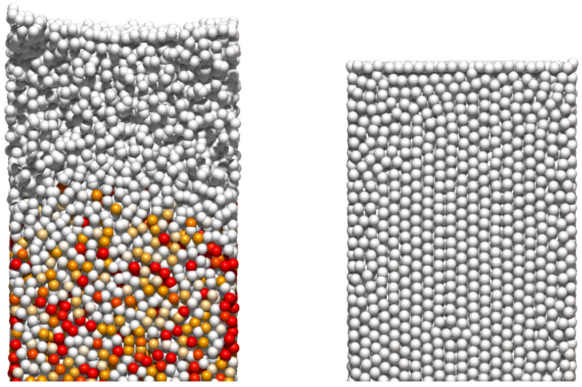


**Figure 5.1.:** Colonne de fluide simulant 80k particules. L'erreur sur la divergence est représentée par la couleur : rouge représente le maximum et blanc le minimum. Les erreurs sur la divergence présentes dans une simulation *Implicit Incompressible SPH* (IISPH) (gauche) provoquent l'apparition d'artefacts au sein du fluide dont l'accumulation provoque l'apparition de larges perturbations au niveau de la surface. Le maintien d'un champ de vitesse avec une divergence nulle par le DFSPH (droite) permet d'empêcher l'apparition de ces artefacts [BK15].

## 5.1 Algorithme DFSPH

Comme nous l'avons présenté lors de la vue d'ensemble des différents algorithmes de simulation de fluide lagrangiens, l'innovation de l'algorithme DFSPH est d'assurer que la divergence du champ de vitesse soit nulle.

### 5.1.1 Généralités

L'algorithme 3 présente l'organisation d'un pas de temps de simulation de l'algorithme DFSPH. Les processus permettant d'assurer une densité constante et une divergence nulle sont illustrés respectivement dans les algorithmes 4 et 5. Ces processus de correction sont mis en place par l'implémentation de boucles prédictives correctives, qui vont évaluer l'erreur présente dans le fluide puis modifier la vitesse des particules de manière à compenser cette erreur. L'erreur évaluée par ces deux processus est respectivement l'erreur entre la densité moyenne et la densité cible et la quantité de divergence présente dans le fluide. Bender et Koschier



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

proposent une variante des formules utilisées un modèle lagrangien pour calculer la variation de vitesse dans à appliquer pour corriger l'erreur détectée permettant d'extraire une partie commune aux deux processus de correction. Cette partie commune $\alpha$ ne dépend que de la position des particules ce qui permet de la calculer une seule fois au début de chacun des pas de temps de simulation :

$$\alpha_i = \frac{1}{\|\sum m_j \nabla W_{ij}\|^2 + \sum \|m_j \nabla W_{ij}\|^2} \tag{5.1}$$

avec $i$ l'indice de la particule concernée, $j$ les indices des particules se trouvant dans le voisinage de la particule $i$, $m$ la masse d'une particule et $\nabla W_{ij}$ le gradiant de la fonction de noyau utilisée pour la distance séparant les deux particules. L'expression de la correction de la vitesse pour la densité et pour la divergence (resp. $\Delta v_i$ et $\Delta v_i^\nu$) peuvent ainsi être exprimés comme ceci :

$$\begin{aligned} \Delta v_i &= -\frac{1}{\Delta t} \sum_j m_j((\rho_i^* - \rho_0)\alpha_i + (\rho_j^* - \rho_0)\alpha_j)\nabla W_{ij} \\ \Delta v_i^\nu &= -\sum_j m_j(\frac{D\rho_i}{Dt}\alpha_i + \frac{D\rho_j}{Dt}\alpha_j)\nabla W_{ij} \end{aligned} \tag{5.2}$$

avec $\rho_0$ la densité du fluide au repos, $\rho_i^*$ l'estimation de la densité future, $\alpha_i$ le facteur calculé par l'équation 5.1, $\frac{D\rho_i}{Dt}$ l'estimation du taux de variation de la densité. La valeur de la densité $\rho_i$ dépend uniquement de la position des particules se trouvant dans le voisinage et peut donc être déterminée en même temps que $\alpha$ en début de chaque pas de simulation. Les différentes variables définies ci-dessus sont calculées selon les formules suivantes :

$$\begin{aligned} \rho_i^* &= \rho_i + \frac{D\rho_i}{Dt}\Delta t \\ \frac{D\rho_i}{Dt} &= m_j(v_i - v_j)\nabla W_{ij} \\ \rho_i &= \sum m_j W_{ij} \end{aligned} \tag{5.3}$$

**Algorithm 3** DFSPH

**for all** particles i **do**
    find Neighborhood $N_i$
**for all** particles i **do**
    compute densities $\rho_i$
    compute factors $\alpha_i$
correctDivergenceError($\alpha$, v)
**for all** particles i **do**
    compute non-pressure forces acceleration $acc_i$
**for all** particles i **do**
    $v_i \mathrel{+}= \Delta t * acc_i$
correctDensityError($\alpha$, v)
**for all** particles i **do**
    $x_i \mathrel{+}= \Delta t * v_i$


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

**Algorithm 4** Constant density solver

1: **function** CORRECTDENSITYERROR($\alpha$, v)
2: **for all** particles i **do**
3: predict density $\rho_i^*$
4: **while** $(\rho_{avg}^* - \rho_0 > \eta) \vee (iter < 2)$ **do**
5: **for all** particles i **do**
6: correct velocity $v_i \mathrel{+}= \Delta v_i$
7: **for all** particles i **do**
8: predict density $\rho_i^*$
9: compute $\rho_{avg}^*$

**Algorithm 5** Divergence-free solver

1: **function** CORRECTDIVERGENCEERROR($\alpha$, v)
2: **for all** particles i **do**
3: predict density variation rate $\frac{D\rho_i}{Dt}$
4: **while** $((\frac{D\rho}{Dt})_{avg} > \eta^v) \vee (iter < 1)$ **do**
5: **for all** particles i **do**
6: correct velocity $v_i \mathrel{+}= \Delta v_i^v$
7: **for all** particles i **do**
8: predict density variation rate $\frac{D\rho_i}{Dt}$
9: compute $(\frac{D\rho}{Dt})_{avg}$

### 5.1.2 Framework SPlisHSPlasH

Une implémentation parallèle sur CPU de l'algorithme DFSPH est proposée dans la librairie SPlisHSPlasH [Ben]. Cette implémentation contient deux modifications de l'algorithme mentionnées dans la publication de Bender et Kochier [BK15].

La première a pour but de limiter les corrections de la vitesse aux particules localisées où le fluide est en surcompression ou bien aux particules où le fluide est en compression. Cette technique est utilisée dans de nombreuses publications du domaine et permet d'éviter les erreurs d'estimation de la densité liées à un voisinage contenant un nombre de particules trop faible, en particulier au niveau de la surface du fluide. En pratique, si la densité future d'une particule estimée par le solveur de densité constante est inférieure à la densité du fluide au repos, alors la correction de la vitesse n'est pas appliquée pour la particule en question. De même, si $\frac{D\rho_i}{Dt} \leqslant 0$ dans le solveur de divergence nulle, alors la vitesse de la particule n'est pas modifiée. Également, la valeur de $\frac{D\rho_i}{Dt}$ est directement fixée à $0$ si le voisinage de la particule contient moins de 20 particules. Cette technique est souvent utilisée pour éviter d'appliquer une correction incorrecte au niveau de la surface de fluide car le voisinage des particules au niveau de la surface n'est pas complet.

La seconde modification vise à améliorer la vitesse de convergence des solveurs de densité constante et de divergence nulle en ajoutant un démarrage à chaud. Ce démarrage à chaud consiste à appliquer une fois les équations 5.2 en remplaçant l'erreur sur la densité, le taux de variation de la densité et le facteur $\alpha$ par les valeurs utilisées lors du pas de temps précédent. En pratique, on remplace $(\rho_i^* - \rho_0)\alpha_i$ et $\frac{D\rho_i}{Dt}\alpha_i$ par deux variables dont la valeur correspond à la somme de leurs valeurs observées au cours des itérations des solveurs (ligne 4 des algorithmes 4 et 5) lors du pas de simulation précédent. On notera que les auteurs ont décidé de faire un démarrage à chaud pour la correction de la divergence seulement si le fluide est en compression. Par conséquent ils estiment $\frac{D\rho_i}{Dt}$ et ne corrigent la vitesse que si cette valeur est positive, bien que cette valeur n'apparaisse pas dans la formule calculant la modification de la vitesse lors du démarrage à chaud. Le démarrage à chaud ajoute donc trois itérations sur toutes les particules : une pour calculer $\frac{D\rho_i}{Dt}$, une pour appliquer la correction de la vitesse pour le solveur de divergence nulle ainsi qu'une pour la correction de la vitesse dans le solveur de densité constante. Son impact sur la vitesse de convergence de chacun des solveurs est étudié dans la section 5.5.1.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

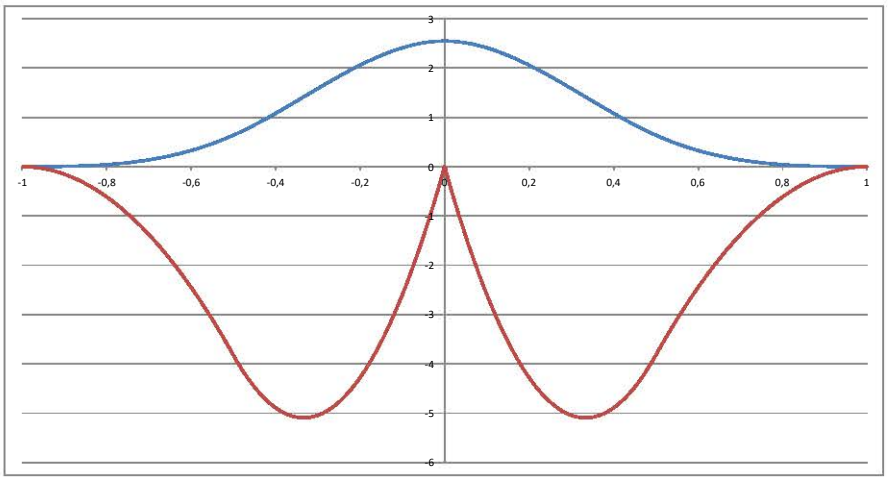



**Figure 5.2.:** Fonction noyau (*cubic spline*) utilisée dans la librairie SPlisHSPlasH [Ben] pour l'algorithme DFSPH. Cette représentation considère un rayon de 1 mètre pour le noyau. En bleu : la fonction noyau ; en rouge : la dérivée.

**Fonction noyau** Dans la librairie SPlisHSPlasH la fonction noyau utilisée est la fonction *cubic spline* présentée dans notre état de l'art :

$$f(q) = \frac{8}{\pi} \begin{cases} 1 - 6q^2 + 6q^3, & 0 \leq q < 0.5 \\ 2(1-q)^3, & 0.5 \leq q < 1 \\ 0, & q \geq 1 \end{cases} \tag{5.4}$$

La dérivée de cette fonction noyau est :

$$\nabla f(q) = \frac{48}{\pi} \begin{cases} \frac{q}{h}(3q - 2), & 0 \leq q < 0.5 \\ \frac{-1}{h}(1-q)^2, & 0.5 \leq q < 1 \\ 0, & q \geq 1 \end{cases} \tag{5.5}$$

Rappelons que $q = \|r\| / h$, $r$ étant la distance entre deux particules et $h$ le rayon de la fonction noyau. Rappelons également que la fonction noyau $W$ (resp. $\nabla W$) est de la forme $W = \frac{1}{h^d} * f(q)$ (resp. $\nabla W = \frac{1}{h^d} * \nabla f(q)$) avec $d$ le nombre de dimensions de la simulation (donc 3 pour des simulations en 3D). La figure 5.2 illustre la fonction noyau pour un rayon de 1 mètre. Notons que, contrairement à ce qui était représenté dans la figure 3.2, la dérivée est bien négative car $q \geq 0$.

## 5.2 Structure de données

Pour réaliser notre simulation de fluide nous distinguons deux types de particules : les particules représentant le fluide et les particules représentant les solides. Ces deux types de particules possèdent un nombre de données en commun (ex : une position). La raison de la séparation en deux types est que lorsqu'une particule de fluide interagit avec un solide la valeur de pression utilisée au niveau du solide est égale à celle de la particule de fluide. Par conséquent, il n'est pas nécessaire d'allouer les données utilisées pour le calcul des propriétés pour les particules représentant un solide. Les particules solides requièrent cependant des


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

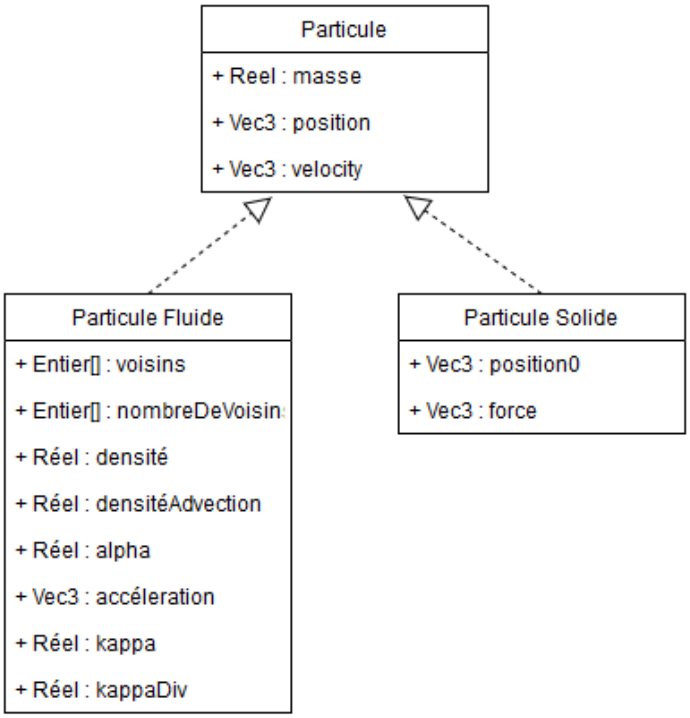



**Figure 5.3.:** Données stockées pour chaque type de particules (fluide ou solide). La classe Particule représente les données communes aux deux types de particules.

données additionnelles, notamment la position relative des particules au solide pour permettre de déterminer la position des particules une fois la nouvelle position du solide calculée par le moteur physique. La figure 5.3 représente les données utilisées pour chaque type de particules. Notons qu'à l'exception de notre gestion du voisinage, les données utilisées sont identiques à celles utilisées dans la librairie SPlisHSPlasH [Ben].

Les données communes aux deux types de particules sont : masse, position et vitesse. Les particules composant les solides ne requièrent que peu de données supplémentaires : la position relative au solide et une force. Cette force correspond à la somme des forces appliquées par les particules de fluide adjacentes. Elle est transmise au moteur physique pour appliquer l'influence du fluide sur les solides. Les particules de fluide nécessitent plus d'informations. Nous pouvons regrouper les données additionnelles selon leur utilisation :

— **calculs de propriétés :** les variables *densité* et *alpha* sont utilisées pour stocker respectivement la densité du fluide et la valeur correspondant à la partie commune des deux processus itératifs-correctifs de l'algorithme DFSPH [BK15]. *densitéAdv* est la variation de la densité permettant de déterminer si le fluide a atteint la densité requise à la fin de chaque itération des processus itératifs-correctifs.

— **accélération :** la variable *accélération* est utilisée pour stocker l'impact des forces externes (ex : viscosité).

— **démarrage à chaud :** les variables *kappa* et *kappaDiv* sont utilisées pour réaliser un démarrage à chaud des processus itératifs-correctifs comme proposé par Bender et Koschier [BK15].



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

— **gestion du voisinage :** les variables *voisins* et *nombreDeVoisins* sont utilisées pour stocker les identifiants des particules se trouvant dans le voisinage de la particule comme présenté dans la section 5.2.1.

### 5.2.1 Gestion du voisinage

Pour réaliser la gestion du voisinage de chaque particule deux approches sont possibles suivant si l'on décide de garder en mémoire ou non pour chaque particule les identifiants des particules se trouvant dans son voisinage. Conserver les identifiants en mémoire permet d'éviter d'avoir à explorer la structure de données utilisée pour l'accélération de la recherche de voisins à chaque nouveau calcul de propriété. Cependant, l'espace mémoire nécessaire pour conserver ces identifiants peut devenir très élevé. Lors de nos tests nous avons observé jusqu'à 68 particules dans un voisinage. Si l'on prend un nombre de 75 pour assurer qu'il n'y ait jamais de dépassement, et en supposant l'utilisation d'entiers de 32 bits, cela veut dire que nous devons ajouter 300 octets de données supplémentaires par particule. Ceci peut poser des problèmes de mémoire pour des simulations comportant un très grand nombre de particules. La carte graphique que nous utilisons pour nos tests dispose de 8192 Mo de mémoire ce qui nous permet de stocker ces informations en mémoire tant que nos simulations ne nécessitent pas des millions de particules. Par conséquent, nous avons choisi de stocker les identifiants des particules voisines.

Nos particules sont distribuées entre plusieurs structures selon si elles font partie du fluide ou d'un corps rigide. Nous devons donc utiliser un identifiant global permettant de déterminer si la particule correspond à un corps rigide et, auquel cas, à quel corps rigide elle appartient. Pour pouvoir déterminer si la particule appartient au fluide lors de l'exploration des particules voisines lors de calculs des propriétés physiques, nous avons décidé de remplir le tableau de particules voisines en premier avec les particules de fluide puis d'y ajouter les particules de solides (figure 5.4). En plus du tableau nous devons donc stocker deux nombres : le nombre de particules de fluides dans le voisinage et le nombre de particules de solides. Pour les particules faisant partie d'un solide, nous pouvons utiliser à notre avantage qu'aucun solide ne nécessite l'utilisation des 32 bits disponibles dans un entier pour identifier une particule car les solides ne sont généralement pas composés d'un grand nombre de particules. Nous pouvons donc réserver une partie de l'entier utilisé pour identifier la particule pour sauvegarder l'identifiant du corps rigide associé. Nous proposons deux méthodes de codage de l'identifiant :

— **Opération bit à bit :** Pour cette solution nous utilisons un décalage à gauche pour placer l'identifiant de la particule dans la partie supérieure de l'entier avant de l'ajouter à l'identifiant du corps rigide. Pour nos simulations nous réservons 8 bits pour l'identifiant du corps rigide ce qui nous laisse 24 bits disponibles pour celui des particules. Pour extraire les deux identifiants nous utilisons des masques nous permettant de conserver uniquement la partie désirée.

— **Multiplication :** Pour obtenir l'identifiant global nous multiplions l'identifiant du corps rigide par une constante $K$ supérieure au nombre maximal de particules pour un solide (ex : $10^6$) avant de lui ajouter l'identifiant de la particule. L'identifiant de la particule et celui du corps solide peuvent être récupérés en conservant respectivement le reste et le quotient de la division euclidienne de l'identifiant global.

Vu que nous n'avons pas remarqué de différence significative entre les temps d'exécution obtenus avec chacune des deux approches, nous avons décidé d'utiliser l'identifiant utilisant les opérations bit à bit dans l'implémentation finale car elles permettent un accès théoriquement plus rapide à l'identifiant du corps rigide.


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

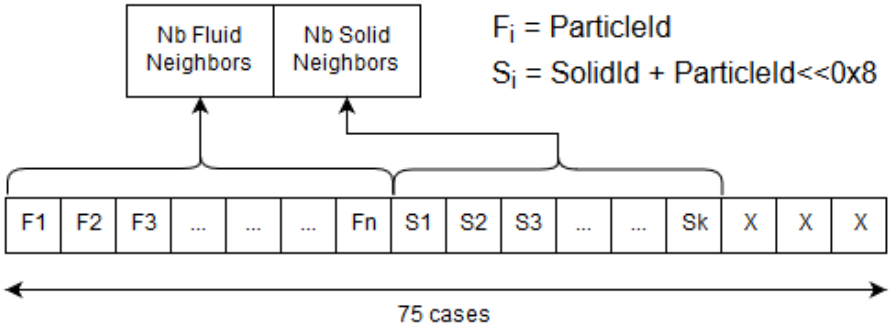



**Figure 5.4.:** Représentation des données stockées pour accéder aux particules se trouvant dans le voisinage d'une particule.

Pour remplir le tableau de particules voisines nous utilisons une structure classique de grille uniforme [Ihm+11]. La taille des cellules de la grille est calculée en fonction de la taille des particules. Le but est que la totalité des voisins d'une particule se trouvent soit dans la même cellule que la particule étudiée soit dans l'une des 26 cellules adjacentes. Le voisinage d'une particule correspond à toutes les particules se trouvant à une distance inférieure au rayon de la fonction noyau. Par conséquent, la taille optimale des cellules correspond exactement au rayon de la fonction noyau, c.à.d. 4 fois le rayon des particules dans la majeure partie des publications du domaine. Cette grille uniforme est également utilisée pour trier régulièrement les particules en fonction de leur position dans la grille de manière à maintenir un haut niveau de cohérence spatiale. Les particules composant les solides sont triées une seule fois lors de leur création. Lors du tri des particules seules les informations conservées entre chaque pas de temps de simulation sont réorganisées :

— **particules de fluide :** masse, position, vitesse et les données pour le démarrage à chaud ($kappa$ et $kappaDiv$).

— **particules de solide :** masse et position.

La construction de la structure utilisant une grille uniforme nécessite trois étapes :

— **Attribution de l'identifiant :** la première étape consiste à attribuer à chaque particule un identifiant correspondant à la cellule de la grille dans laquelle se trouve la particule.

— **Tri des particules :** les indices des particules sont ensuite triés en fonction de l'identifiant précédemment attribué. S'il est nécessaire de trier les propriétés des particules, pour améliorer la cohérence spatiale, elles sont également triées lors de cette étape. Dans le cas contraire, pour limiter les calculs effectués, seul l'indice des particules est trié. À la fin de cette étape, on obtient un nouveau tableau *Particules idx triés* présentant les indices des particules ordonnés suivant leur identifiant de cellule et non plus leur indice.

— **Recherche du début de chaque cellule :** Le but de cette étape est de rechercher la case du tableau *Particules idx triés* correspondant à la première particule de chaque cellule de la grille. En pratique cela revient à faire un histogramme cumulatif *Cellule début-fin* sur le tableau associant chaque particule à son identifiant de cellule.

La figure 5.5 illustre la structure de données ainsi produite. La partie basse de l'image illustre les données après l'étape d'attribution des identifiants. Le reste de l'image correspond aux données obtenues à la fin des trois étapes : le tableau *Particules idx triés* représentant les



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

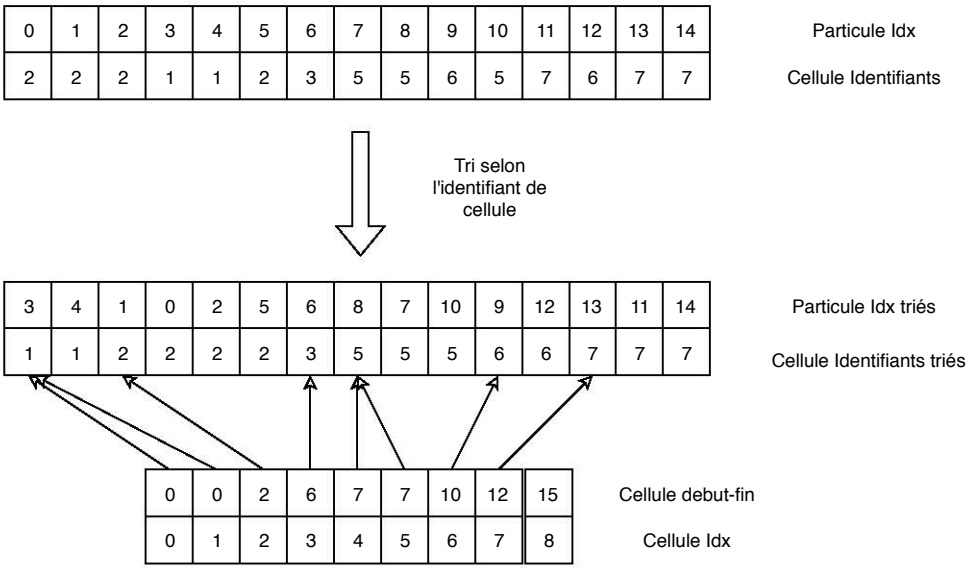



**Figure 5.5.:** Structure de données représentant la grille uniforme utilisée. Une neuvième cellule est ajoutée à la grille pour indiquer le nombre total de particules et éviter de créer un cas particulier pour la dernière cellule de la grille

indices des particules triées en fonction de leur identifiant visible au milieu de l'image et l'histogramme *Cellule début-fin* contenant la position dans le tableau *Particules idx triés* de la première particule de chaque cellule de la grille uniforme visible en haut de l'image. Au début de chaque nouveau pas de simulation, il est nécessaire de répéter ces trois étapes pour mettre à jour la structure de données.

Comme nous le verrons dans la section 5.2.2 les données sont organisées selon une structure de tableau, c.à.d. que chaque ensemble de particules (un pour le fluide et un pour chaque solide) contient un tableau de particules. Ces ensembles de particules étant distincts, il semble donc nécessaire d'ajouter une structure de grille uniforme à chaque ensemble de particules et d'effectuer les trois étapes de mise à jour de la structure de données pour chaque ensemble de particules. Cependant, il est possible de limiter les calculs effectués en utilisant les deux considérations suivantes :

— **Les objets immobiles sont toujours à jour.** En effet, si un objet est immobile entre deux pas de simulation, les particules n'ont pas été déplacées et la structure de grille uniforme est donc constante. Par conséquent, nous différencions les ensembles de particules correspondant à des objets immobiles et nous n'effectuons les calculs nécessaires à la construction de la grille uniforme qu'une seule fois en début de simulation.

— **Construire une structure de grille uniforme commune à toutes les particules requiert moins de calculs.** Construire une structure commune à toutes les particules possède deux avantages. Le premier est que chacune des trois étapes nécessaires à la construction peut être hautement parallélisée. Par conséquent, si l'un des ensembles de particules possède un nombre faible de particules, le taux d'occupation du matériel disponible sera faible (en particulier pour une implémentation sur GPU). Utiliser une structure commune à toutes les particules permet donc une diminution du temps de calcul nécessaire. Le second avantage est une empreinte mémoire plus petite avec une meilleure cohésion spatiale. Considérons le cas où chaque ensemble de particules a sa


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

propre structure de grille uniforme. Pour chaque ensemble, nous aurons un histogramme indiquant l'indice de la première particule de chaque cellule de la grille. Lorsque l'on cherche à connaître toutes les particules contenues dans une cellule il est nécessaire d'explorer tous les histogrammes. Chaque histogramme étant alloué séparément, les zones mémoires accédées ne seront donc pas adjacentes. Utiliser une seule structure de grille globale permet d'avoir un seul histogramme. Cela permet non seulement de diminuer l'espace mémoire requit, mais également d'avoir les indices des particules contenues dans une cellule adjacente en mémoire ce qui permet une exploration plus rapide de notre structure de grille.

En prenant en compte ces deux considérations nous obtenons un système comprenant deux structures de grille uniforme. La première représente le positionnement des particules associées à des objets immobiles et les trois étapes d'initialisation de cette structure ne sont exécutés qu'une seule fois en début de simulation. La seconde représente le positionnement de toutes les particules dynamiques (fluide et solides dynamiques) et est réinitialisée au début de chaque pas de temps de simulation.

Ce système possède une limitation au niveau de la structure représentant l'organisation des particules dynamiques : elle ne produit pas un indice permettant de trier les particules de fluide. Nous verrons lors de nos études expérimentales (section 5.5.1) que trier régulièrement les particules de fluide permet un gain de performances très important car cela améliore la cohérence spatiale des données. Or, vu que toutes les particules dynamiques sont regroupées dans une seule structure de grille uniforme, l'indice trié $Idx_{sorted}$ produit lors de la deuxième étape de l'initialisation de la structure ne permet plus de trier les particules de fluide car elles sont mélangées avec les particules de solides dynamiques. Par conséquent, pour les pas de simulation où les particules de fluide doivent être triées, nous utilisons une structure de grille uniforme supplémentaire spécifique aux particules de fluide.

La figure 5.6 récapitule le processus de recherche du voisinage et la structure de donnée est visualisée par la classe *NeighborsSearchDataSet* dans la figure 5.7.

### 5.2.2 Structure de tableaux

Comme nous l'avions expliqué dans la section 3.3 de notre état de l'art, il existe deux possibilités pour l'organisation globale des données de notre simulateur de fluide : une structure de tableaux (SoA) ou un tableau de structure (AoS). Nous avions vu qu'une une structure de tableaux (SoA) assure une meilleure cohérence spatiale des données, améliorant ainsi la vitesse d'accès aux données. Nous avons donc utilisé cette approche pour notre structure globale de données. Nous utilisons une structure spécifique aux particules de fluide ainsi qu'une structure pour chaque solide (figure 5.7).

Nous pouvons réaliser une estimation de l'espace mémoire nécessaire à cette structure de données. Pour réaliser cette estimation nous allons supposer des entiers sur 4 octets et des réels en simple précision, c.à.d. également sur 4 octets. Toutes les équations qui suivent utilisent toutes la variable $nbParticules$, cependant cette variable indique le nombre de particules considérées par la structure associée à la formule en question et, par conséquent, la variable ne représente pas la même quantité pour toutes les équations.

La classe *NeighborsSearchDataSet* correspond à la structure de données pour la grille uniforme utilisée pour réaliser la recherche de voisins. La taille $T_{NSDS}$ de cette structure de données dépend non seulement du nombre de particules qu'elle considère mais également



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

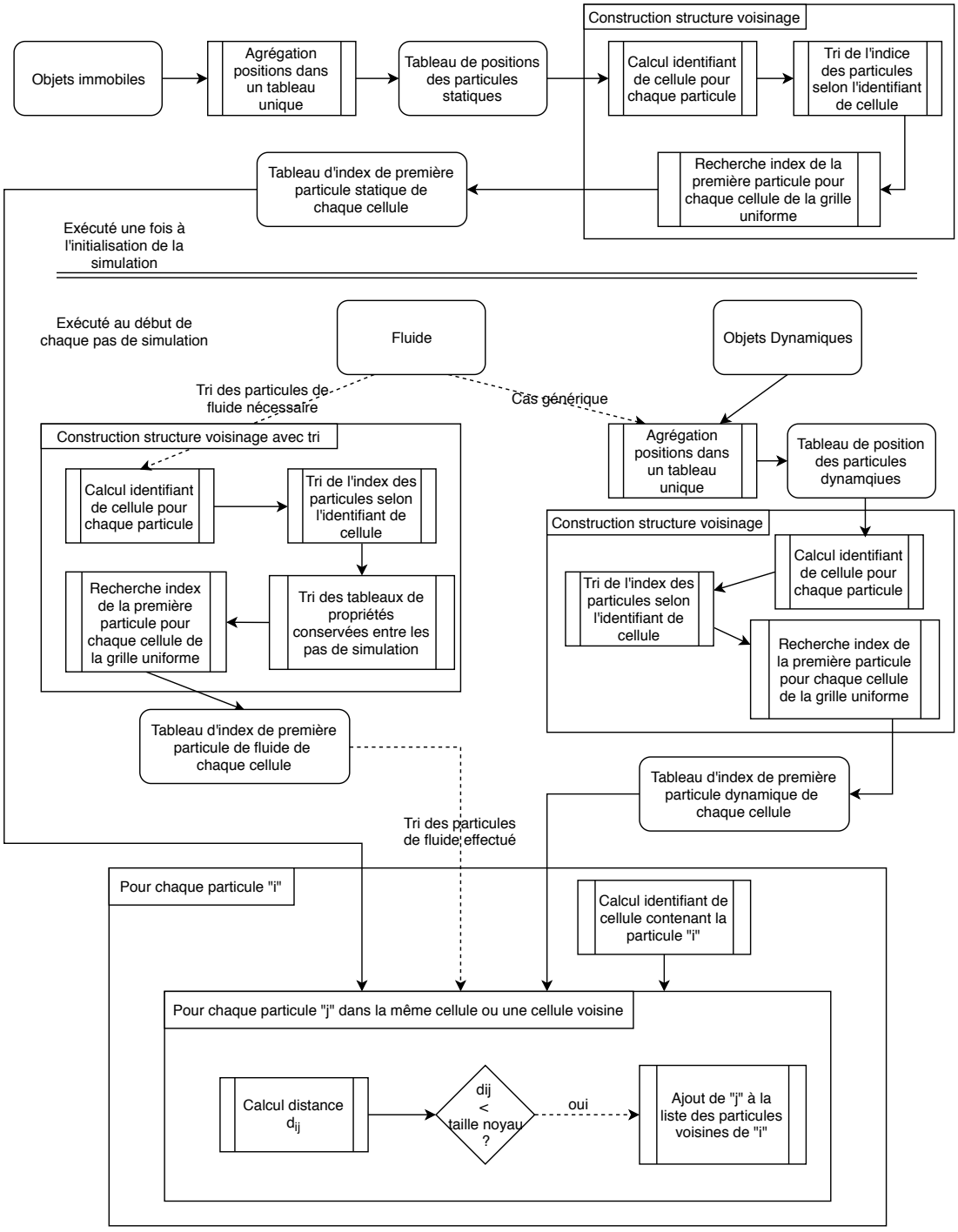



**Figure 5.6.:** Vue d'ensemble du processus de recherche des particules voisines. Les rectangles arrondis représentent les données et les boites avec doubles bordures les étapes de calcul. Notre système utilise principalement deux structures de grille uniforme permettant ainsi d'initialiser la structure relative aux particules immobiles qu'une seule fois lors de l'initialisation de la simulation. Une structure additionnelle spécifique aux particules de fluide peut être requise pour les pas de temps où le tri des données des particules de fluide est requis.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

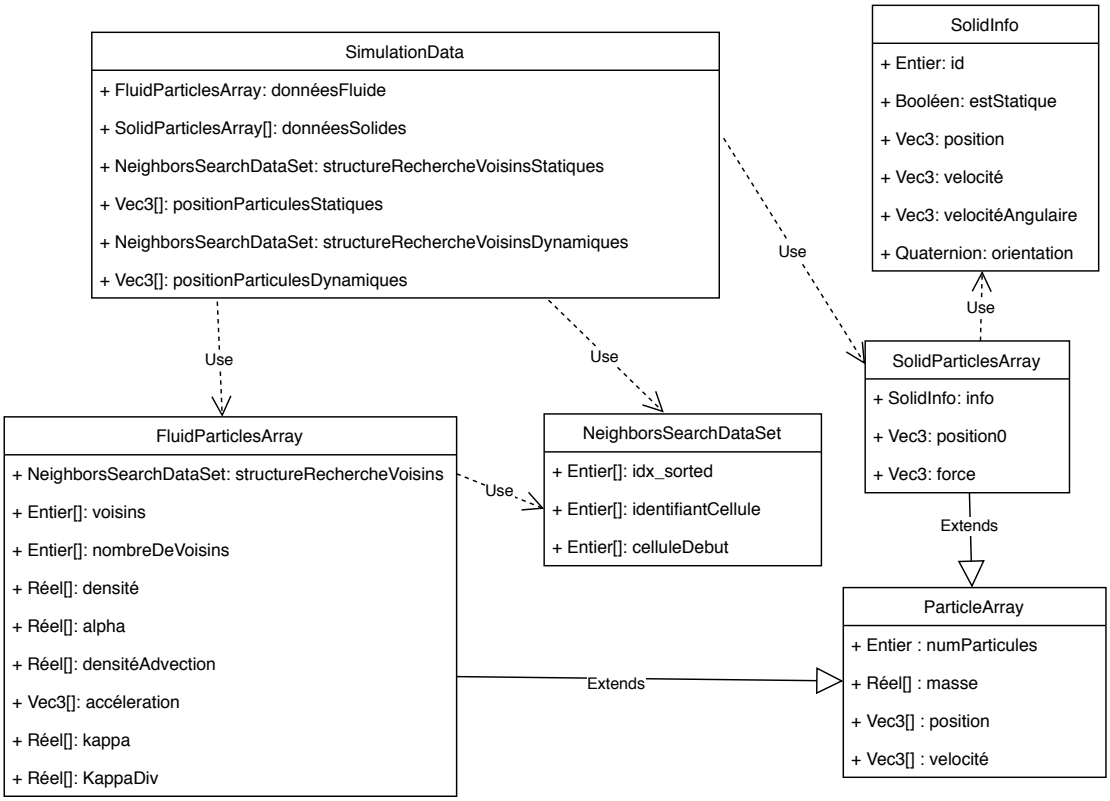



**Figure 5.7.:** Structures de données utilisées par notre simulateur de fluide.

du nombre de cellules de la grille uniforme $T_{NSDS} = 8 * nbParticules + 4 * nbCellules$. La mémoire $T_{PA}$ requise pour la partie commune aux particules de solide et de fluide *ParticleArray* ne dépend que du nombre de particules concernées $T_{PA} = 4 + 28 * nbParticules$. La structure contenant les particules représentant un solide *SolidParticleArray* nécessite les $56$ octets de la structure indiquant l'état du corps rigide *SolidInfo* en plus des informations concernant les particules ce qui nous donne une empreinte mémoire définie par la formule suivante $T_{SPA} = 56 + T_{PA} + 24 * nbParticules = 56 + 52 * nbParticules$. La structure de données répertoriant l'ensemble des particules de fluide nécessite une structure de recherche de voisins pour les pas de temps où un tri des particules est nécessaire en plus des informations sur les particules de fluide elles-mêmes, ce qui nous donne la formule suivante $T_{FPA} = T_{NSDS} + T_{PA} + 340 * nbParticules = 4 + 378 * nbParticules + 4 * nbCellules$. Une large partie, 300 octets par particule, de cette taille est due au stockage des indices des particules se trouvant dans le voisinage dans la variable $voisins$. Les deux structures de recherche de voisins ainsi que les tableaux de positions présents dans la classe *SimulationData* sont utilisés pour le système d'agrégation de recherche de voisins présenté dans la section précédente. Leur taille dépend du nombre de particules immobiles $nbParticulesimmobile$ et du nombre de particules dynamiques (particules de fluide comprises) $nbParticulesDynamiques$. Notons que la somme de ces valeurs correspond au nombre total de particules dans la simulation $nbParticulesTotal = nbParticulesimmobile + nbParticulesDynamiques = nbParticulesFluide + \sum_i nbParticulesSolide_i$.

Au final, notre structure de données a l'empreinte mémoire suivante :



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

$$T_{SD} = T_{FPA} + \sum_{i}^{nbCorpsRigides} T_{SPA\_i} + T_{NSDS}(nbParticulesimmobiles) + T_{NSDS}(nbParticulesDynamiques) + 12 * nbParticulesTotal$$

$$T_{SD} = 398 * nbParticulesFluide + 72 * nbParticulesSolides + 12 * nbCellules + 4 * nbCorpsRigides + 56$$

Pour illustrer l'ordre de grandeur de la mémoire utilisée, nous pouvons calculer l'empreinte mémoire de la simulation de fluide au repos utilisée pour une grande partie de nos expérimentations. Cette simulation est composée de 78000 particules de fluide et 54800 particules de solides représentant les bordures (immobile). Les bordures décrivent une boite rectangulaire de 2 mètres de coté et 4 mètres de haut. Pour des soucis d'alignement des données la grille uniforme doit avoir un nombre de cases correspondant à un multiple de deux pour chaque côté et la grille uniforme est de forme cubique pour simplifier son utilisation. Les cellules, ont une taille de 2 fois le diamètre des particules, donc de 0.1 mètre pour des particules de 5 cm de diamètre. Par conséquent, il faut au minimum 40 cellules pour la hauteur et nous avons fixé la taille des côtés de la grille uniforme à 64 cellules, soit un total de 262144 cellules. L'empreinte mémoire de cette simulation est d'environ 38 Mo. Cette structure de données permet de réaliser des simulations contenant plus de 10 millions de particules sans atteindre la limite d'environ 8Go, de la carte graphique utilisée dans nos test.

## 5.3 DFSPH sur GPU

### 5.3.1 Implémentation parallèle

Une simulation lagrangienne d'un fluide utilise couramment un grand nombre de particules pour assurer un réalisme physique correct. Par conséquent, l'approche généralement utilisée pour proposer une implémentation parallèle GPU d'un tel fluide est d'attribuer chaque particule à un thread. Cette approche permet de maximiser l'utilisation du nombre de threads disponibles très élevé sur carte graphique. Il est cependant important de noter que les algorithmes de simulation de fluide lagrangienne ne sont parallélisables que par partie. Par exemple, il est possible de déterminer la valeur de la pression pour chaque particule en parallèle. Cependant, la pression au niveau d'une particule dépend de la valeur de la densité du fluide au niveau des particules présentes dans son voisinage. Par conséquent, nous devons attendre d'avoir déterminé la valeur de la densité pour toutes les particules avant de pouvoir commencer à calculer la pression. Dans une implémentation sur carte graphique cela va résulter en une succession de kernel GPU. Une question importante est de se demander si attribuer un kernel GPU à chaque propriété et exécuter plusieurs kernels en parallèle quand les propriétés concernées ne dépendent pas l'une de l'autre est plus intéressant que d'écrire un kernel unique aux deux propriétés. La seconde solution a l'intérêt de permettre de charger les informations nécessaires communes aux deux propriétés dans le cache au début du kernel plutôt que d'avoir à les recharger pour chaque propriété depuis la mémoire globale, beaucoup plus lente sur carte graphique. L'algorithme DFSPH (algorithme 3 page 78) ne nécessite pas de modification importante car les parties constantes des formules ont déjà été extraites et sont calculées en début de pas de simulation. Nous utilisons un kernel CUDA à chaque fois que l'algorithme requiert d'itérer sur toutes les particules. Bien que la recherche


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

du voisinage soit visualisée en une seule étape dans l'algorithme 3, nous avons vu dans la section 5.2.1 qu'elle est composé de quatre étapes séparées dans le cas de la construction d'une grille uniforme : calcul de l'identifiant de la cellule pour chaque particule, tri de l'indice des particules en fonction de leur cellule, recherche de la première particule pour chaque cellule et exploration des cellules voisines pour chaque particule pour sauvegarder les indices des particules voisines. Également, nous utilisons le processus de démarrage à chaud proposé par Bender et Kochier (section 5.1.2), ce qui nécessite trois kernels additionnels. Ceci mène à un total de 15 kernels différents, les kernels des lignes 3 et 8 des solveurs prédictifs-correctifs (algorithmes 4 et 5 page 79) étant le même kernel. Nous pouvons cependant remarquer que l'estimation de $\frac{D\rho_i}{Dt}$ est indépendante des valeurs de $\rho$ et $\alpha$. Par conséquent nous pouvons calculer $\rho$ et $\alpha$ et la première estimation de $\frac{D\rho_i}{Dt}$ pour le solveur de divergence nulle dans un unique kernel. Au final, nous avons donc 14 kernels différents. La figure 5.8 donne une visualisation du processus (kernels et organisation) exécuté au cours d'un pas de temps de simulation.

### 5.3.2 Optimisation

Bien que les publications proposant une implémentation GPU d'un algorithme lagrangien soient rares de très nombreuses publications proposent des implémentations parallèles CPU. Les auteurs proposent généralement des optimisations ou des modifications génériques à n'importe quel algorithme lagrangien ayant pour but de diminuer le temps de calcul nécessaire pour chaque pas de simulation. Dans cette section, nous nous proposons d'étudier certaines de ces modifications pour déterminer si elles permettent d'obtenir un gain de performance notable dans une implémentation sur carte graphique. Les résultats et conclusions de ces études sont présentés dans la section 5.5.1.

**Cohérence spatiale des données** Comme nous l'avons présenté dans la section 3.3 de notre état de l'art, il existe deux indices couramment utilisés pour explorer une grille uniforme : l'indice linéaire et l'indice de Morton. L'indice linéaire est l'indice classiquement utilisé dans une grille 3D. L'indice de Morton permet de décrire une exploration de la grille avec une meilleure cohérence spatiale qu'un indice classique (c.à.d où des cases proches ont un indice proche). Ihmsen et al. [Ihm+11] proposent une comparaison de ces deux types d'indices. Ils obtiennent de meilleures performances avec l'indice de Morton dans le cas d'une grille régulière. En particulier leurs travaux font apparaître un facteur d'ordre deux entre les temps de création nécessaire suivant l'indice utilisé. Domingez et al. [DCG11] proposent une optimisation de la création de la structure de données lors de l'utilisation d'un indice classique en se servant du fait qu'avec un tel indice les cellules voisines peuvent être explorées par groupes de trois (figure 3.3c). De plus, le calcul de l'indice de Morton d'une case nécessite des opérations bit à bit ce qui augmente les calculs nécessaires par rapport à un indice classique. La raison principale derrière l'utilisation de l'indice de Morton est sa meilleure cohérence spatiale. Cet indice est donc supposé apporter de meilleures performances lors des calculs des propriétés d'une particule en offrant une meilleure utilisation du cache lors de l'exploration des particules du voisinage. Pour maintenir la cohérence spatiale au cours de la simulation, il est nécessaire de régulièrement trier les particules, c.à.d. les tableaux de position, masse, vitesse ainsi que ceux permettant le démarrage à chaud. Il ne semble pas y avoir de consensus quant à la fréquence de tri optimale. Bender et Koschier [BK15] trient les particules tous les 500 pas de simulation et Ihmsen et al. [Ihm+11] effectuent le tri tous les 100 pas. Pour évaluer l'efficacité des différentes configurations (indice linéaire, indice de Morton ou indice linéaire utilisant les cases consécutives) nous définissons deux expériences. Dans la première, nous mesurons la vitesse de création de la grille uniforme ainsi que le temps nécessaire pour



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

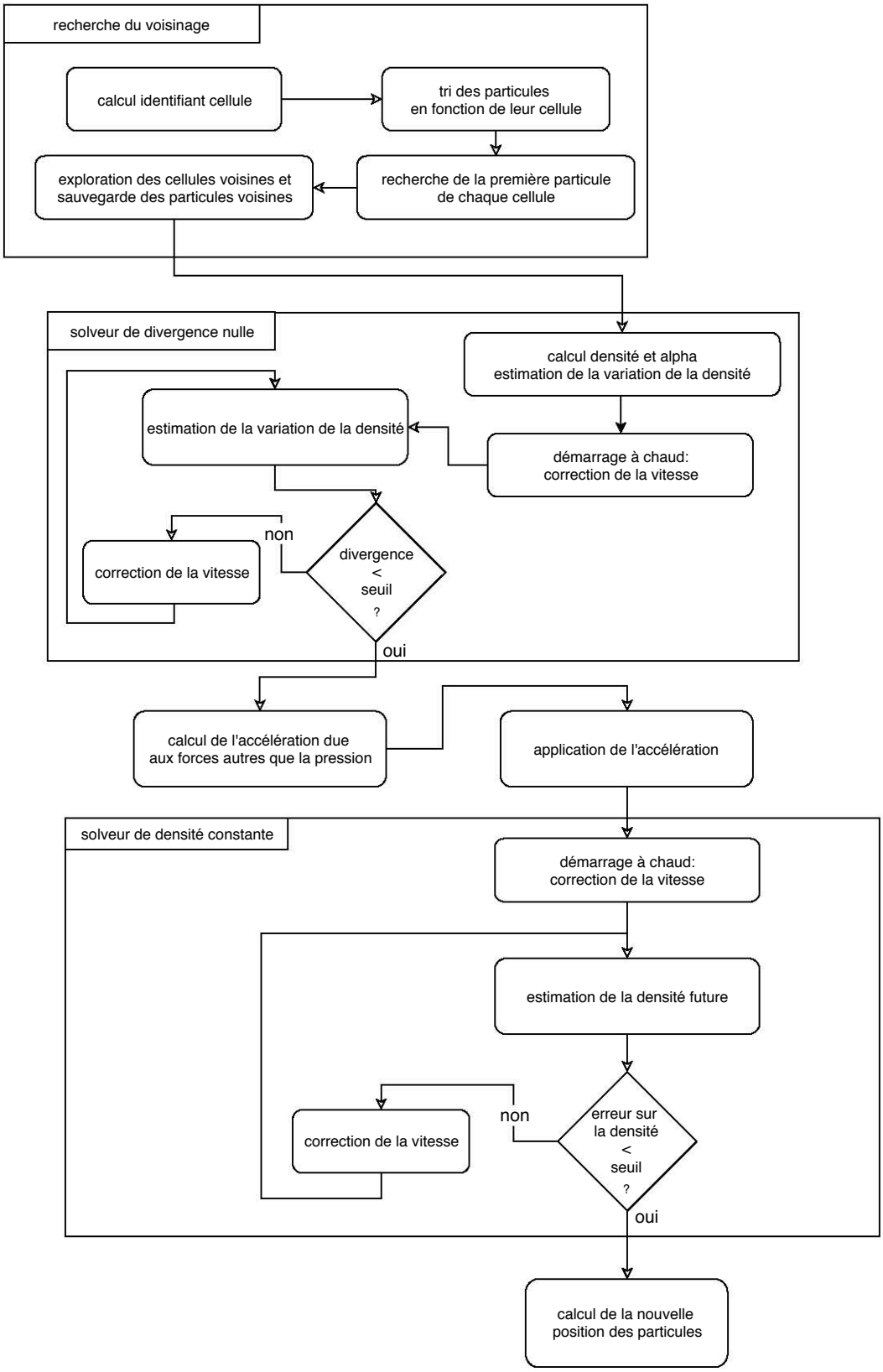



**Figure 5.8.:** Représentation graphique du processus (kernels et organisation) exécuté au cours d'un pas de temps de simulation. Les kernels CUDA sont représentés sous la forme de boites arrondies.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

sauvegarder le voisinage de chaque particule. Cette première expérience est réalisée avec un ordre aléatoire des particules nous permettant ainsi d'éliminer le facteur de cohérence spatiale des données. Dans un second temps, nous mesurons le temps total nécessaire à un pas de temps de la simulation pour chaque indice, classique ou de Morton, en fonction de la fréquence de tri des particules. Ce test nous permettra d'évaluer l'importance de la cohérence spatiale des données pour chacun des deux indices.

**Pré-calcul des fonctions de noyaux** Bender et Koschier [BK15] proposent de créer une table de correspondance permettant de pré-calculer la fonction de noyau et sa dérivée. Pour leurs expériences, ils utilisent 1000 points d'échantillonnage et reportent un gain de performance de $30\%$. Les accès mémoire étant plus coûteux sur GPU que CPU, nous pouvons nous attendre à une diminution de l'efficacité d'un tel système pour une implémentation sur carte graphique. Pour vérifier l'impact de ce système, nous avons décidé d'implémenter une table de correspondance identique à celle proposée par Bender et Koshier et nous comparons les performances à celles obtenues avec un calcul direct des valeurs des fonctions de noyaux.

**Démarrage à chaud** Certaines publications récentes [Ihm+14a ; BK15] utilisent un démarrage à chaud des processus prédictifs-correctifs pour limiter le nombre d'itérations nécessaires à l'obtention de la stabilité désirée. Cependant, les auteurs ne présentent pas en détail l'impact de celui-ci sur les performances obtenues. Également, certains auteurs limitent les valeurs utilisées pour le démarrage à chaud à $50\%$ des valeurs du pas de temps précédent [Ihm+14a] et d'autres utilisent la totalité de la valeur [BK15]. Pour évaluer le démarrage à chaud nous étudions son impact dans le cas de l'algorithme DFSPH.

## 5.4 Fenêtre dynamique de simulation pour fluide lagrangien

Pour toute simulation, le niveau de détail obtenu dépend de la précision des calculs effectués. Dans le cas des simulations de fluide lagrangiennes il existe deux critères principaux affectant le niveau de précision des calculs : le pas de temps de simulation et la taille des particules.

Comme nous l'avons présenté lors de notre étude de l'évolution des algorithmes SPH (section 3.2.2), les algorithmes récents (PCISPH, IISPH, DFSPH) utilisent des processus prédictifs-correctifs pour permettre l'utilisation de pas de temps plus élevés sans causer de perte de précision. Certaines publications récentes utilisent un pas de temps adaptatif, ce qui leur permet de ne pas avoir à restreindre leur simulation à des pas de temps faibles dans le cas où ceux-ci ne sont nécessaires que pour une courte durée au cours de la simulation [BK15]. Le pas de temps maximal est généralement déterminé selon la condition de Courant-Friedrich-Levy (CFL) :

$$\Delta t = 0.4 \frac{d}{\|v_{max}\|} \tag{5.6}$$

Avec $d$ le diamètre des particules de fluide et $v_{max}$ la vitesse maximale observée lors du pas de temps précédent. Cependant si le pas de temps de simulation du fluide est dynamique il est nécessaire que celui de la simulation de solide le soit aussi ce qui est une limitation importante pour notre cas d'application . Comme nous l'avons vu dans l'étude de notre contrôleur pour des fréquences faibles (section 4.4.3), les paramètres de certains composants, dans notre cas les régulateurs PD, varient en fonction de la fréquence de simulation. Il serait complexe



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

de gérer une fréquence de simulation dynamique. Par conséquent, cette approche n'est pas adaptée à nos travaux.

En plus d'influencer la précision des calculs, la taille des particules affecte également la taille mémoire nécessaire à la simulation d'un volume donné. Une simulation avec des particules plus grosses limite les détails obtenus mais permet de simuler un volume de fluide plus important sans nécessiter des temps de calculs prohibitifs. Ceci rend l'utilisation de tailles de particules dynamiques bien plus intéressantes que l'utilisation de pas de temps dynamique. L'objectif habituel des travaux proposant des manipulations dynamiques de la taille des particules est la mise en place de zones de simulations avec une haute précision dans des zones précises d'une simulation moins détaillée [SG11 ; OK12]. Bien que les résultats obtenus par ces publications soient prometteurs, la gestion des transferts de particules entre les deux niveaux de précision reste complexe et introduit un coût non négligeable en temps de calcul.

Bien que l'utilisation de multiples tailles de particules soit intéressant si l'on veut augmenter le niveau de détail dans une zone précise de la simulation, cette méthode n'est adaptée que si l'état du fluide hors de la zone haute précession est également important. Si l'on considère une simulation où l'état du fluide hors de la zone de haute précision n'a que peu d'intérêt alors il serait préférable de seulement simuler la zone d'intérêt avec une haute précision et d'utiliser une méthode ne nécessitant pas de simulation autour de celle-ci, par exemple une méthode procedurale générant des vagues à la surface du fluide. Le scénario type pour ce genre de simulation est la simulation d'un océan avec quelques zones d'intérêt fixes, telles que des rochers. Si la zone d'intérêt n'est pas fixe, par exemple lors du déplacement d'un solide dans le fluide, simplement définir des zones de transition entre le fluide procédural et le fluide simulé ne suffit plus. Un tel scénario nécessite une méthode permettant de déplacer la zone simulée en plus d'une méthode permettant de calculer les conditions aux bordures de la simulation en fonction de l'état du fluide procédural. À notre connaissance, il n'existe pas de travaux proposant de telles méthodes pour des simulations lagrangiennes. Cependant, cette problématique a été étudiée pour des fluides eulériens et hybrides. En particulier, Stomakhin et Selle [SS17] ont proposé une méthode permettant le déplacement de bateaux dans une zone simulée par une méthode hybride au sein d'un océan généré procéduralement. Dans leur introduction, les auteurs explicitent qu'une méthode naïve consisterait à simplement déplacer la boite solide représentant les bordures de la zone simulée pour suivre le navire ce qui force le fluide qu'elle contient à également se déplacer. Cette approche ne permettrait pas d'obtenir le résultat désiré. Le fluide ne doit pas être déplacé, seule la zone simulée doit l'être. Les auteurs continuent en expliquant que la méthode correcte consiste à ajouter du fluide à la nouvelle position de la zone simulée et à retirer celui qui se trouve à l'ancienne position. Cela revient à avoir une source de fluide dans le sens du déplacement et un puits à l'opposé. La source du fluide doit permettre d'initialiser le volume de fluide ajouté à la simulation de façon à ce qu'il corresponde au fluide procédural (vitesse, niveau de liquide). Les concepts utilisés pour leur méthode sont relativement simples. Ils commencent par définir un maillage décrivant les limites de l'espace simulé. Ce maillage est ensuite discrétisé dans la représentation eulérienne ce qui permet de déterminer quelles sont les cellules qui correspondent aux limites de la zone simulée. Ils calculent les propriétés du fluide dans ces cellules en fonction de l'état du fluide procédural ce qui leur permet de définir les conditions aux limites de la zone simulée et ainsi de déterminer les propriétés du fluide (densité, pression, vitesse) simulé au niveau de la bordure. Lors de la transition dans la représentation lagrangienne, utilisée pour gérer le déplacement des quantités de fluide, ils créent des particules non seulement dans les cellules contenant du fluide simulé mais également dans les cellules qui correspondent au maillage utilisé pour définir les limites de l'espace simulé. Ces particules supplémentaires leur


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

permettent de représenter les transferts entre le fluide simulé et le fluide procédural. Une fois le déplacement des particules effectué ils peuvent calculer le volume de fluide présent à l'intérieur de la zone de simulation en fonction des positions et vitesses des particules. En pratique, leur système ne requiert que deux étapes supplémentaires par rapport à une simulation hybride classique : la discrétisation du maillage et le calcul des propriétés dans les cellules correspondant au maillage.

### 5.4.1 Problématique et objectifs

La méthode permettant la simulation d'une fenêtre dynamique proposée par Stomakhin et Selle [SS17] est très intéressante car elle permet la simulation locale d'un océan généré par une méthode procédurale. Cependant, plusieurs points limitent l'utilisation directe de leur méthode pour nos simulations. Le point le plus évident est que la méthode de simulation qu'ils utilisent est une approche hybride et non pas lagrangienne. Bien que les transferts de fluide entre la zone simulée et l'extérieur soit fait sur l'une des étapes de simulation leur méthode hybride utilise une représentation particulaire. Le calcul des propriétés du fluide au niveau des bordures de la zone simulée est réalisé dans une représentation lagrangienne. L'avantage de cette solution est que lors de l'étape de transfert toutes les particules sont des particules de fluide, il n'y a pas de différence entre les particules qui sont dans la zone de simulation et celles qui sont hors de la zone. Avec un modèle lagrangien, les particules qui sont hors de la zone de simulation sont les particules utilisées pour calculer les propriétés du fluide et elles représentent un solide (une boite par exemple si notre fenêtre de simulation est rectangulaire). Cela cause un problème majeur : les méthodes utilisées actuellement pour représenter les bordures de l'espace de simulation d'un fluide lagrangien n'utilisent pas des particules avec une distribution similaire à celle d'un fluide au repos, comme nous l'avons vu lors de notre étude des méthodes de gestion des conditions aux limites pour un fluide lagrangien (section 3.2.4). Dans la méthode la plus courante, proposée par Akinci et al. [Aki+12], les solides sont représentés avec une seule couche de particules et le volume de chaque particule de cette couche est déterminé dynamiquement. Par conséquent, une particule de solide représente généralement un volume plus élevé qu'une particule de fluide. Il est possible d'utiliser plusieurs couches de solides pour représenter les bordures de notre espace de simulation de manière à ce que le volume calculé pour chaque particule de solide soit identique au volume d'une particule de fluide. Cependant, une telle méthode nécessiterait toujours que la disposition des particules de solide soit similaire à celle d'un fluide au repos. Même si l'on suppose que l'on dispose d'une méthode, qui n'existe pas à notre connaissance, permettant de distribuer les particules de solide de manière similaire à un fluide au repos, il reste le problème du calcul de propriétés telles que la pression au niveau des particules de solide de manière à ce que les interactions avec les particules de fluide soient réalistes. Pour ce faire, il faudrait ajouter une autre couche de particules autour des particules de solides de manière à ce que ces dernières aient un voisinage complet permettant ainsi un calcul correct des propriétés du fluide. Cette solution semble possible, mais elle a un défaut majeur : elle nécessite un nombre de particules solides au niveau des bordures de l'espace de simulation beaucoup plus élevé que le nombre utilisé avec la méthode de Akinci et al. [Aki+12]. La conséquence la plus évidente est que cette nouvelle approche augmenterait de manière probablement significative le temps de calcul nécessaire à chaque pas de simulation. L'un des objectifs de cette thèse est d'obtenir une simulation de fluide aussi interactive que possible. Par conséquent, nous avons choisi de ne pas continuer nos recherches dans une direction où il serait obligatoire d'effectuer des transferts entre les particules de solides représentant les bordures du fluide et les particules du fluide.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

Au lieu de chercher à simuler une fenêtre dynamique dans un espace complexe tel qu'un océan avec la présence de vagues, nous allons tout d'abord nous concentrer sur un cas plus simple : un océan plat. Ce scénario est beaucoup plus simple car sur un océan plat, l'état du fluide hors de la zone simulée est identique à celui dans la zone simulée. Également, dans ce scénario il n'y a pas besoin de transferts de particules entre la zone simulée et l'extérieur du fluide car un océan plat est théoriquement au repos. Ceci nous permet de nous concentrer sur la difficulté associée à une fenêtre de simulation dynamique : créer de nouvelles particules dans le sens du déplacement de la fenêtre simulée et retirer les particules là où elle se trouvait précédemment. L'utilisation d'une fenêtre dynamique est généralement justifiée par l'envie de suivre un objet se déplaçant au sein du fluide. Dans ce cas, l'intérêt d'une simulation d'un fluide vient des interactions réalistes entre l'objet et le fluide. En pratique, si nous plaçons un objet solide au centre de la zone simulée, le plus important est que le déplacement de la zone simulée ne fasse pas apparaître de perturbations majeures dans le fluide à proximité de l'objet simulé. Si la méthode proposée provoque l'apparition de perturbations au niveau des bordures de l'espace de simulation, il est nécessaire qu'elles soient assez faibles pour qu'elles se dissipent avant d'atteindre l'objet simulé. En pratique, cela veut dire que nous nous autorisons des manipulations majeures des particules au niveau des bordures de l'espace de simulation tant que le fluide au niveau du personnage n'est pas perturbé.

Pour résumer, notre système de fenêtre de simulation dynamique aura les objectifs suivants :

— Utilisation d'une représentation strictement lagrangienne

— Représentation d'un volume de liquide plat

— Absence de transferts de particules entre la fenêtre simulée et l'espace extérieur

— Simulation du fluide au niveau du centre de la fenêtre simulée non perturbée par le déplacement de la fenêtre

— Minimisation des calculs réalisés

### 5.4.2 Fenêtre dynamique de fluide lagrangien sans transfert

Avant de présenter notre méthode nous voudrions faire remarquer trois points importants :

— **Le nombre de particules simulées doit rester constant.** En effet, vu que le niveau de fluide est constant, le volume de fluide simulé doit également être constant peu importe où se trouve la fenêtre de simulation. Par conséquent, le nombre de particules présentes dans la simulation doit rester constant au cours du déplacement de la fenêtre.

— **La disposition des particules de fluide proches des bordures de l'espace de simulation est différente de celle au centre de la simulation.** Il suffit de regarder l'état de la simulation près de l'un des angles de la fenêtre simulée lorsque le fluide est au repos pour remarquer que les particules prennent une disposition particulière. Comme nous pouvons le voir sur la figure 5.9, les particules de fluide au niveau de l'angle s'alignent parfaitement. Ceci n'est pas dû à une disposition particulière des particules représentant la bordure du fluide car dans l'exemple utilisé pour la figure 5.9 les particules de la bordure ne sont pas organisées de manière régulière (leurs position sont déterminées par un échantillonnage de Poisson de la surface d'une boite). La conséquence est que lors du déplacement de la fenêtre il sera plus intéressant d'ajouter et de retirer des particules légèrement à l'intérieur de la zone simulée au lieu d'effectuer ces opérations directement au niveau de la bordure de manière à éviter l'introduction d'artefacts dus à la disposition particulière des particules. Cela nous permettra d'éviter d'avoir à considérer les problèmes liés à des dispositions particulières de particules.


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

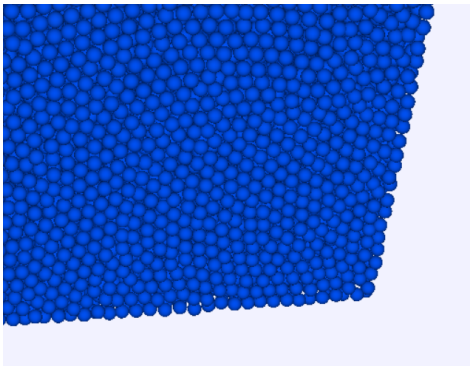


**Figure 5.9.:** Configuration de particules près d'un des coins de la simulation dans un scénario de fluide au repos. Nous pouvons observer un alignement élevé des particules au niveau de l'angle de la simulation.

— **La fenêtre dynamique doit être rectangulaire.** La méthode que nous proposons vise à déplacer les particules en fonction de leur cellule de la grille uniforme utilisée pour construire le voisinage. Afin de limiter la complexité des calculs nécessaires, la fenêtre dynamique doit permettre la définition de tranches de fluides perpendiculaires au sens de déplacement de la fenêtre et au sein desquelles la distance de chaque cellule à la bordure de la fenêtre est constante.

La figure 5.10 (en deux parties) propose une vue d'ensemble du processus de déplacement de la fenêtre de simulation. Dans la figure 5.10a nous visualisons une vue en coupe du fluide dans un plan perpendiculaire au sens du mouvement désiré. Rappelons que nous nous limitons à des déplacements perpendiculaires à la bordure de la fenêtre dynamique. Nous définissons une colonne de fluide par toutes les particules se trouvant dans une cellule de la grille uniforme avec la même coordonnée dans le sens du déplacement. Rappelons que la taille d'une cellule de la grille dynamique est fixée à 4 fois le rayon des particules de fluide. Dans la suite de ce document, chaque colonne de fluide est associée à un numéro pour simplifier la présentation du processus. La première étape consiste à déplacer les particules de solide représentant les bordures de la fenêtre dynamique (figure 5.10b). Comme nous l'avons expliqué, la distribution des particules proches des angles de la fenêtre dynamique est particulière. Par conséquent, pour éviter l'introduction de perturbations dans le fluide, nous avons décidé de déplacer telles quelles (sans modifier les positions de particules à l'intérieur des colonnes) les deux colonnes de fluides contenant les angles de la fenêtre dynamique, ici les colonnes 0 et 13 (figure 5.10c). À la fin de cette étape nous remarquons deux choses. Premièrement, les particules de la colonne 1 ont dû être retirées du fluide pour être remplacées par les particules de la colonne 0. À l'inverse, il y a maintenant une nouvelle colonne entre les colonnes 12 et 13 ne contenant aucune particule. Nous déplaçons alors toutes les particules de la colonne 1 dans cette nouvelle colonne (figure 5.10d). Avant de continuer à manipuler nos particules étudions l'état de notre fluide. Nous avons maintenant 3 paires de colonnes adjacentes qui ne l'étaient pas originellement : 0-2, 12-1 et 1-13. Ceci a de grandes chances de provoquer des problèmes de stabilité au niveau de la jonction des deux colonnes de chacune de ces paires. En effet, si dans chacune des colonnes l'une des particules se trouvait proche de la bordure, il est maintenant possible qu'au niveau de la jonction certaines particules se trouvent très proches les unes des autres. Cela provoquerait l'apparition de fortes surpressions dans le fluide pouvant mener à des artefacts voire d'importantes perturbations dans le fluide. Avant de chercher une solution à ce problème, il est intéressant de chercher à minimiser le nombre de jonctions. Il est possible de se limiter à 2 jonctions en modifiant les positions des particules de la colonne 1 de manière à ce qu'elles correspondent aux particules de la



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

colonne 12 (figure 5.10e). Cela permet d'éviter l'apparition d'une jonction entre la colonne 1 et la colonne 13. Pour une meilleure lisibilité, nommons 12' la colonne 1 avec les particules repositionnées selon la colonne 12. Il nous reste maintenant 2 paires présentant une jonction potentiellement instable : 0-2 et 12-12'. Si nous représentons maintenant les positions de toutes les particules (figure 5.10f), nous modifions la couleur des particules se trouvant à une faible distance de la jonction, en jaune ou en rouge selon si elles se trouvent à gauche ou à droite de la jonction.

Nous pouvons voir les nombreuses superpositions entre les particules se trouvant au niveau de la jonction. Si l'on ne traite pas ces superpositions elles provoqueront des surpressions qui mèneront à d'importantes perturbations dans le fluide. Il est donc nécessaire de déplacer certaines de ces particules vers la surface du fluide de manière à retirer la présence de superpositions. La solution optimale serait de calculer la densité au niveau de chacune de ces particules puis de déplacer les particules pour lesquelles nous détectons une densité supérieure à celle des particules proches. Cependant cette solution serait coûteuse en temps de calcul et nécessiterait deux kernels CUDA séquentiels. La solution que nous illustrons dans notre exemple (figure 5.10g) est de retirer les particules rouges pour lesquelles la distance à la particule jaune la plus proche est inférieure à un seuil (ex : par exemple deux tiers du diamètre d'une particule). Il est possible d'utiliser d'autres conditions, par exemple une solution naïve est de simplement déplacer les particules proches de la jonction. Suivant la règle utilisée le fluide obtenu sera plus ou moins stable et un système de seuillage des vitesses peut être nécessaire au cours des pas de temps suivant le déplacement de la fenêtre dynamique. Les détails d'un tel système sont décrits dans la section suivante (section 5.4.3). Dans notre exemple, les particules déplacées vers la surface sont placées les unes à coté des autres dans le plan horizontale, la première particule étant placée au niveau de la bordure du fluide. En bleu pâle nous avons représenté les particules qui se trouvaient auparavant au niveau de la jonction. Les particules roses correspondent à des particules de la colonne 1 qui n'ont pas été associées à une nouvelle position dans le cas où la colonne 1 contient plus de particules que la colonne 12. La section 5.4.4 présente les détails des règles employées pour déterminer la position des particules déplacées.

### 5.4.3 Amortissement de particules

Comme nous l'avons évoqué précédemment, selon la règle utilisée pour déterminer quelles sont les particules se trouvant aux plans de jonction devant être déplacées, il peut être nécessaire de limiter les vitesses des particules au cours des pas de simulation suivant le déplacement de la fenêtre dynamique. En principe un tel système ne devrait pas être nécessaire si nous retirions la totalité des particules provoquant une surpression. Cependant, déplacer un nombre élevé de particules n'est pas une solution idéale et déplacer des particules vers la surface du fluide n'est pas une solution idéale non plus. Si un nombre trop élevé de particules sont déplacées vers la surface cela provoque l'apparition de deux problèmes. Premièrement, le retrait de ces particules risque de faire apparaître d'une partie "vide" à l'intérieur du fluide ce qui provoque une dépression. Le second problème est que les particules ajoutées au niveau de la surface vont "appuyer" sur la surface du fluide. Cela génère une vague qui se propage à la surface entraînant ainsi des perturbations dans la zone centrale de la simulation que nous essayons de garder imperturbée. L'amplitude de ces deux problèmes est directement liée au nombre de particules déplacées. Il est donc dans notre intérêt de minimiser le nombre de particules à déplacer. L'objectif est donc d'arriver à déterminer le nombre maximum de surpressions que nous pouvons autoriser sans qu'elles ne provoquent des perturbations trop importantes. Il est possible de limiter l'impact de ces surpressions en limitant la vitesse des


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

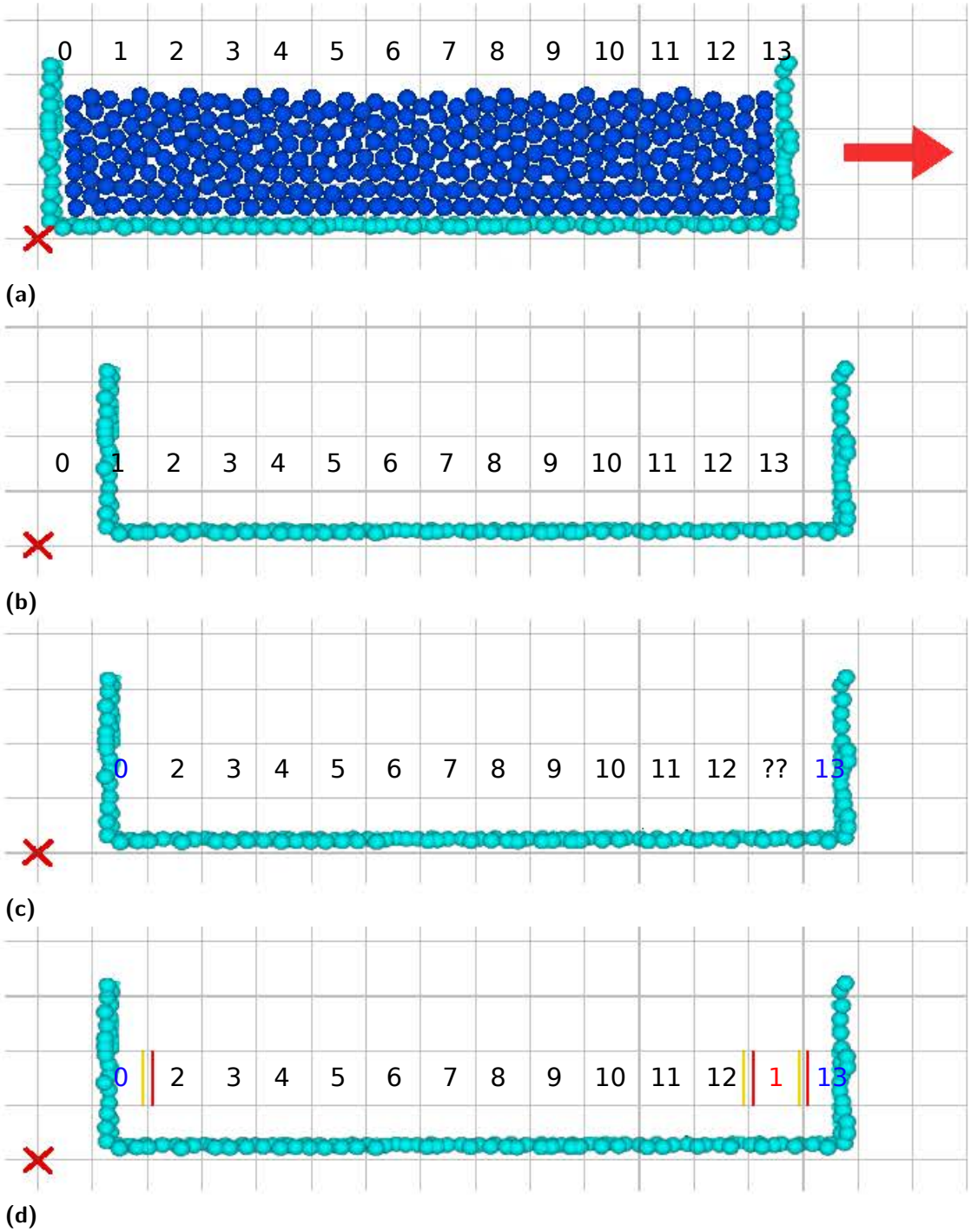



**Figure 5.10.:** Représentation du processus de déplacement de notre fenêtre dynamique de simulation (partie 1). La croix rouge représente le point d'origine du monde simulé. La flèche rouge de la première image représente la direction de déplacement.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

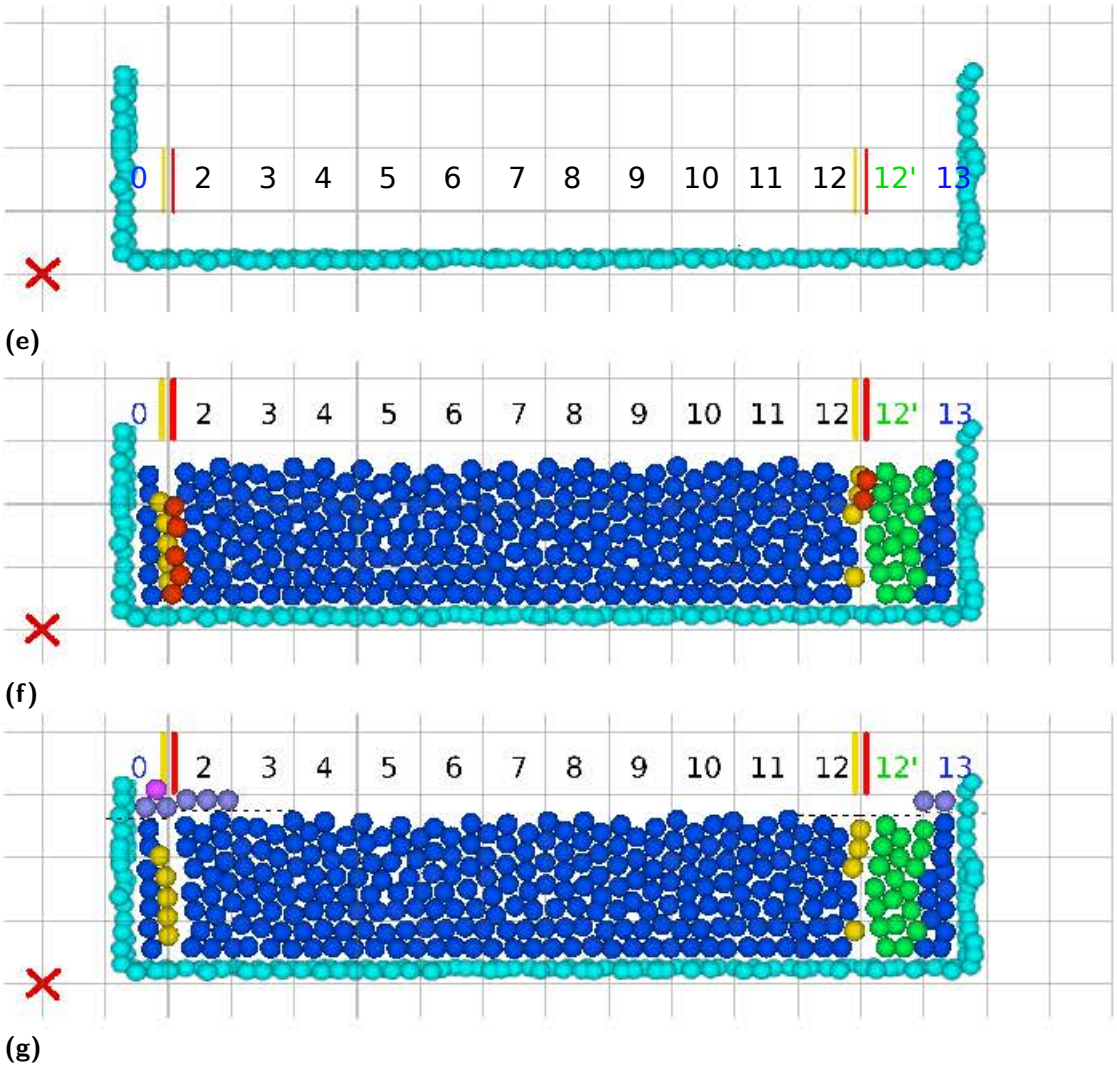



**Figure 5.10.:** Représentation du processus de déplacement de notre fenêtre dynamique de simulation (partie 2). La croix rouge représente le point d'origine du monde simulé.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

particules. Si leur impact est réduit il est possible de déplacer moins de particules tout en conservant le même niveau de stabilité. C'est la raison pour laquelle nous introduisons un système d'amortissement des particules.

Notre système fonctionne simplement en seuillant la vitesse maximale des particules à des zones précises de la simulation pour une courte durée de temps. L'objectif est de laisser quelques pas de simulation au fluide pour que les particules au niveau des surpressions soient légèrement déplacées et que les forces produites soient trop faibles pour provoquer des perturbations importantes. Pour ce faire, il est théoriquement nécessaire de définir comme zone de limitation toutes les zones où des particules trouvent dans leur voisinage des particules qui n'auraient pas naturellement été là. Les deux zones évidentes sont les deux plans au niveau desquels nous avons eu des jonctions de zones de fluide. Les particules se trouvant dans la zone centrale du fluide au niveau de la bordure du fluide sont également dans ce cas car nous avons déplacé les particules formant la bordure de l'espace de simulation. Par conséquent, il devrait être nécessaire d'également ajouter une zone d'amortissement au niveau des bordures du fluide qui sont perpendiculaires au déplacement de la fenêtre de simulation. En pratique, nous avons observé que les surpressions au niveau des bordures du fluide sont relativement faibles et n'ont un impact notable que si elles sont très nombreuses. Cependant, l'amortissement d'une particule n'ayant aucun coût en temps d'exécution, nous avons choisi de conserver l'amortissement des particules proches des bordures de l'espace de simulation pour retirer les quelques artefacts visuels présents.

### 5.4.4 Placement des particules sur la surface

Dans cette section, nous allons décrire les règles employées pour déterminer la nouvelle position des particules pour lesquelles une surpression est détectée au niveau des plans de jonction. Avant tout, rappelons que si nous plaçons un grand nombre de particules au niveau de la surface du fluide, cela entraîne la création d'une vague. Par conséquent, s'il nous est possible de placer les particules à déplacer sous la surface du fluide cela nous permettrait de limiter le nombre de particules au-dessus de la surface. Lors de la quatrième étape du processus de déplacement de notre fenêtre dynamique (figure 5.10d) nous avons déplacé les particules se trouvant dans la colonne 1 vers la colonne 12' leur position de manière à dupliquer la colonne 12. Si la colonne 1 contient plus de particules que la colonne 12, alors les particules restantes sont simplement placées au niveau de la surface. Cependant, au cours de nos expériences, nous avons remarqué que le cas inverse, c.à.d. la colonne 12 contient plus de particules que la colonne 1, est bien plus fréquent. La première règle de positionnement des particules à déplacer est donc de compléter la colonne 12' s'il reste des positions disponibles. Notons qu'il est nécessaire de répéter la vérification au niveau du plan de jonction pour les particules nouvellement placées dans la colonne 12' car elles peuvent présenter une surpression.

S'il reste des particules à déplacer, nous choisissons forcés de les placer au niveau de la surface. Étant donné que ces particules proviennent des plans de jonction qui sont proches des bordures du fluide dans l'axe du déplacement, nous décidons de les placer au plus proche des bordures en question. Supposons dans un premier temps que nous répartissons équitablement les particules entre les deux bordures. Nous définissons au niveau de chaque bordure une zone correspondant à $1/5$ de la longueur de la zone simulée dans l'axe du déplacement. Dans le plan horizontal, les particules sont simplement placées selon une grille régulière de manière à ce qu'elles soient adjacentes les unes aux autres. Les particules sont placées dans cette grille en commençant par la ligne la plus proche de la bordure et dans les lignes suivantes à



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

chaque fois qu'une nouvelle ligne est complétée. Pour déterminer la position verticale des particules, nous regardons la hauteur maximale des particules pour chaque colonne de la grille uniforme utilisée pour effectuer la recherche de voisins. La hauteur des particules déplacées correspond à la hauteur maximale incrémentée du diamètre d'une particule de manière à ce que la nouvelle particule soit tangente à la surface du fluide existant. Si assez de particules sont déplacées pour compléter une couche horizontale, les particules sont placées dans une seconde couche avec une position verticale incrémentée encore une fois du diamètre des particules et ainsi de suite jusqu'à ce que toutes les particules soient placées. Pour permettre à notre fluide d'être dans un niveau de compression aussi proche que possible de notre fluide au repos, les couches de particules déplacées avec un indice impair (la première couche ayant l'indice 0) sont décalées d'une distance correspondant au rayon des particules selon les deux axes du plan horizontal. Cette imbrication des couches de particules déplacées et nous permet de diminuer la distance entre les couches. Dans nos expériences, nous avons pu observer que la distance minimale entre deux couches ne provoquant pas d'artefact correspond à $80\%$ du diamètre des particules. Le résultat obtenu par cette méthode de placement des particules est illustré dans la figure 5.11a.

Bien que cette méthode de placement fonctionne correctement, elle a tendance à amplifier la vague générée par le déplacement des particules. La raison est que le placement des particules est toujours démarré proche de la bordure du fluide. Par conséquent au cours de déplacement successifs de la zone de simulation, cette méthode de placement a tendance à créer des empilements de particules au niveau de la bordure. Ceci augmente la hauteur du fluide au niveau de la bordure, amplifiant ainsi l'amplitude de la vague générée. Pour régler ce problème nous pouvons choisir une meilleure règle pour la position initiale pour le placement des particules. Pour déterminer cette première position, nous utilisons la position au sein de la zone de placement possible pour laquelle la surface du fluide est la plus faible. Cette nouvelle position initiale permet effectivement de diminuer l'amplitude de la vague observée. Cependant, une méthode de placement plus organique des particules (représentant réellement un fluide au repos) pourrait encore améliorer le résultat obtenu. Le résultat obtenu par cette méthode de placement des particules est illustré dans la figure 5.11b.

Enfin revenons sur notre choix de répartir les particules à placer équitablement entre les deux bordures dans l'axe du déplacement. Vu que nous nous déplaçons vers l'une des deux bordures, les perturbations générées au niveau de cette bordure auront un impact plus important sur le centre de la simulation. En effet, les perturbations introduites vers l'arrière du mouvement nécessitent plus de temps pour atteindre le centre de l'espace de simulation car la distance entre le point de génération de la perturbation et le centre de la simulation augmente à chaque déplacement de la fenêtre dynamique. Par conséquent, il semble intuitif de placer un maximum de particules au niveau de la bordure se trouvant vers l'arrière du déplacement pour que la vague ainsi produite se dissipe avant d'atteindre le centre de la zone de simulation. Dans nos expériences, nous avons pu voir que même avec un rapport de 1 :9, c.à.d. 1 particule vers l'avant pour 10 particules déplacées, et une vitesse de déplacement de $0.5m.s^{-1}$ les vagues générées par les particules déplacées vers la surface au niveau des deux bordures atteignent le centre de la zone simulée simultanément. Cette nouvelle répartition diminue de manière très significative l'amplitude de la vague observée. Le rapport de distribution des particules dépend probablement de la vitesse de déplacement de la zone dynamique et un rapport dynamique dépendant de ce paramètre pourrait améliorer la généricité de notre solution.


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

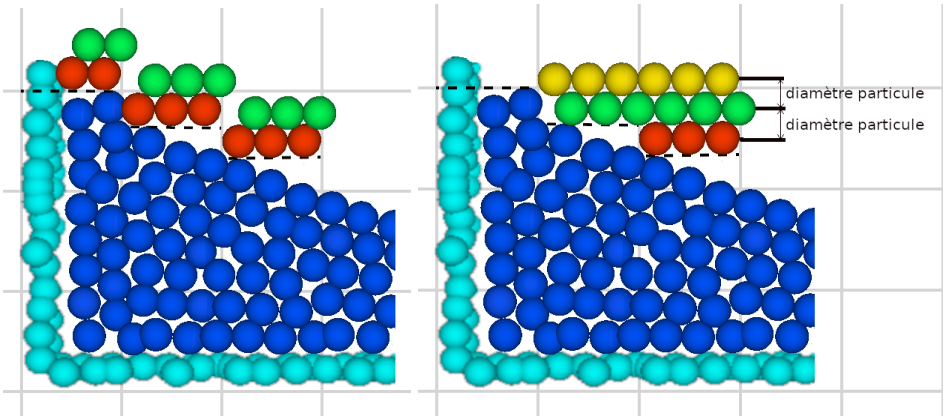



**Figure 5.11.:** Vue latérale d'un exemple de placement de particules au niveau de la surface du fluide ; (a) : méthode basique ; (b) : méthode pour limiter la vague. Dans cet exemple la longueur de la zone de placement autorisé correspond à trois colonnes de fluide. Les couleurs placées au niveau de la surface sont les particules de couleur rouge, vert et jaune, chaque couleur correspondant à une couche de particules dans cet ordre. Notons que dans nos expériences une troisième couche n'a jamais été observée, et n'a été donné dans la figure b pour montrer que la hauteur des différentes couches des particules ne dépend que de la hauteur de la colonne de fluide la moins élevée.

## 5.5 Expérimentations

Nous avons réalisé nos expérimentations sur un ordinateur portable MSI GT62VR 6RE. Cet ordinateur dispose d'un processeur I7-6700HQ et d'une carte graphique NVIDIA GeForce GTX 1070. Les interactions entre le fluide et les solides sont gérées en utilisant la discrétisation proposée par Akinci et al. [Aki+12]. Les forces autres que les forces de pression appliquées au fluide sont la viscosité, la tension de surface et la gravité. La viscosité est calculée en utilisant la formulation proposée par Schechter et Bridsoné [SB12]. La tension de surface est calculée en utilisant le modèle proposé par Akinci et al. [AAT13]. Pour notre implémentation, nous utilisons le *CUDA OpenGL interop* pour allouer directement dans OpenGL les tableaux de position et de vitesse des particules évitant ainsi d'avoir à faire une copie lors du rendu. Pour cette section, le pas de temps de la simulation est fixé à 3ms (333 Hz). Les temps de calcul indiqués ne prennent en compte que la partie simulation physique et ne contiennent pas le temps de rendu. Pour toutes nos expériences le rendu est effectué à chaque fois qu'un pas de simulation est terminé.

### 5.5.1 Optimisation

Avant de comparer les résultats de notre implémentation sur GPU avec l'implémentation CPU parrallèle proposée par Bender et Koschier dans le framework SPlisHSPlasH, nous réalisons les expériences proposées dans la section 5.3.2 pour déterminer la configuration apportant les meilleures performances sur carte graphique. Pour ces expériences, nous utilisons une simulation comportant 78000 particules de fluide et 54800 particules de solide représentant les limites de l'espace de simulation.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

### Cohérence spatiale des données

La principale différence entre les différentes configurations (Morton, linéaire et linéaire avancé) est le niveau de cohérence spatiale que chacun permet d'obtenir. Pour réaliser nos expériences nous utilisons une simulation où le fluide est au repos. Cela permet d'avoir un voisinage aussi complet que possible pour chaque particule augmentant ainsi l'importance de la cohérence spatiale des données. Pour notre premier test, nous comparons les temps d'exécution de la recherche du voisinage pour chaque configuration dans le cas où l'ordre des particules est aléatoire, ce qui nous offre une faible cohérence spatiale. Ce test a pour but d'étudier les temps d'exécution de chaque configuration lorsque le niveau de cohérence spatiale n'impacte pas les performances. Ceci permet de vérifier si l'utilisation d'un indice requirant plus de calculs a un impact visible sur le temps d'exécution. Trois configurations sont explorées : l'indice de Morton, un indice linéaire classique, et un indice linéaire avec une exploration des 27 cellules du voisinage en utilisant des groupes de 3 cellules consécutives comme proposé par Domingez et al. [DCG11]. Les résultats sont visibles dans la table 5.1.

| | Morton | linéaire | linéaire avancé |
|---|---|---|---|
| création de la structure | 2.12 | 2.15 | 2.15 |
| recherche des voisins | 6.82 | 6.87 | 6.52 |

**Tableau 5.1.:** Temps d'exécution (en ms) nécessaire à la recherche du voisinage des particules. Les temps sont séparés en deux parties : la création de la structure accélératrice et l'utilisation de cette structure pour trouver les particules présentes dans le voisinage.

Nous remarquons que les temps de calcul pour les deux indices, Morton et linéaire, sont similaires ce qui veut dire que le coût des calculs plus complexes de l'indice de Morton est négligeable relativement au temps de calcul total. Également nous pouvons voir que l'exploration des cellules consécutives proposée par Domingez et al. [DCG11] permet en effet de diminuer le temps nécessaire à l'exploration de la structure de données de manière significative, environ $10\%$. Dans le cas d'une implémentation où le voisinage ne peut pas être sauvegardé pour chaque particule, cette amélioration pourrait permettre un gain de performance notable de par la répétition de l'exploration de la structure. Cependant, la structure de données n'est explorée qu'une seule fois et par conséquent ce gain de performance, 0.35ms, est relativement négligeable par rapport au temps de calcul total pour un pas de simulation complet, environ 28.54ms (table 5.2). Pour le reste de nos expériences, nous utiliserons l'algorithme de Domingez et al. [DCG11] à chaque fois qu'un indice linéaire est utilisé.

Pour notre second test, nous mesurons le temps moyen nécessaire à un pas de simulation complet pour un fluide au repos. Nous comparons l'utilisation de l'indice de Morton et de l'indice linéaire avancé avec un ordre aléatoire des particules ou bien avec des particules parfaitement ordonnées suivant l'indice utilisé. Cela nous permet d'observer l'impact de la cohérence spatiale des données pour chacun des deux indices (table 5.2). Pour les configurations "particules non-triées", les particules sont placées dans un ordre aléatoire différent à chaque pas de temps.

La différence observée entre les temps d'exécution des deux indices pour les particules non triées est due à l'exploration groupée des cellules voisines lors de la recherche du voisinage. Cet écart est négligeable car il est inférieur à la marge d'erreur observée due aux variations du nombre d'itérations nécessaires pour le solveur de densité constante. Les résultats obtenus avec les particules triées montrent un agrandissement de l'écart de performance entre les deux


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

| | Morton | linéaire avancé |
|---|---|---|
| particules non triées | 84.33 | 83.90 |
| particules triées | 31.75 | 28.54 |

**Tableau 5.2.:** Temps d'exécution (en ms) moyen pour un pas de simulation selon l'indice utilisé et l'ordre des particules. Cette comparaison a pour but de permettre de visualiser l'impact de la cohérence spatiale des données sur le temps d'exécution.

| | Recherche voisins | Solveur divergence | Forces externes | Maj vitesses | Solveur densité | Maj positions |
|---|---|---|---|---|---|---|
| Morton | 5.10 | 5.17 | 1.17 | 0.14 | 20.01 | 0.14 |
| Linéaire avancé | 4.57 | 4.74 | 1.06 | 0.12 | 17.92 | 0.12 |

**Tableau 5.3.:** Répartition du temps d'exécution entre les différentes étapes d'un pas de temps. Les particules sont triées en fonction de l'indice utilisé pour maximiser la cohérence spatiale des données.

indices en faveur de l'indice linéaire d'environ $10\%$. Si on regarde le détail de la distribution du temps de calcul pour chacun des indices (table 5.3), on voit que cette fois l'écart entre les deux configurations est distribué entre les différentes étapes de la simulation de manière homogène. Ce résultat nous indique que l'indice linéaire nous permet une meilleure cohérence spatiale que l'indice de Morton. Cette observation contre-intuitive est possiblement due au fait que notre simulation n'utilise pas la totalité de la grille régulière mais seulement une petite zone de celle-ci. L'indice de Morton est supposé offrir une meilleure cohérence spatiale mais cela n'est peut-être vrai que lorsque la totalité de la grille est utilisée. Cependant ce genre de condition limite énormément le nombre de configurations de simulation possible car très peu de simulations de liquides utilisent la totalité de la grille de manière fréquente (en particulier la partie haute de la grille). Notons que dans le cas d'une simulation de gaz cette condition serait respectée et les résultats de cette expérience pourraient être différents. pour nos expérimentations, nous avons choisi d'utiliser l'indice linéaire avancé pour nos simulations de manière à bénéficier des temps de calcul plus faibles.

### Pré-calcul des fonctions de noyau

Pour notre seconde expérience nous allons étudier l'impact des tables de correspondance utilisées pour le calcul des fonctions de noyau comme proposé par Bender et Koschier [BK15] (section 5.3.2). Nous utilisons la configuration précédente avec le fluide au repos et les particules triées. Les tables de correspondance utilisées sauvegardent 1000 valeurs.

| | calcul direct | table de correspondance |
|---|---|---|
| 32 bits | 28.34 | 31.52 |
| 64 bits | 55.88 | 41.92 |

**Tableau 5.4.:** Temps d'exécution (en ms) moyen pour un pas de simulation suivant si une table de correspondance ou un calcul direct est utilisé pour déterminer les valeurs de la fonction de noyau. Nous réalisons des mesures séparées suivant si les nombres flottants utilisent une simple précision ou une double précision.

Nous pouvons voir (table 5.4) que l'utilisation d'une table de correspondance augmente le temps de calcul d'environ $10\%$. Ce résultat semble contredire les observations de Bender et



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

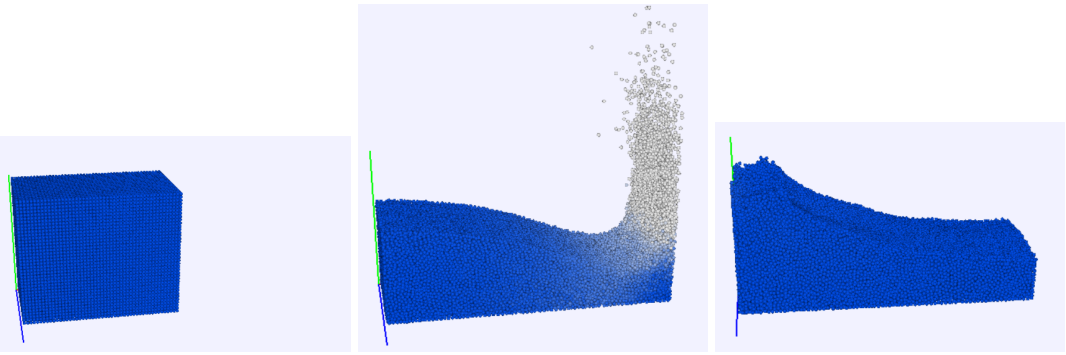


**Figure 5.12.:** Exemple de cassage de barrage. De gauche à droite les images sont : état initial, simulation après 200 pas de temps, simulation après 1000 pas de temps.

Kochier [BK15] qui observent un gain de performance de $30\%$. Cela s'explique par le fait que nous utilisons des réels en simple précision dans notre implémentation alors qu'ils utilisent des réels en double précision. Nos expériences utilisant des nombres flottants en double précision montrent en effet un gain de performance significatif. Nous noterons tout de même que le gain de performance, environ $25\%$, est légèrement plus faible que celui obtenu par Bender et Kochier. Ces résultats nous indiquent que les accès mémoire sur GPU sont assez lents pour annuler le gain de performance lié à la réduction des calculs nécessaires si l'on utilise des réels en simple précision. L'efficacité d'une carte graphique Nvidia de gamme grand public est très faible pour des calculs en double précision. En particulier, l'architecture GP100 (gtx1080, gtx1070...) dispose de 32 fois moins de cœurs 64 bits que 32 bits, ce qui explique que pour les calculs en double précision le gain de performance dépasse le coût des accès en mémoire. Dans nos expériences, nous avons remarqué qu'utiliser des flottants en simple précision en nuit pas significativement à la stabilité du fluide. Par conséquent, nos expériences sont réalisées en simple précision et nous n'utilisons pas de tables de correspondance pour le calcul des fonctions de noyau.

### Démarrage à chaud

Cette dernière expérimentation vise à évaluer l'importance du démarrage à chaud. Nous comparons les performances obtenues avec et sans le démarrage à chaud. Notre simulateur n'utilise qu'une seule itération du solveur de divergence nulle dans le cas d'un fluide au repos. Pour cette comparaison nous regarderons la moyenne du nombre d'itérations de chacun des solveurs ainsi que le temps moyen pour un pas de simulation sur deux scénarios : les 200 premiers pas de temps d'un scénario de cassage de barrage (figure 5.12) et 500 pas de temps d'un fluide au repos. Les résultats sont présentés dans la table 5.5.

| | Solveur densité (nombre d'itérations) | | Solveur divergence (nombre d'itérations) | | Temps moyen (en ms) | |
|---|---|---|---|---|---|---|
| démarrage à chaud | sans | avec | sans | avec | sans | avec |
| cassage de barrage | 16.9 | 5 | 6.3 | 5 | 48.04 | 27.24 |
| fluide au repos | 71.9 | 7.8 | 1 | 1 | 164.60 | 28.53 |

**Tableau 5.5.:** Nombre d'itérations de chacun des solveurs et temps d'exécution moyen pour un pas de temps de simulation suivant si un démarrage à chaud est utilisé. Deux scénarios sont présentés : les 200 premiers pas de temps d'un cassage de barrage et 500 pas de temps d'un fluide au repos.


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

Nous pouvons voir que l'utilisation d'un démarrage à chaud permet de réduire de manière significative le nombre d'itérations nécessaires en particulier pour la densité. L'utilisation de cette amélioration est critique dans le cas d'un fluide car elle permet de réduire le temps d'exécution de plus de $80\%$. Le scénario du cassage de barrage nous montre que lorsque le fluide est en mouvement le gain de performance est moins important, environ $45\%$, car le nombre d'itérations du solveur de densité constante est plus faible. Cette différence de gain de performance s'explique par deux raisons. Lorsque le fluide est au repos, les valeurs de pression au niveau des particules ne varient que très peu entre deux pas de simulation ce qui permet une utilisation très performante d'un démarrage à chaud. La seconde raison est qu'un fluide au repos est composé principalement de particules en compression. Par conséquent, si l'on corrige la compression au niveau d'une particule, on augmente celle au niveau des particules voisines. Ceci explique pourquoi le nombre d'itérations nécessaires au solveur de densité dans le cas d'un fluide au repos est beaucoup plus élevé que dans le cas d'un fluide en mouvement.

Suite à ces résultats nous pouvons donc proposer une configuration optimale pour notre simulateur de fluide : indice linéaire avancé pour la structure d'accélération pour la recherche du voisinage en utilisant un accès groupé pour les cellules consécutives, particules triées suivant l'indice, réels 32 bits, calcul direct des valeurs des fonctions de noyau et utilisation d'un démarrage à chaud. Si l'une de nos simulations utilise des réels 64 bits pour plus de précision, nous l'indiquerons dans la description de l'expérience et utiliserons des tables de correspondance pour le calcul des fonctions de noyau. Il est difficile d'estimer la fréquence optimale de tri des particules pour laquelle le temps d'exécution additionnel lié au tri des particules est compensé par le gain dû à la meilleure cohérence spatiale. Cependant, nous avons vu que les gains possibles sont élevés, également nous avons observé au cours de ces expériences que le temps de calcul nécessaire au tri est relativement faible, moins d'une milliseconde dans toutes les simulations utilisées. Nous avons donc choisi d'effectuer le tri tous les 15 pas de simulation. Avec cette fréquence de tri, le temps de calcul additionnel pour le tri des particules est de moins de $0.25\%$ du temps total de simulation. Rappelons que, pour un fluide au repos, nous avons observé une réduction du temps moyen nécessaire à un pas de temps de $66\%$ entre une simulation parfaitement triée et une simulation non triée. Ce gain important de performance justifie une fréquence de tri élevée. Notons cependant qu'il serait possible de diminuer la fréquence de tri quand le fluide est au repos car les particules ne bougent pas beaucoup. Cependant, il est justifié de ne pas optimiser davantage cet aspect de la simulation, car le temps de calcul nécessaire au tri des particules étant déjà négligeable lorsqu'il est effectué une fois tous les 15 pas de temps.

### 5.5.2 Fenêtre dynamique de simulation

Dans cette section nous allons explorer les résultats obtenus lors de l'utilisation de notre système de fenêtre dynamique de simulation. Dans un premier temps nous allons explorer une configuration "standard", puis nous présenterons des configurations spécialisées ainsi que les limites du système.

La configuration standard que nous utilisons pour évaluer notre système est la suivante :

— Zone de simulation carrée de 2 mètres de côté. Cette zone contient 36000 particules de fluide de 2.5cm de rayon, ce qui correspond à environ 1 mètre de hauteur de fluide au repos.

— Les particules déplacées à la surface du fluide lors du déplacement de la fenêtre dynamique sont les particules à l'avant du plan de jonction dans le sens du déplacement.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

Les particules déplacées sont les particules se trouvant à moins de $75\%$ de leur diamètre d'une particule de l'autre côté du plan de jonction. $90\%$ des particules déplacées à la surface sont placées à l'arrière de la zone de simulation dans le sens du déplacement et les $10\%$ restants sont placés vers l'avant.

— Les particules pour lesquelles un amortissement de la vitesse est effectué sont les particules proches d'un plan de jonction ou d'une des bordures latérales de la zone de simulation. La distance d'effet correspond au diamètre des particules. Dans ces zones, la vitesse des particules est limitée à une vitesse permettant un déplacement max de $\frac{1}{25}$ du diamètre de la particule en un pas de temps au cours des 5 pas de simulation suivant le déplacement de la zone simulée.

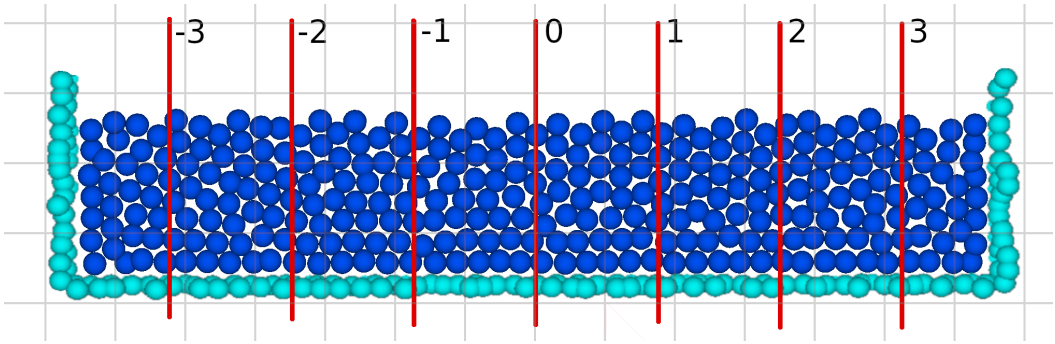



**Figure 5.13.:** Visualisation des plans utilisés pour étudier les perturbations introduites dans le fluide par le déplacement de la fenêtre dynamique.

La métrique que nous utilisons pour évaluer les perturbations dues à notre fenêtre dynamique est une estimation du flot provoqué par le décalage de la fenêtre. En pratique nous définissons 7 plans répartis de manière homogène dans la direction du déplacement (espacés de 0.2 mètres pour une zone simulée de 2 mètres de coté). Un plan est placé au centre de la zone simulée et trois sont placés vers l'arrière et trois vers l'avant. Le but est de déterminer la quantité de perturbations induites par les manipulations des particules au niveau des bordures. Nous associons à chaque plan une tranche de fluide centrée sur le plan et de largeur 0.2 mètres. La figure 5.13 permet de visualiser le placement des plans sur une vue de coté du fluide. Au sein de chaque tranche nous calculons la norme de la moyenne signée des vitesses des particules. Cela nous donne une estimation du déplacement global des particules au sein de chaque tranche que nous appellerons flot dans la suite de nos expérimentations. Pour un fluide au repos sans mouvement nous observons une valeur similaire au niveau de tous les plans. L'évolution de la valeur dans le temps au niveau du plan central est représentée dans la figure 5.14.

Nous pouvons voir que même pour un fluide au repos la valeur observée est bruitée. Cependant, la valeur moyenne observée est de $0.03m.s^{-1}$, ce qui correspond à un déplacement de $9.10^{-5}m$ avec nos pas de temps de 3ms, soit $0.18\%$ du diamètre des particules ce qui est négligeable.

Maintenant regardons l'évolution de la valeur lorsque nous déplaçons notre fenêtre dynamique (figure 5.15).

Le premier phénomène que nous pouvons observer est que notre système introduit bel et bien des perturbations au sein du fluide. Au niveau du plan central (plan numéro 0) cette perturbation atteint à son maximum un niveau 4 fois supérieur à la perturbation moyenne d'un fluide au repos. Notons tout de même que la perturbation au niveau de la zone centrale et de l'avant du fluide reste un flot inférieur à $1\%$ du diamètre des particules. De plus


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

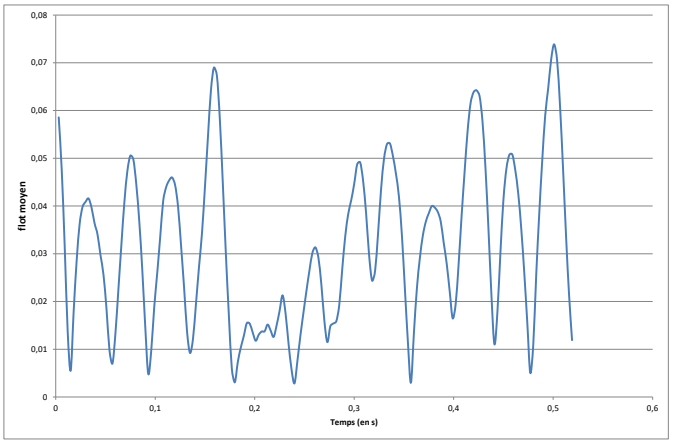



**Figure 5.14.:** Évolution du flot moyen au niveau du plan d'étude central dans de fluide au repos. Le flot moyen est calculé par la norme de la moyenne signée des vitesses des particules.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0,318038 | 0,298628 | 0,228916 | 0,187956 | 0,213393 | 0,246254 | 0,217052 |
| 2 | 0,287204 | 0,231458 | 0,18558 | 0,163031 | 0,159729 | 0,168968 | 0,150627 |
| 1 | 0,219869 | 0,159369 | 0,110238 | 0,104807 | 0,112046 | 0,138217 | 0,156707 |
| 0 | 0,175043 | 0,115611 | 0,0621808 | 0,0316906 | 0,0593632 | 0,0968165 | 0,126841 |
| | -3 | -2 | -1 | 0 | 1 | 2 | 3 |

**Tableau 5.6.:** Détail du flot moyen au sein du fluide en fonction de la position dans le sens du mouvement ainsi que de la hauteur. La couche supérieure ne contient aucune particule de fluide d'où le flot nul. Les valeurs bleues correspondent à un flot faible et les valeurs rouges à un flot élevé.

la perturbation introduite est complètement dissipée en 0.1s, soit 33 pas de temps. Nous pouvons observer un phénomène intéressant, les maximums sont tous atteints au même moment alors que nous aurions pu nous attendre à une propagation de la perturbation. Ceci est probablement dû au caractère incompressible du fluide associé à une zone de propagation faible.

La perturbation devrait être théoriquement plus forte au niveau de la surface du fluide car les particules déplacées devraient avoir comme conséquence la création d'une vague au niveau de la surface comme nous en avons discuté lors de la présentation du système d'amortissement des particules en section 5.4.3. Pour vérifier cette hypothèse nous découpons chaque tranche de fluide dans le sens de la hauteur selon des couches de 0.25m. Pour chaque sous-section nous regardons la valeur à l'instant où la perturbation au centre du fluide est à son maximal (table 5.6).

Dans cette table nous pouvons constater deux choses. Premièrement, bien que les plans de jonction produisent une perturbation, pour notre configuration ils ne sont pas la source principale de perturbation. En effet la perturbation maximale est observée au niveau de la surface. Cela indique que notre méthode de placement des particules au niveau de la surface est la partie du système introduisant le plus de perturbations malgré le fait que nous avons cherché à minimiser le nombre de particules replacées avec notre système d'amortissement. Notre deuxième constatation est que la perturbation semble se dissiper



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

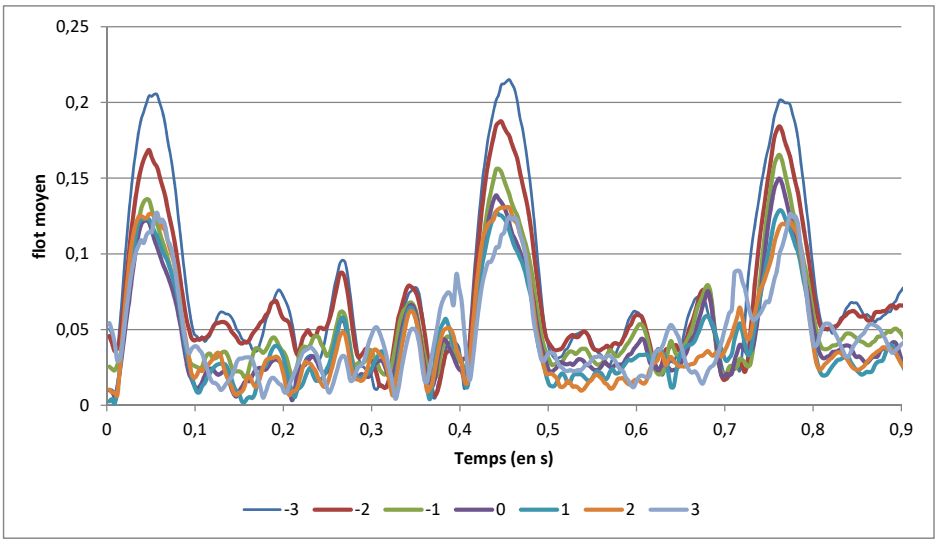



**Figure 5.15.:** Évolution du flot moyen au sein du fluide lors du déplacement de la zone de simulation pour chacun des 7 plans d'étude. Les déplacements de la fenêtre dynamique sont effectués environ toutes les 0.4 secondes.

relativement rapidement non seulement dans le sens du déplacement mais également dans la profondeur du fluide.

Maintenant que nous avons évalué notre configuration standard nous pouvons proposer une configuration spécialisée pour une simulation ayant un sens de déplacement principal. En effet, si notre simulation se déplace majoritairement dans une direction, il est possible de réduire la perturbation observée au niveau du centre de la simulation en utilisant une fenêtre dynamique rectangulaire. Pour cette configuration nous utilisons une fenêtre dynamique d'une longueur de 4 mètres dans la direction du mouvement et d'une largeur de 2 mètres. Pour avoir la même hauteur de fluide nous utilisons environ 75000 particules. Également, bien que nous calculions toujours le flot sur des tranches de fluide de 0.2 mètres, les plans de référence pour ces tranches sont maintenant espacés de 0.4 mètres pour nous offrir une vue de l'ensemble de la zone de simulation.

Notre système de fenêtre dynamique de simulation permet bien de maintenir un niveau de stabilité satisfaisant au centre de la zone de simulation en particulier lorsque la fenêtre de simulation utilisée est allongée dans le sens du déplacement.

Comme nous pouvons le voir sur la figure 5.16, utiliser une fenêtre dynamique plus grande dans le sens du déplacement diminue bien l'intensité des perturbations dans la zone centrale (plan numéro 0). Malgré le fait que les plans soient plus espacés nous observons que les perturbations introduites par notre système sont beaucoup plus faibles dans cette nouvelle configuration. Au niveau de la zone centrale du fluide, le niveau de perturbation est négligeable. Si l'on regarde le pic de perturbation dû au premier déplacement, à environ 0.45s, nous pouvons voir la propagation de la perturbation comme nous l'avions supposé. En effet, les zones éloignées du centre atteignent leur maximum de perturbation légèrement avant les zones centrales.


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

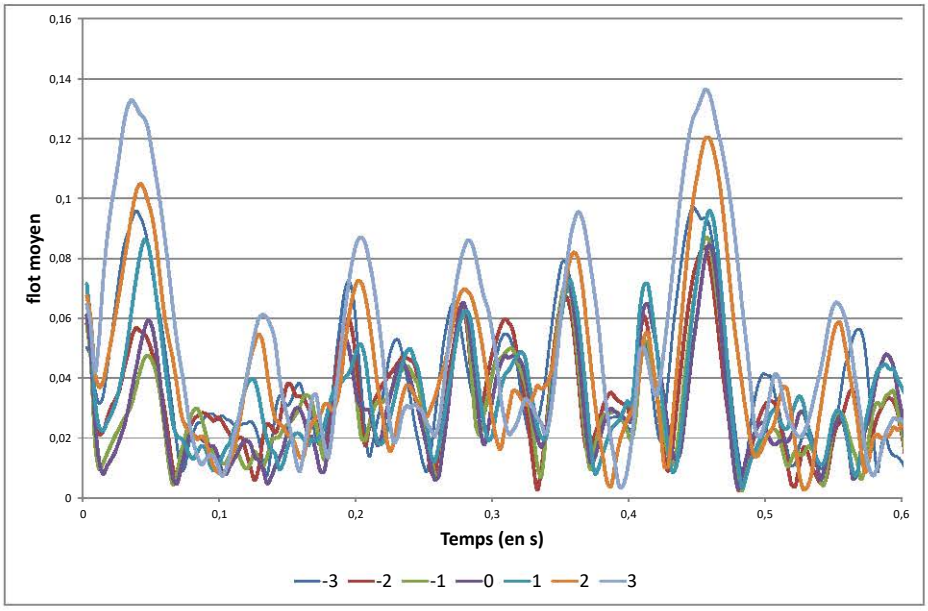



**Figure 5.16.:** Évolution du flot moyen au sein du fluide lors du déplacement de la zone de simulation. La zone de simulation pour cette expérience est deux fois plus longue que large avec la longueur dans le sens du déplacement. Les déplacements de la fenêtre dynamique sont effectués environ toutes les 0.4 secondes.

### 5.5.3 GPU vs CPU

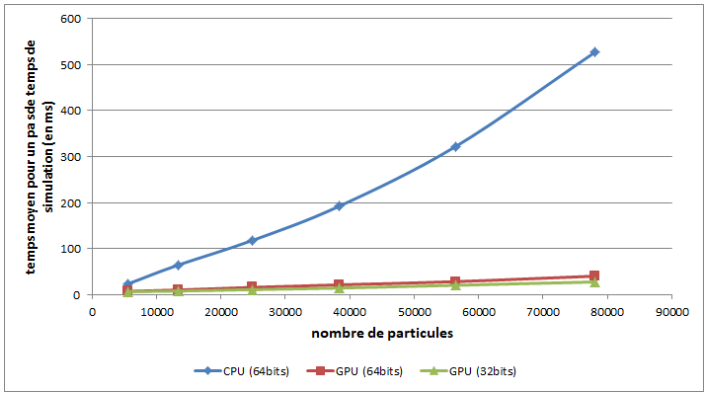
Dans cette section nous proposons de comparer les performances de notre simulateur de fluide à l'implémentation de l'algorithme DFSPH présente dans la librairie SPlisHSPlasH. Nous comparons trois configurations : CPU 64bits, GPU 64bits et GPU 32bits. Nous n'effectuons pas de comparaison avec une version CPU 32 bits pour des raisons techniques liées à l'utilisation de la librairie Eigen. Nous comparons le temps nécessaire à la simulation des 1000 premiers pas de temps d'un cassage de barrage (figure 5.12) pour différentes quantités de particules de fluide. La figure 5.17a donne l'évolution du temps de calcul en fonction du nombre de particules. La figure 5.17b donne l'évolution du gain de performance des versions GPU relativement à la version CPU 64 bits.

Nous constatons que notre implémentation GPU nous permet d'obtenir un gain de performance important. Ce gain de performance augmente linéairement avec le nombre de particules tant que la simulation contient un nombre suffisant de particules. La chute du gain relativement à la version CPU pour un nombre de particules faible s'explique par une sous-exploitation des capacités de la carte graphique. Une différence majeure par rapport aux facteurs d'accélération obtenus par l'implémentation GPU de l'algorithme IISPH par Goswami et al. [GEF15] est que leur facteur d'accélération se stabilise dès que le nombre de particules dépasse 20 000 alors que le nôtre continue d'augmenter. Cela semble indiquer que notre implémentation exploite moins bien la carte graphique. Une des raisons possibles est que l'algorithme DFSPH nécessite plus de kernels GPU que l'algorithme IISPH, ce qui diminue le facteur d'occupation des cœurs disponibles.

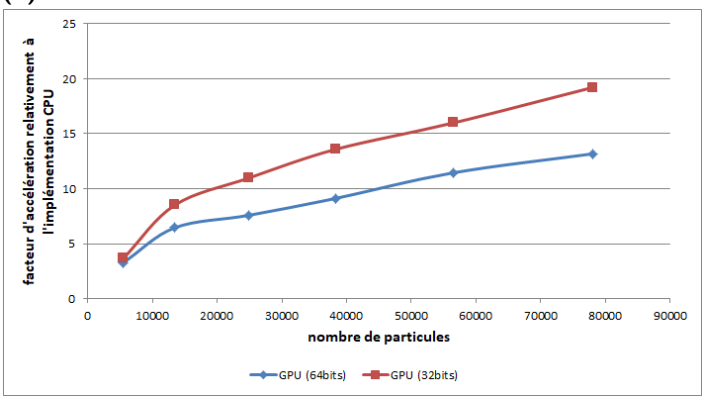
Un phénomène intéressant est que, contrairement à ce que nous pouvions attendre, le temps d'exécution de la version CPU (figure 5.17a) n'évolue pas parfaitement linéairement avec le nombre de particules mais l'augmentation devient plus rapide pour les grands nombres de particules. La raison de ce phénomène est que le nombre d'itérations des solveurs de densité



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés




(a)

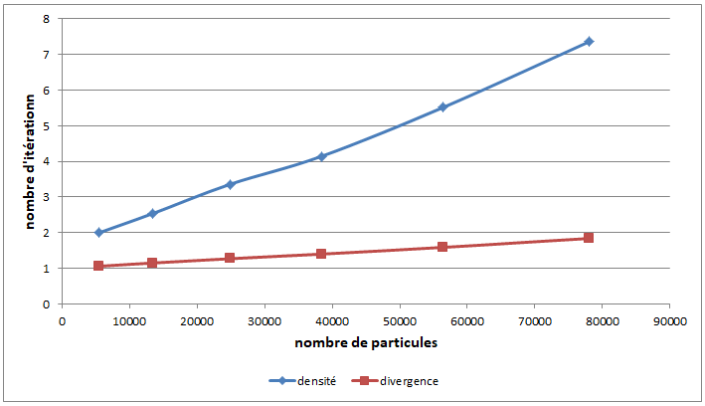



(b)

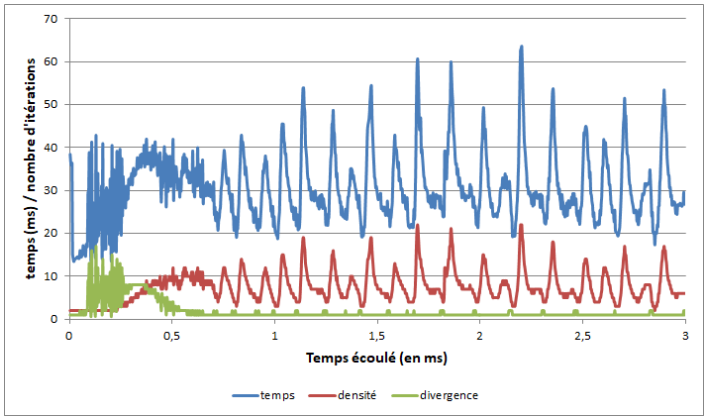
**Figure 5.17.:** Comparaison de nos implémentations GPU 32 bits et GPU 64 bits de l'algorithme DFSPH à l'implémentation CPU parralèle proposée dans la librairie SPlisHSPlasH. Nous comparons le temps de simulation nécessaire aux 1000 premiers pas de simulation d'un scénario de cassage de barrage. (a) Évolution du temps de simulation en fonction du nombre de particules. (b) Évolution du gain de performance de notre implémentation GPU relativement à l'implémentation CPU 64bits



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés




(a)




(b)

**Figure 5.18.:** La figure (a) présente l'évolution du nombre d'itérations des solveurs de densité constante et divergence nulle en fonction du nombre de particules de fluide présentes dans la simulation. La figure (b) permet de visualiser l'évolution de la durée du pas de temps de simulation ainsi que du nombre d'itérations des solveurs de densité constante et divergence nulle en fonction du temps. Le scénario de simulation utilisé est un cassage de barrage. La majeure partie du fluide est au repos après 0.7 secondes de simulation.

constante et de divergence nulle est lié au nombre de particules présentes dans la simulation. Ceci est dû au fait que pour des nombres de particules plus faibles, moins de particules ont un voisinage complet. Par conséquent, le nombre de particules pouvant présenter une surpression devant être corrigée est plus faible. Nous illustrons ce phénomène dans la figure 5.18a, sur laquelle nous pouvons observer que le nombre d'itérations moyen nécessaire à chacun des solveurs augmente linéairement avec le nombre de particules.

Comme nous l'avons évoqué lors de notre étude du démarrage à chaud, le nombre d'itérations nécessaires à chacun des deux solveurs, de densité constante et divergence nulle, dépend de l'état de la simulation. Pour le vérifier nous représentons dans la figure 5.18b l'évolution de la durée du pas de temps de simulation ainsi que du nombre d'itérations de chacun des solveurs en fonction du temps. Pour cette expérience nous utilisons un scénario de cassage de barrage contenant environ 78000 particules de fluide.

Nous remarquons que le nombre d'itérations nécessaires à chacun des solveurs varie bien au cours de la simulation. Ce phénomène est très visible pour le solveur de divergence nulle qui constitue une partie importante des calculs tant que le fluide est en mouvement et ne nécessite en moyenne qu'une seule itération lorsqu'il est au repos.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

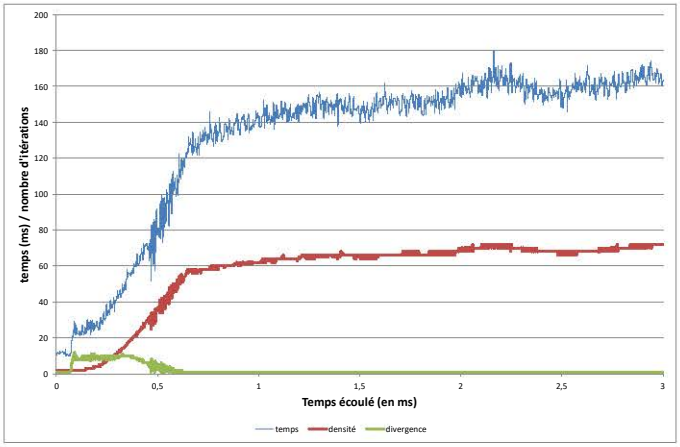



**Figure 5.19.:** Durée du pas de temps de simulation et nombre d'itérations des solveurs de densité constante et divergence nulle en fonction du temps. La simulation n'utilise pas les démarrages à chaud des processus prédictifs-correctifs. Le scénario de simulation utilisé est un cassage de barrage.

### 5.5.4 Stabilité du fluide

Sur la figure 5.18b nous remarquons que le nombre d'itérations nécessaire pour un fluide au repos est très instable. Si l'on regarde de près le nombre d'itérations nécessaires au solveur de divergence nulle, nous observons que à chaque pas de temps précédent l'augmentation du nombre d'itérations du solveur de densité constante, le solveur de divergence nulle nécessite deux itérations alors qu'une seule est observée normalement. Également, si nous regardons le fluide dans sa globalité nous pouvons voir que celui-ci n'est pas parfaitement stable. Il est possible d'observer un cycle composé d'une lente compression du fluide suivi d'une rapide expansion. Notre hypothèse est que ceci est dû au démarrage à chaud. Bien qu'il permette une convergence plus rapide de chacun des solveurs, un nombre d'itérations faible semble causer une moins bonne distribution de la correction au sein du fluide.

Pour vérifier cette hypothèse, nous réalisons un scénario de cassage de barrage en n'utilisant pas de démarrage à chaud. Le nombre d'itérations de chacun des solveurs ainsi que le temps moyen d'un pas de simulation sont représentés sur la figure 5.19. Conformément à notre hypothèse, nous pouvons voir que, sans démarrage à chaud, le nombre d'itérations du solveur de densité constante est stable. Il serait probablement très intéressant d'étudier en détail les limites du système de démarrage à chaud pour vérifier s'il n'est pas possible de supprimer les instabilités que la version actuelle provoque. Notons tout de même que, malgré ces instabilités, nous avons préféré maintenir l'utilisation du démarrage à chaud car le gain de performance associé est très élevé, un facteur de plus de 4. Une solution permettant de diminuer l'amplitude des perturbations introduites par le démarrage à chaud est d'augmenter le nombre d'itérations minimum des solveurs. En particulier, lorsque le fluide est au repos, le solveur de divergence utilise principalement une seule itération. La table 5.7 récapitule le nombre d'itérations moyen du solveur de divergence ainsi que l'écart type pour différents nombres minimaux d'itérations du solveur de divergence lorsque le fluide est au repos. Nous pouvons voir qu'augmenter le nombre d'itérations minimum du solveur de divergence améliore de manière significative la stabilité du solveur de densité. En particulier, utiliser un minimum de 2 itérations permet de diviser par plus de deux l'écart-type. Nous pouvons observer qu'utiliser plus de 5 itérations minimales n'améliore pas l'écart-type davantage. Nous pouvons aussi voir qu'augmenter le nombre minimal d'itérations du solveur de divergence augmente le nombre moyen d'itérations


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

du solveur de densité. Ceci est probablement dû au fait que le fluide étant plus stable la lente compression suivie par la rapide expansion ne sont plus présentes. Le fluide reste donc compressé au maximum et nombre d'itérations de densité nécessaires est plus élevé. La conséquence est une augmentation du temps moyen pour un pas de simulation pour chaque augmentation du nombre minimal d'itérations du solveur de divergence.

| Min iter divergence | 1 | 1+1 | 2 | 2+1 | 3 | 3+1 | 5 | 10 |
|---|---|---|---|---|---|---|---|---|
| Avg iter densité | 7.92 | 8.37 | 8.06 | 9.15 | 8.69 | 9.38 | 9.04 | 9.01 |
| Écart-type iter densité | 3.73 | 1.66 | 1.59 | 1.53 | 1.30 | 1.55 | 1.15 | 1.23 |
| Temps moyen | 33.54 | 35.30 | 37.10 | 36.79 | 39.76 | 38.28 | 46.56 | 56.63 |

**Tableau 5.7.:** Moyenne et écart type du nombre d'itérations du solveur de densité ainsi que le temps d'exécution moyen pour un pas de simulation (en ms) en fonction du nombre minimal d'itérations du solveur de divergence. Les colonnes avec un nombre d'itérations suivi d'un "+1" indiquent des configurations où le démarrage à chaud a été remplacé par une itération minimale supplémentaire. Cette expérience est réalisée avec un fluide au repos sur 1000 pas de simulation.

Étant donné qu'augmenter le nombre d'itérations minimal du solveur de divergence stabilise le solveur de densité, on pourrait supposer que l'inverse est également vrai. Cependant, nous n'avons pas observé de stabilisation de la perturbation du nombre d'itérations du solveur de divergence visible en début de cassage de barrage lorsque nous avons augmenté le nombre d'itérations minimal du solveur de densité. Notons tout de même que si l'on décide de ne pas utiliser le démarrage à chaud pour le solveur de divergence cette perturbation disparaît. Or, le démarrage à chaud de ce solveur nécessite un temps d'exécution similaire à une itération. De plus, pour des nombres d'itérations faibles, réaliser une itération supplémentaire applique probablement des variations de vitesse plus significatives que celles appliquées par le démarrage à chaud. Nos simulations pour lesquelles nous remplaçons le démarrage à chaud du solveur de divergence pour une itération minimale supplémentaire nous permettent de noter deux problèmes pour un fluide au repos. Les résultats sont visibles dans les colonnes "+1" de la table 5.7. Nous pouvons voir qu'avec 3 itérations minimales, le démarrage à chaud apporte plus de stabilité qu'effectuer une itération supplémentaire. Ceci est dû au fait que le démarrage à chaud semble réduire l'intensité d'un bruit très haute fréquence mais de faible amplitude affectant le nombre d'itérations du solveur de densité. En conclusion, nous avons observé dans nos expériences qu'utiliser le démarrage à chaud des deux solveurs avec deux itérations minimales pour le solveur de densité et trois pour le solveur de divergence nous permet d'obtenir une très bonne stabilité du fluide sans calcul superflu pour un fluide au repos.

## 5.6 Discussions

La limitation la plus évidente est que notre système de fenêtre dynamique de simulation ne permet de représenter qu'un océan plat. En particulier notre système ne permet pas de représenter des vagues ou des courants vu qu'il n'y a pas de transferts de particules entre la zone simulée et le reste du fluide. Bien que cela suffise pour les travaux réalisés dans cette thèse, il est important d'améliorer ce système pour augmenter le nombre de scénarios simulables.

Une seconde limite de notre fenêtre dynamique de simulation est due aux déplacements de particules qui sont effectués à chaque fois que la zone de simulation est déplacée. En



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

particulier, ajouter des particules vers l'avant du fluide de manière à dupliquer une tranche de fluide n'est pas la solution la plus stable possible. La solution optimale serait de combler la colonne de fluide vide introduite lors du déplacement des bordures du fluide avec des particules ayant une disposition similaire à un fluide au repos. Cela permettrait d'éviter d'introduire des perturbations au niveau des plans de jonction. De plus, avec un tel système il ne serait pas nécessaire de replacer des particules vers la surface du fluide. Nous avons montré que celles-ci sont la principale source de perturbation dans notre système et leur suppression serait une excellente piste de recherche pour améliorer notre système de fenêtre dynamique de simulation.

La capacité à placer des particules dans un état similaire à un fluide au repos est à notre avis le critère qui permettrait la plus grande amélioration d'un système de déplacement de fenêtre dynamique. En effet, en plus de pouvoir déplacer le fluide existant sans introduire de perturbation, nous aurions la possibilité d'ajouter du fluide dans la simulation et donc de simuler des vagues voire potentiellement des courants. Nous pouvons proposer des pistes de recherche pouvant mener à un ajout stable de particules au repos. La première serait d'utiliser un modèle statistique qui utiliserait des états de simulation pour avoir un modèle de référence pour le placement des nouvelles particules. Une seconde idée serait d'utiliser une simulation locale pour quelques pas de simulation de la zone où les particules ont été ajoutées. Pour cette simulation locale, le reste du fluide serait considéré comme immobile. Cela permettrait aux nouvelles particules d'adapter leurs positions aux particules existantes. Le coût évident de cette seconde solution est une utilisation accrue des ressources de calcul à chaque fois que de nouvelles particules sont ajoutées. Cependant, cette seconde solution est peut-être la méthode la plus directe au problème du placement des nouvelles particules car elle permet aussi de calculer les propriétés des particules telles que les valeurs du démarrage à chaud en plus de la position des particules.

Une autre limitation de notre simulation de fluide est l'introduction du bruit dans le nombre d'itérations du solveur de densité dû au démarrage à chaud. La création d'un système de démarrage à chaud n'introduisant pas de perturbation serait une amélioration particulièrement intéressante de la simulation lagrangienne car le bruit que le démarrage à chaud actuel provoque a probablement des conséquences notables en particulier lorsque le fluide est en interaction avec des objets. Nous n'avons cependant, pas étudié les raisons causant l'apparition de perturbations avec le démarrage à chaud actuel et ne connaissons pas d'autre méthode de démarrage à chaud ayant été utilisées dans des simulations lagrangiennes.

## 5.7 Bilan

Dans ce chapitre nous avons présenté notre simulateur de fluide lagrangien. Nous avons choisi d'implémenter l'algorithme *divergence-free SPH* (DFSPH) proposé par Bender et Koschier [BK15]. Cet algorithme améliore la stabilité du fluide en assurant que la divergence du champ de vitesse des particules reste nulle au cours de la simulation.

Nous avons proposé une implémentation entièrement GPU ne nécessitant aucun transfert vers le CPU libérant ainsi du temps de calcul pour la réalisation d'autres taches en parallèle de la simulation de fluide.

Nous avons proposé une nouvelle méthode permettant le déplacement d'une fenêtre dynamique de simulation. Bien que cette méthode soit actuellement limitée à un déplacement


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

sur un océan plat, nous espérons pouvoir continuer son amélioration pour permettre le déplacement sur un océan généré procéduralement.

Nous avons proposé une comparaison de plusieurs optimisations proposées par des travaux existants. Nous avons remarqué que l'utilisation d'un indice de Morton pour la structure accélératrice permettant la recherche du voisinage d'une particule n'offre pas de meilleure performance qu'un indice linéaire classique, contrairement aux indications des publications précédentes. Également, l'utilisation de tables de correspondance pour déterminer la valeur des fonctions de noyau n'offre un gain de performance que dans le cas d'une implémentation utilisant des réels en double précision.

Nous avons comparé notre implémentation avec celle proposée dans la librairie SPlisHSPlasH. Cette comparaison nous a montré que le facteur d'accélération entre l'implémentation CPU et notre implémentation GPU augmente avec le nombre de particules simulées, allant de $4$ pour 7000 particules à $20$ pour 80 000 particules.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

# 6 Simulation de personnage en interaction avec un fluide

L'un des objectifs de cette thèse est de concevoir un système permettant à un personnage de se déplacer dans un fluide simulé. Nous souhaitons comparer les résultats obtenus par ce système à ceux obtenus avec notre contrôleur précédent ([CPB15]) qui utilise un modèle simplifié pour calculer l'impact du fluide sur le personnage. En particulier nous désirons étudier si les perturbations introduites dans le fluide par le personnage en mouvement ont un impact visible sur les forces du fluide appliquées sur le personnage et par conséquent sur le contrôle du personnage.

Comme nous l'avons présenté précédemment, l'approche la plus intéressante, pour faire interagir de manière réaliste un personnage animé et un fluide, est d'utiliser une méthode de contrôle basée sur la physique pour le personnage et d'utiliser une modélisation du fluide permettant de calculer les forces que le fluide applique sur les membres du personnage à chaque instant de l'animation. Cette approche a été principalement utilisée pour animer des créatures aquatiques simples [TT94 ; Tan+11] ou obtenir des mouvements de natation de personnages bipèdes [YLS04 ; Kwa+10]. La méthode la plus commune pour déterminer l'impact du fluide sur le personnage est de mettre en place un modèle simplifié du fluide. Pour permettre la prise en compte de l'impact qu'a le personnage en mouvement sur l'état du fluide, une simulation du fluide est parfois préférée [Kwa+10 ; Tan+11 ; Si+14]. Notons que le fluide est simulé par un modèle eulérien ou mixte pour pouvoir bénéficier d'un haut niveau de précision physique. Comme nous l'avons précédemment évoqué ces modèles ont comme défaut majeur de nécessiter de larges ressources de calcul. Lorsque le fluide est simulé physiquement, les méthodes visant à simuler la nage d'un personnage nécessitent des temps de simulation d'au minimum plusieurs secondes par pas de simulation même pour des personnages 3D simples [Tan+11]. Nous avons comme objectif la création d'un système au minimum interactif, nécessitant l'utilisation d'une simulation lagrangienne de fluide comme présenté dans la section précédente.

Lorsque le mouvement désiré est un mouvement bipède, un modèle simplifié est généralement utilisé pour déterminer l'impact du fluide sur le mouvement. Lentine et al. [Len+11] simulent la marche d'un personnage soumit à des interactions avec un milieu venteux. Bermudez et al. [Ber+18a] ainsi que nos travaux précédents [CPB15] considèrent un personnage marchant dans un milieu liquide. L'impact du fluide sur le personnage est calculé par des formules permettant de calculer la résistance d'un fluide au mouvement. Notre contrôleur considère également la poussée d'Archimède mais ne considère pas la friction du fluide. À l'inverse, Bermudez et al. prennent en compte les forces dues à la friction mais n'intègrent pas la poussée d'Archimède dans leur modèle. Notons tout de même que Bermudez et al. n'appliquent pas directement les forces du fluide sur les parties du personnage sur lesquelles elles sont produites. Les forces du fluide sont agrégées et appliquées en une force unique sur un modèle réduit du personnage ne comprenant qu'un seul corps rigide situé au niveau du centre de masse du personnage. Par conséquent bien que leur personnage soit ralenti par la présence du fluide, les différents membres du personnage ne sont pas individuellement affectés par la présence du fluide. Par conséquent, nous avons choisi de réutiliser le modèle



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

simplifié du fluide utilisé dans nos travaux précédents comme point de comparaison avec notre fluide simulé.

## 6.1 Estimation du fluide par modèle simplifié

Cette section présente le modèle simplifié que nous avions utilisé pour estimer l'impact du fluide sur le mouvement du personnage [CPB15]. Ce modèle simplifié décompose l'impact du fluide en deux composantes : la poussée d'Archimède et la résistance du fluide au mouvement. L'équation de la poussée d'Archimède est la suivante :

$$\vec{F_A} = \rho V \vec{n} \tag{6.1}$$

Avec $\rho$ la densité du fluide, $V$ le volume de fluide déplacé et $\vec{n}$ le vecteur unitaire vertical. Bien que cette formule semble simple, estimer le volume immergé est difficile pour des objets n'étant pas représentés par des formes géométriques simples. Une méthode simple permettant d'estimer le volume immergé pour des formes non triviales est d'utiliser une représentation sous forme de voxels. Le volume immergé peut être estimé par la somme du volume des voxels immergés.

L'équation permettant de calculer la résistance du fluide au mouvement est différente suivant si le flot est laminaire, c.à.d. la direction du déplacement du fluide est identique en tout point, ou turbulent (i.e. non laminaire). Dans notre cas d'application, où la vitesse de déplacement est à l'échelle humaine et le fluide possède des propriétés proches de l'eau, le flot est considéré turbulent. Par conséquent, nous utilisons l'équation suivante pour calculer la résistance au mouvement :

$$\vec{F_D} = \frac{1}{2}\rho v^2 (C_D A + C_F S) \tag{6.2}$$

Avec $v$ la vitesse du fluide, $C_D$ et $C_F$ des constantes dépendant respectivement de la forme de l'objet et de sa friction avec le fluide. $S$ représente la surface de l'objet et la valeur $A$ correspond à la surface obtenue par projection orthogonale dans un plan perpendiculaire au déplacement du fluide. L'utilisation directe de cette équation demande de connaître les coefficients $C_D$ et $C_F$. Ces coefficients sont généralement déterminés à travers des expérimentations empiriques. Dans un premier temps étudions la partie de l'équation ne dépendant pas de la friction du fluide (i.e. $C_F = 0$). Pour des objets n'étant pas symétriques selon tous les axes, $C_D$ dépend de l'orientation de l'objet. Les objets de notre simulation changent d'orientation à chaque pas de simulation. Il est donc inenvisageable de déterminer $C_D$ empiriquement . Au cours de nos simulations, nous avons fixé le coefficient de forme $C_D$ à 1, ce qui correspond à la valeur moyenne pour une personne debout, ou à une forme cylindrique (e.g. cable).

Cette équation considère que la vitesse de l'objet relative au fluide est la même en tout point de l'objet. Dans nos simulations, les membres du personnage sont en rotation et par conséquent, la vitesse relative entre le fluide et l'objet n'est pas constante en tout point de l'objet. Pour résoudre ce problème, chacun des objets est découpé en sous-parties sur lesquelles la vitesse relative au fluide est considérée constante à un instant ce qui nous permet de déterminer le plan de projection de chaque partie. Nous pouvons ainsi appliquer la formule 6.2 sur chacune


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

des parties ce qui nous donne un champs de forces représentant l'impact du fluide sur l'objet étudié.

La dernière difficulté correspond à la partie de l'équation que nous n'avons pas encore considérée : la résistance due à la friction. Nous ne sommes pas arrivés à aboutir à une équation correcte pour représenter la friction car celle-ci est principalement étudiée pour des formes extrêmement aérodynamiques telles que des ailes d'avion. Dans un flot turbulent, à moins que le coefficient de forme $C_D$ soit très faible, la résistance due à la friction n'apporte qu'une contribution négligeable aux forces totales. Cependant, une formule adaptée à un flot laminaire pour l'étude d'objets simples dans un flot turbulent est parfois utilisée. Les forces de résistance au mouvement calculées par les formules de flot laminaire ne varient que linéairement relativement à la vitesse du fluide et par conséquent la force due à la friction apparaît comme négligeable. Bien que nous ne soyons pas arrivés à déterminer une méthode de calcul exacte permettant de calculer les forces de friction, nous nous sommes appuyés sur le fait que ces forces semblent être négligeables pour les retirer de notre modèle simplifié.

L'équation utilisée pour calculer les forces de résistance au mouvement du fluide dans notre modèle simplifié est donc la suivante :

$$\vec{F_D} = \frac{1}{2} C_D \rho \sum_i (v_i^2 A_i) \tag{6.3}$$

Avec $C_D = 1$ et $i$ les différentes parties du solide après le découpage dû à la contrainte d'avoir une surface perpendiculaire au déplacement du fluide. Chaque force ayant un point d'action différent $P_i$, il est nécessaire d'utiliser un bras de levier pour déterminer le moment induit par chacune d'entre elles lorsqu'elles sont appliquées sur le centre de masse du corps rigide $P_{COM}$.

$$\vec{M_D} = \sum_i (P_{COM} - P_i) \otimes (F_{D\_i}) \tag{6.4}$$

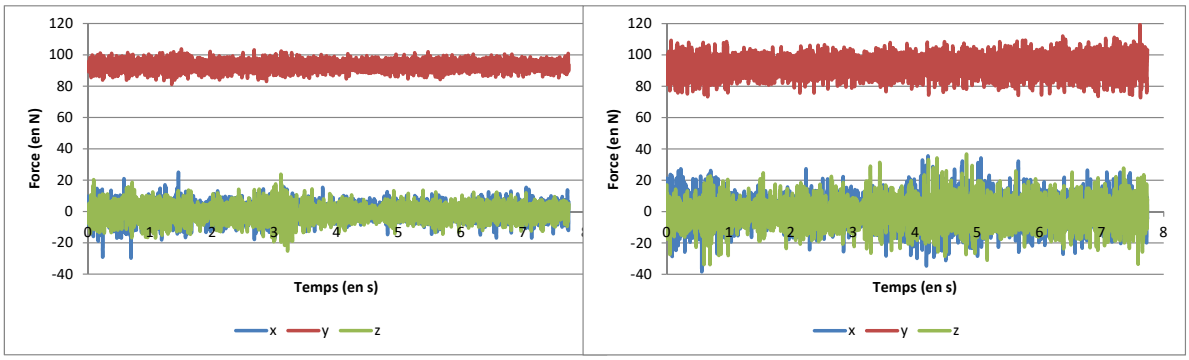
Avec $F_{D\_i} = \frac{1}{2} 0.8 \rho v_i^2 A_i$. Ceci nous permet de ne pas avoir à transmettre un nombre important de forces au moteur physique ce qui pourrait avoir comme effet un ralentissement des calculs pour le pas de simulation suivant.

## 6.2 Simulation de fluide lagrangienne en interaction avec des objets

Pour le fluide simulé, nous utilisons notre simulateur de fluide lagrangien présenté dans la section précédente (section 5). Nous utilisons le système de déplacement de fenêtre de simulation dynamique que nous avons présenté dans la section 5.4 pour permettre au personnage d'être en permanence en contact avec le fluide. Avant d'étudier les interactions entre le personnage et le fluide, il est important d'étudier les interactions entre le fluide et un corps rigide.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés




(a) (b)

**Figure 6.1.:** Évolution de la résultante des forces appliquées par le fluide sur le solide au cours du temps. Les figures a et b correspondent à une simulation utilisant respectivement 7 et 3 itérations minimales pour le solveur de divergence.

### 6.2.1 Stabilité des forces du fluide

Comme nous l'avons remarqué à la fin de la section relative aux résultats de notre simulation de fluide (section 5.5.4), les nombres d'itérations des solveurs de densité et de divergence ne sont pas constants. Nous avons vu que cette instabilité provoque, lorsqu'elle est élevée, un mouvement de compression et d'expansion régulier du fluide. Il est probable qu'un tel mouvement provoque une variation des forces appliquées par le fluide sur un solide en interaction avec le fluide. Nous avons vu qu'il est possible de réduire ce mouvement d'expansion et de compression en augmentant le nombre minimal d'itérations du solveur de divergence. Cependant, à moins d'utiliser un nombre minimal d'itérations supérieur à 5, il est toujours possible d'observer des variations du nombre d'itérations du solveur de densité, ce qui nous indique la présence de bruit dans notre simulation. Nous avions précédemment conseillé d'utiliser au minimum deux itérations pour le solveur de densité et trois pour le solveur de divergence. Cette section a pour but de vérifier si cette configuration permet d'obtenir des forces stables ou si une étape de lissage des forces est nécessaire.

Vérifions en premier si les variations du nombre d'itérations des solveurs ont bien un impact sur la stabilité du fluide. Pour ce faire, nous étudions la variation au cours du temps des forces sur un objet immobile entièrement immergé dans un fluide au repos. Les figures 6.1a et 6.1b représentent l'évolution de la résultante des forces du fluide sur le solide avec respectivement 7 itérations minimales du solveur de divergence, ce qui rend le fluide quasiment parfaitement stable, et 3 itérations minimales. Si nous comparons l'amplitude de la variation des forces entre la simulation utilisant 7 itérations minimales et celle utilisant 3 itérations minimales, nous pouvons constater que l'utilisation de plus d'itérations du solveur de divergence diminue le bruit dans les interactions avec un solide. Cependant, même avec 7 itérations minimales nous observons un bruit non négligeable dans les forces appliquées par le fluide alors que le bruit sur le nombre d'itérations du solveur de densité est faible, comme nous l'avons vu dans la table 5.7 lors de l'étude de notre simulateur de fluide. Cela veut dire qu'il existe du bruit indépendamment du nombre d'itérations utilisé par les solveurs de la simulation du fluide. Il n'est pas réaliste d'utiliser des grands nombres d'itérations minimales car cela augmenterait fortement le temps nécessaire pour chaque pas de simulation (table 5.7).

Une solution permettant d'augmenter la stabilité de la force est d'appliquer la moyenne des forces observées sur les derniers pas de simulation au lieu d'appliquer directement la force prévue pour le pas de simulation en cours. Cette solution échange de la réactivité contre de la


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

| Nb pas de simulation | 1 | 2 | 3 | 4 | 5 | 7 | 10 |
|---|---|---|---|---|---|---|---|
| Ecart-type | 8.21 | 6.31 | 4.95 | 3.97 | 3.46 | 2.72 | 1.73 |
| Diminution relative (en %) | | 23.07 | 21.66 | 19.68 | 12.93 | 10.66 | 12.10 |

**Tableau 6.1.:** Ecart-type de la force appliquée par un fluide sur un solide. La force considérée est une moyenne temporelle sur un nombre de pas de simulation indiqué sur la première ligne. La ligne "diminution relative" indique la diminution de l'écart-type par rapport à une moyenne utilisant un pas de temps de moins. L'écart-type est calculé sur 2500 pas de simulation.

stabilité. Nous avons évalué le gain de stabilité, par l'intermédiaire de l'écart-type, en fonction du nombre de pas de simulation utilisés pour calculer la moyenne (table 6.1). Nous pouvons voir qu'utiliser une moyenne diminue fortement l'écart-type. Cependant, nous observons une réduction de la diminution relative de l'écart-type entre les simulations utilisant 4 ou moins pas de temps pour la moyenne et les simulations utilisant 5 pas de temps ou plus. Ceci s'explique par le fait qu'il y a deux bruits différents. Le premier est un bruit très haute fréquence qui est dû aux légères variations entre deux pas de temps. C'est le bruit dont la correction permet une amélioration importante de l'écart-type avec une moyenne sur 4 ou moins pas de simulation. Le second bruit est plus basse fréquence, environ 20Hz dans nos simulations, et correspond aux variations du nombre d'itérations du solveur de densité. La figure 6.2a illustre l'évolution de la force appliquée par le fluide sur le solide sur l'axe vertical ainsi que les deux bruits que nous avons mentionné. Considérons séparément les composantes (x, y, z) de la force appliquée par le fluide et étudions leurs écarts-type (figure 6.2b). Nous pouvons remarquer que son évolution avec le nombre de pas de simulation utilisés pour calculer la moyenne n'est pas identique sur tous les axes. L'écart-type sur l'axe vertical est moins élevé avec une moyenne utilisant peu de pas de simulation (moins de 3) mais diminue moins vite avec l'augmentation du nombre de pas utilisés pour calculer la moyenne. À partir de 4 pas de simulation ou plus, la diminution de l'écart-type semble devenir identique. La différence majeure entre les trois axes est que, le fluide et le solide étant au repos, les forces n'étant pas sur l'axe vertical (y) contiennent uniquement du bruit. L'écart-type initialement plus faible sur l'axe y semble indiquer que les composantes de la force pour lesquelles il n'y a pas d'amplitude significative présentent plus de bruit blanc. Ce bruit étant principalement retiré dès que nous avons plus de 4 pas de simulation dans la moyenne, cela explique pourquoi le taux de diminution de l'écart-type est plus fort sur les axes x et z pour des moyennes utilisant peu de pas de simulation. Au cours de nos expérimentations, nous avons observé que le bruit blanc élevé n'est pas uniquement observable sur les composantes pour lesquelles la résultante des forces est nulle, mais sur toutes les composantes sur lesquelles la force ne représente pas une partie significative de la force totale.

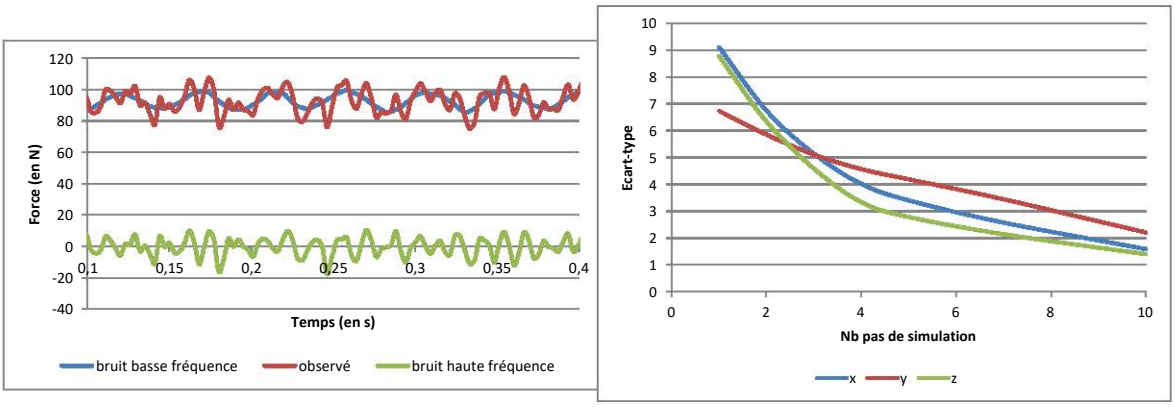
La fréquence du bruit dû à la variation du nombre d'itérations des solveurs étant élevée, il n'est pas raisonnable d'annuler le bruit avec une moyenne. En effet, cela diminuerait fortement la réactivité des solides aux modifications de la simulation de fluide. Nous avons donc choisi d'utiliser une moyenne sur 4 pas de simulation, ce qui retire la majeure partie du bruit blanc haute fréquence.

### 6.2.2 Intensité des forces du fluide

Le but de cette section est de vérifier que les forces appliquées par le fluide sur un solide correspondent bien aux forces théoriques pour un cas simple. Pour ce test, nous immergeons des solides pour lesquels le volume est connu et relevons la force moyenne appliquée par

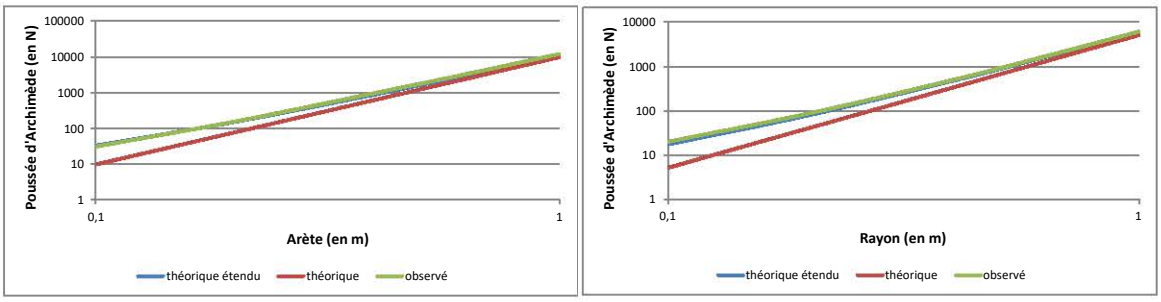


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés




(a) (b)

**Figure 6.2.:** (a) Force appliquée par le fluide sur l'axe vertical et bruits présents. La courbe verte représente le bruit haute fréquence, et la courbe bleue visualise le bruit basse fréquence observé quand le bruit haute fréquence est retiré.( b) : Évolution de l'écart-type de la force appliquée par un fluide sur un solide en fonction du nombre de pas de simulation utilisé pour calculer la force moyenne pour chaque composante.




(a) (b)

**Figure 6.3.:** Évolution en fonction de la taille du corps rigide de la poussée d'Archimède théorique et de la poussée d'Archimède observée dans notre simulation lorsque le fluide et le corps rigide sont au repos. Les objets solides pour les figures 6.3a et 6.3b sont respectivement un cube et une sphère, l'axe des abscisses indiquant leur dimension (longueur d'arête et rayon). Les particules sont positionnées selon une distribution de Poisson à la surface de l'objet. Les courbes des valeurs théoriques étendues correspondent à la poussée d'Archimède théorique pour un objet dont la dimension a été légèrement étendue de manière à ce que les particules (y compris leur rayon) soient inclues dans le volume de l'objet.

le fluide lorsque la simulation est au repos. De cette manière nous pouvons évaluer la poussée d'Archimède théorique et la comparer aux forces observées. Pour les forces observées dans la simulation nous relevons une moyenne sur 100 pas de temps pour s'assurer que le bruit présent dans le fluide, que nous avons étudié dans la section précédente, n'a pas d'impact sur la force relevée. Les objets que nous utilisons pour cette comparaison sont des cubes et des sphères avec différentes dimensions. Pour déterminer la position des particules représentant l'objet nous utilisons la discrétisation présente dans la librairie SPlisHSPlasH. Cette discrétisation dispose les particules sur la surface des objets selon une distribution de Poisson.

Comme nous pouvons le voir sur les figures 6.3a et 6.3b il y a une assez grande disparité entre la valeur théorique et la valeur observée, en particulier pour des objets de petite taille pour lesquels la force observée est plus de 3 fois supérieure à la force théorique. La raison est que le volume représenté par les particules ne correspond pas au volume désiré. En


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

effet, les particules ayant un rayon, si elles sont positionnées à la surface de l'objet, alors les particules représentent en pratique un objet plus large que celui désiré. Pour vérifier cette hypothèse nous avons également représenté sur les figures 6.3a et 6.3b les poussées d'Archimède théoriques pour les mêmes objets dont la dimension aurait été augmentée par le diamètre des particules (0.05 mètres dans cette simulation). Nous pouvons voir que cette nouvelle courbe est bien plus proche des valeurs observées dans la simulation avec une erreur maximale de $15\%$ pour les sphères et $10\%$ pour les cubes, alors que cette erreur maximale était supérieure à $300\%$ dans les deux cas avec nos valeurs théoriques précédentes.

La conclusion que nous retenons de cette expérience est que pour représenter le volume d'un objet dans une simulation SPH, il faut que les particules le représentant soient disposées sous la surface de celui-ci et non pas sur la surface. Cependant, il est possible que le choix de placer les particules sur la surface soit justifié par le fait qu'une telle configuration génère des forces de résistance au mouvement réaliste. Malheureusement, il ne nous est pas possible de vérifier cette hypothèse. Les particules étant placées sur la surface des objets dans la librairie SPlisHSPlasH, il est possible que cette hypothèse soit vraie. Par conséquent, une solution consiste à appliquer d'un facteur de diminution des forces du fluide sur les solides au lieu de placer les particules à l'intérieur des objets. Cependant, cette solution ne fonctionne que si les objets présents dans la simulation sont indépendants les uns des autres. En effet, avec cette méthode le rayon des particules dépasse de la bordure des objets. Si deux objets sont proches nous observerons donc une superposition des objets. Par conséquent, certaines des particules des objets ne seront plus en contact avec le fluide. Les conséquences de la présence de multiples objets proches sont étudiées dans la section 6.2.3.

La solution que nous avons choisie est de générer les particules à l'intérieur des corps rigides. Cependant, les dimensions du pied de notre personnage ne permettent pas d'utiliser cette approche. En effet, les pieds sont représentés par un pavé rectangulaire de dimensions 5 x 3,8 x 14,5 (en cm). Les particules ayant un rayon de 2,5 cm il n'est pas possible de placer des particules de manière à ce qu'elles soient contenues dans le pavé. La solution que nous avons choisie est de représenter le pied par une ligne de particules en acceptant que les particules dépassent légèrement du pavé. La table 6.2 répertorie les forces observées et les forces théoriques pour les différentes parties de la jambe du personnage lorsqu'elles sont placées séparément dans un fluide au repos. Les forces observées sur les deux parties principales de la jambe correspondent bien aux forces théoriques. Bien que l'erreur sur les forces observées pour le pied soit élevée, ceci est en grande partie dû au manque de précision de la modélisation de celui-ci.

| Membre | cuisse | tibia | pied |
|---|---|---|---|
| Force théorique | 34.4 | 29.8 | 9.7 |
| Force observée | 35.3 | 31.7 | 2.7 |

**Tableau 6.2.:** Poussée d'Archimède (en N) observée sur chacun des membres du personnage lorsqu'ils sont placés dans un fluide au repos et représentés par des particules placées à l'intérieur du volume de chaque membre.

### 6.2.3 Interaction avec des objets proches

Dans cette section nous allons étudier les conséquences sur les interactions entre le fluide et des objets si ces derniers se trouvent proches les uns des autres. Nous nous intéressons à ces interactions car au niveau des articulations de notre personnage articulé, les deux membres reliés sont presque en contact. Les membres du personnage ne se déplaçant pas relativement



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

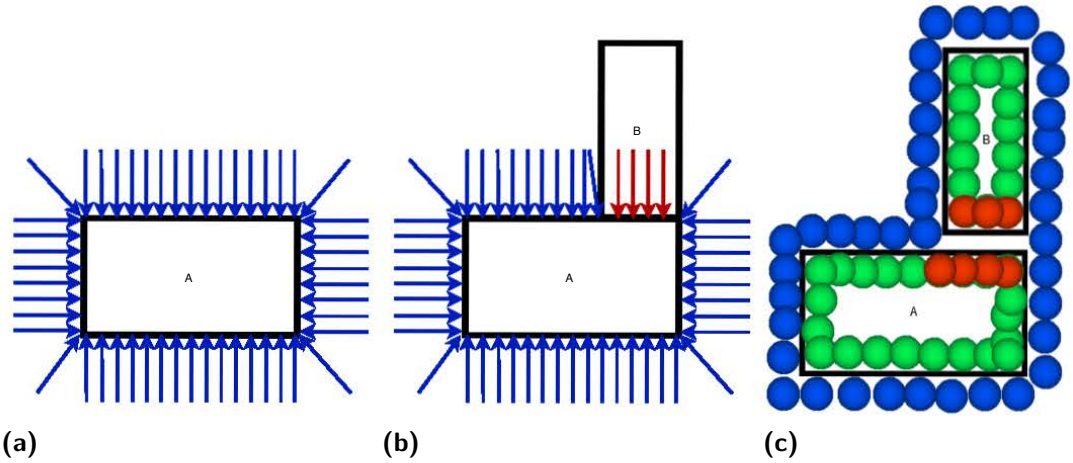



**Figure 6.4.:** (a) Représentation des forces de pression autour d'un objet immergé A. (a) Représentation des forces de pression lorsqu'un second objet B se trouve en contact avec l'objet A (en rouge les forces manquantes sur A dues à la présence de B). (c) Exemple de scénario d'objets proches avec un modèle lagrangien. Bleu : particules de fluide. Vert : particules de solide en contact normal avec le fluide. Rouge : particules de solide pour lesquelles l'objet proche empêche un contact normal avec le fluide.

aux articulations auxquels ils sont attachés, les membres du personnage sont maintenus proches, deux à deux, les uns des autres tout au long de la simulation.

Si l'on considère deux objets immobiles dans un fluide au repos, l'impact du fluide sur chacun des objets devrait correspondre strictement à leur poussée d'Archimède. Par conséquent, la force appliquée par le fluide sur chaque objet devrait rester identique, peu importe la position des objets à l'intérieur du fluide. Cependant, si l'on considère un fluide simulé par une méthode lagrangienne, les forces appliquées par le fluide simulé sur chaque objet peuvent ne pas correspondre à leur poussée d'Archimède. Ceci s'explique par le fait que la force à appliquer sur un objet correspond à l'intégrale des forces, au niveau de sa surface, dues à la pression du fluide (figure 6.4a). Si une partie de la surface n'est pas en contact avec le fluide, alors il n'y aura pas de force de pression sur la partie en question (figure 6.4b). Dans un fluide réel il serait nécessaire d'avoir un contact parfait entre plusieurs objets pour que la présence d'un objet ait un impact sur les forces de pression observées par les autres objets. Cependant, dans un fluide simulé par une méthode lagrangienne, les forces de pression sont dues à la présence des particules de fluide. Il suffit donc que les objets soient suffisamment proches pour empêcher la présence de particules de fluide entre deux objets pour observer une absence de forces de pression sur certaines parties de la surface des objets (figure 6.4c). Cela veut dire que, même dans un système au repos où les objets ne créent pas de déplacement du fluide, la distance relative entre les différents objets de la simulation et la taille des particules ont un impact sur la force appliquée par le fluide sur les objets. Un manque de particules à la surface d'un objet a pour conséquence de changer l'intensité et la direction de la poussée d'Archimède de cet objet. Cela a pour conséquence que la force appliquée par le fluide sur les objets dans notre simulation ne correspond pas forcément à une force strictement verticale et son intensité ne correspond pas forcément au poids du volume de fluide déplacé, même lorsque le fluide est au repos.

Bien que la force appliquée sur chaque objet dépende des zones en contact avec les autres objets, la somme des forces appliquées sur les différents objets correspond approximativement à la somme des poussées d'Archimède de chacun des objets. On peut facilement le visualiser


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

en considérant un objet unique qui serait l'agrégation des objets présents dans la simulation. Cet objet a un volume correspondant à la somme des volumes des objets le composant et il n'y a aucun autre objet présent dans la simulation. Par conséquent, toute la surface de ce nouvel objet est en contact avec du fluide et la force appliquée par le fluide correspond à la poussée d'Archimède. Notons que la force sur le nouvel objet sera légèrement plus grande que la somme des forces sur ces composants. En effet, il aura probablement une surface comprenant des cavités trop petites pour que les particules de fluide y rentrent et par conséquent le volume effectivement observé dans la simulation sera probablement légèrement supérieur à la somme des volumes des objets le composant. Pour résumer, lorsque l'on a un groupe d'objets proches les uns des autres, la force due à la poussée d'Archimède sur chaque objet n'est pas forcément verticale mais la somme des forces observées correspond bien à la somme des poussées d'Archimède observée lorsque les objets sont isolés.

Ce phénomène de redistribution de la poussée d'Archimède limite l'approche consistant à discrétiser des objets en plaçant des particules à leur surface associée à un facteur de réduction pour que les forces appliquées correspondent au bon volume. En effet, nous avons vu que le facteur dépend de la relation entre le volume de l'objet et le diamètre des particules. dans une chaîne de corps rigides, la poussée d'Archimède appliquée sur chaque objet de la chaîne ne dépend pas uniquement du volume de l'objet mais dépend également du volume des autres objets ainsi que de l'orientation des différents objets composant la chaîne. Si tous les objets n'ont pas le même facteur de réduction (i.e. la même forme géométrique), il est impossible de déterminer quel facteur de réduction doit être appliqué sur la force appliquée par le fluide sur l'un des objets de la chaîne.

### Objet proche des limites du fluide

Dans une simulation de fluide lagrangienne, les bordures du fluide sont représentées dans la simulation par un ensemble de particules solides, correspondant à une boite rectangulaire englobant nos simulations. Par conséquent, si un objet s'approche des bordures de notre espace de simulation, certaines parties de l'objet peuvent ne pas être en contact avec des particules de fluide. Dans cette section nous étudions les forces appliquées par le fluide sur un objet lorsque celui-ci est proche des bordures du fluide.

La première chose que nous constatons est que si l'on définit le sol de la simulation des objets rigides et le sol de la simulation du fluide comme le même plan, alors les forces appliquées par le fluide sur un solide au repos sur le sol ne représenteront pas une interaction réelle. Sur la figure 6.5 nous pouvons voir que si l'on a une boite en interaction avec le fluide au repos sur le sol, la résultante des forces appliquées par le fluide sur la boite a une composante verticale négative. Ceci est dû au fait qu'il n'y a aucune particule de fluide en sous la boite solide. Ce résultat n'est pas désiré dans une telle simulation car la résultante des forces appliquées par le fluide correspond à la poussée d'Archimède et devrait avoir une composante verticale positive. Une solution simple à ce problème consiste simplement à définir le sol de la simulation de fluide plus bas que celui de la simulation de solides. De cette manière il y aura toujours des particules de fluide en dessous des solides. Ce problème est également observable au niveau des bordures latérales du fluide. Cependant, le personnage étant maintenu au centre du volume de fluide dans nos simulations par notre fenêtre dynamique de simulation, il n'a pas été nécessaire de modifier les bordures latérales du fluide



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

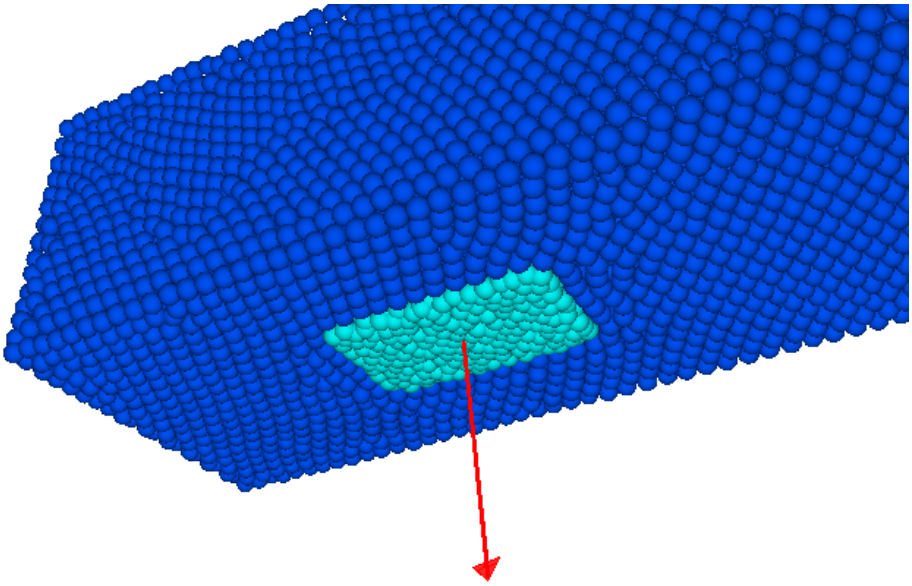


**Figure 6.5.:** Exemple de simulation de fluide où une boite rectangulaire est au repos sur le sol. La caméra est placée sous le fluide. La direction de la force appliquée par le fluide sur le solide est visible par la flèche rouge.

## 6.3 Résultats

### 6.3.1 Comparaison entre le fluide simulé et le modèle réduit

Dans un premier temps, il est intéressant de comparer les différences dans l'impact du fluide sur des corps rigides observées entre le modèle de fluide simplifié et le fluide simulé. L'objectif de ces premières expériences est de déterminer si l'impact d'un fluide simulé physiquement présente des forces qui ne sont pas représentées dans le modèle simplifié. Pour cette comparaison nous estimerons l'impact du fluide sur les jambes du personnage au cours d'un mouvement de marche. Cependant, nous ne ferons qu'évaluer l'impact du fluide et n'appliquerons pas les forces obtenues sur le personnage. Ceci nous permettra d'obtenir l'impact du fluide pour avec un mouvement proche de notre cas d'application final tout en s'assurant que le mouvement utilisé est identique lors de l'évaluation de l'impact du fluide pour les deux modèles.

Dans le modèle simplifié du fluide, les différents membres du personnage sont considérés comme indépendants les uns des autres. Hors comme nous l'avons expliqué précédemment, les membres de la jambe étant proches les uns des autres, l'impact du fluide simulé sur un des membres de la jambe est affecté par la présence des membres voisins. C'est pourquoi la comparaison du modèle simplifié de du fluide simulé est effectué en trois étapes successives. Pour ces trois expériences, nous relevons l'évolution des forces appliquées par le fluide sur le personnage au cours d'un cycle de marche (c.à.d. une phase de support suivie d'une phase de vol). Nous relevons les valeurs sur 25 cycles consécutifs et calculons l'évolution moyenne des forces au cours du cycle. Ceci nous permettra de réduire les perturbations des valeurs dues au bruit basse fréquence présenté dans la section 6.2.1 ainsi que dues à notre système de fenêtre dynamique. En premier, nous n'incluons dans la simulation du fluide que le tibia d'une des deux jambes. Cela nous permet de comparer les deux modèles lorsque le corps rigide étudié est isolé. Dans un second temps, nous incluons dans la simulation du fluide


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

le tibia d'une jambe et à tous les membres de la seconde jambe. Le but de cette seconde expérience est de déterminer si la présence de la seconde jambe a une influence sur l'impact du fluide observé par le tibia. Enfin, dans une dernière expérience, nous inclurons dans la simulation les particules correspondant aux membres d'une seule jambe et étudierons l'impact du fluide sur chacune des parties pour déterminer l'amplitude des modifications (e.g. forces supplémentaires) dues à l'utilisation de plusieurs corps rigides proches les uns des autres. Pour cette dernière expérience nous comparerons l'impact du fluide simulé sur les membres de la jambe lorsque toutes les membres sont présents dans une simulation de fluide unique avec l'impact observé lorsque chacun des membres de la jambe est simulé dans une simulation séparée. Cette expérience a pour but d'étudier l'impact de la proximité des corps rigides dans notre fluide lagrangien. Rappelons que ces trois expériences ont pour but d'étudier des comportements particuliers. Les expériences incluant la totalité des membres ds jambes du personnage sont présentées dans une section ultérieure.

#### Interaction entre fluide et solide isolé

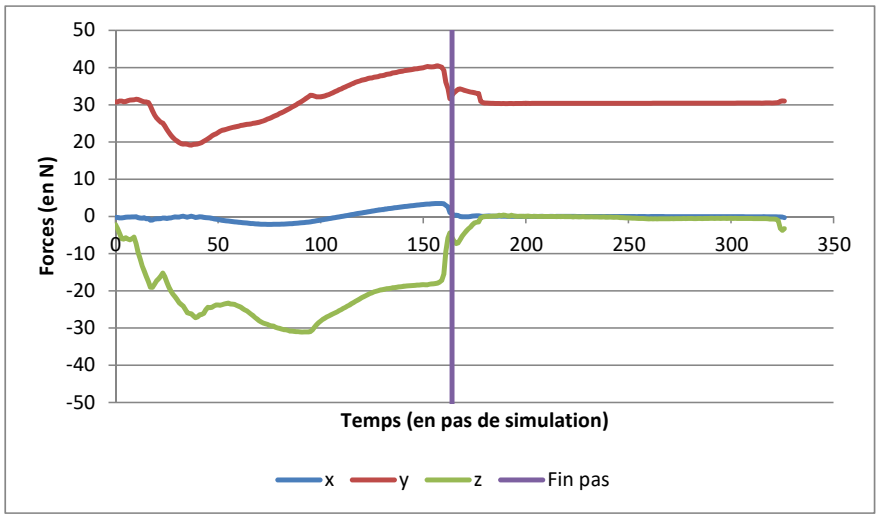
Dans notre première expérience nous étudions l'évolution de l'impact du fluide sur le tibia du personnage lorsque c'est le seul membre du personnage inclus dans la simulation de fluide. Cette expérience a pour but de comparer le fluide lagrangien au fluide simplifié dans la configuration supposée lors de la conception du modèle du fluide simplifié. Les figures 6.6a et 6.6b représentent l'évolution de la résultante des forces appliquées par le fluide sur le tibia avec respectivement le modèle réduit du fluide et le fluide simulé. La différence la plus notable est que l'intensité des forces sur les axes x et z, respectivement les axes coronal et sagittal, est bien plus élevée, entre 3 et 4 fois plus élevés, avec la simulation lagrangienne. Nous avons noté que lorsqu'il n'y a pas de mouvement de l'objet selon un axe, par exemple selon les axes x et y de la phase d'appui, les forces sont identiques entre les deux configurations. Par conséquent, il est apparent que le fluide simulé applique des forces de résistance au mouvement bien plus élevées que le modèle simplifié.

Nous avions d'abord supposé que cette différence est due à la présence de calculs de friction dans le modèle lagrangien alors que le modèle simplifié ne prend pas en compte la friction du fluide. Cependant, nous observons que l'amplitude des forces appliquées par le fluide sur le tibia reste identique même lorsque l'on utilise un coefficient de friction de zéro pour le fluide lagrangien (figure 6.7). Ce résultat pouvait être prédit car le modèle de friction utilisé dans notre simulation lagrangienne ne fait qu'ajouter des forces supplémentaires au niveau des interactions entre les particules de fluide sans jamais prendre en compte la présence des particules de solide.

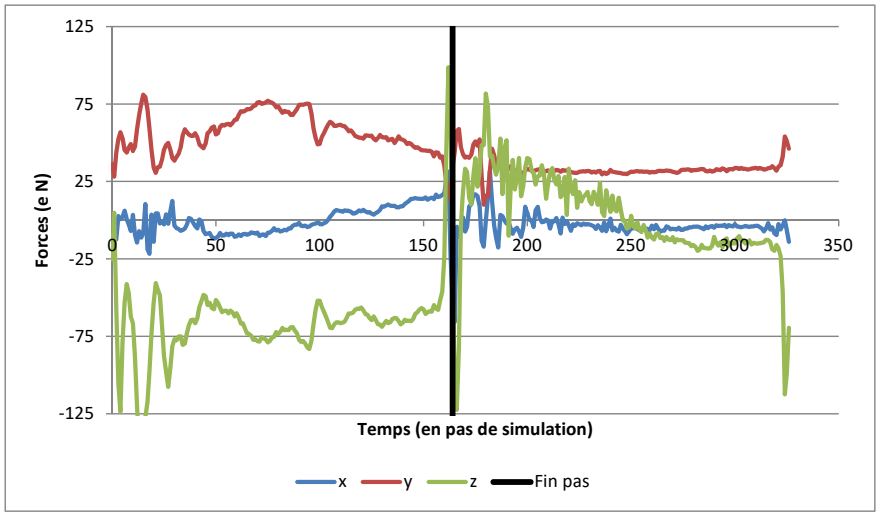
Une seconde explication possible pourrait être l'utilisation d'un coefficient de forme $C_D$ (équation 6.3) non adapté aux primitives géométriques représentant les jambes du personnage lors de l'application de la formule du modèle simplifié calculant la résistance au mouvement. Pour vérifier cette hypothèse, nous pouvons déplacer une primitive simple dont nous connaissons le coefficient de forme. Pour nous assurer que le modèle simplifié est employé dans un cas simple, l'objet ne doit pas être en rotation et son déplacement doit être linéaire. Pour notre expérience l'objet est entièrement immergé pour nous assurer qu'il n'y a pas d'effet de bord. Le coefficient de forme $C_D$ utilisé est de 1. La vitesse de déplacement de l'objet accélère au cours de l'expérience et nous relevons l'évolution en fonction de la vitesse du rapport entre la force calculée par le modèle simplifiée et la force appliquée pour le fluide simulé (figure 6.8). Avant d'analyser les résultats, notons que le coefficient de forme pour une sphère



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés




(a)




(b)

**Figure 6.6.:** Évolution de l'estimation de la résultante des forces appliquées par le fluide sur le tibia du personnage au cours d'un cycle de marche ; (a) : modèle simplifié ; (b) : simulation lagrangienne. Le corps étudié est le tibia qui se trouve dans la jambe en phase de vol au début du cycle de marche.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

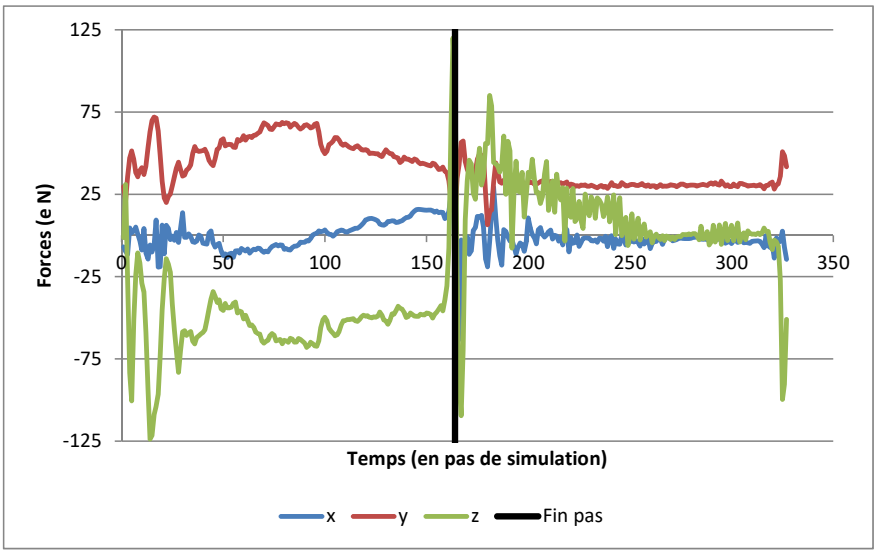



**Figure 6.7.:** Évolution de l'estimation de la résultante des forces appliquées par le fluide sur le tibia du personnage au cours d'un cycle de marche. Le fluide est simulé par notre simulateur lagrangien avec un coefficient de friction de zéro.

est de 0.5 et que celui d'un tube infini est que 1.0 1.3. Par conséquent, le coefficient $C_D$ que nous utilisons est de 0.5 pour les simulations utilisant une sphère et 1 pour les expériences utilisant des capsules. De cette manière, si les forces de résistance au mouvement du fluide simulé correspondent bien à un flot turbulent nous devions observer un rapport constant d'environ 1 entre les forces du fluide simulé et celles du fluide simplifié. Or, dans la figure 6.8, nous observons que le rapport entre les forces des deux modèles n'est même pas constant. Le rapport des forces augmente progressivement jusqu'à une vitesse seuil à partir de laquelle il devient constant. Notons que la vitesse seuil et la valeur du rapport une fois la vitesse seuil atteinte dépendent de la taille de la sphère utilisée. Une augmentation du rayon de la sphère entraîne une vitesse seuil plus élevée et un rapport après le seuil également plus élevé. Notons également que pour les rayons de 0.1m et 0.3m le rapport des forces une fois le seuil atteint (respectivement 0.9 et 1.1) est proche de la valeur théorique, qui est de 1. Cependant, pour la sphère utilisant une dimension similaire aux primitives composant le personnage ce rapport n'est que d'environ 0.4. Nous pouvons établir deux conclusions à partir de ces observations :

— L'augmentation du rapport en fonction de la vitesse nous indique que pour les vitesses jusqu'à la vitesse seuil le fluide n'est pas dans un état turbulent autour de l'objet dans la simulation lagrangienne. En effet, si cela était bien le cas alors nous aurions relevé un rapport constant. Le résultat actuel semble nous indiquer la présence d'au minimum une phase où le fluide applique une force de résistance au mouvement pour un fluide laminaire et probablement la présence d'une phase de transition entre les deux états.

— La présence d'un rapport proche de 1, valeur du rapport théorique, entre les forces du modèle simplifié et les forces simulées semble indiquer que pour des objets avec des dimensions suffisamment grandes relativement à la taille des particules le fluide lagrangien applique des forces de résistance au mouvement réalistes. Cependant, la présence d'un rapport inférieur à 0.5 pour la sphère avec un rayon correspondant à la dimension principale des formes géométriques utilisées pour représenter les jambes du personnage, indique que le fluide applique des forces plus de deux fois trop élevées si la dimension de l'objet est trop proche de la taille des particules. Nous noterons également



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

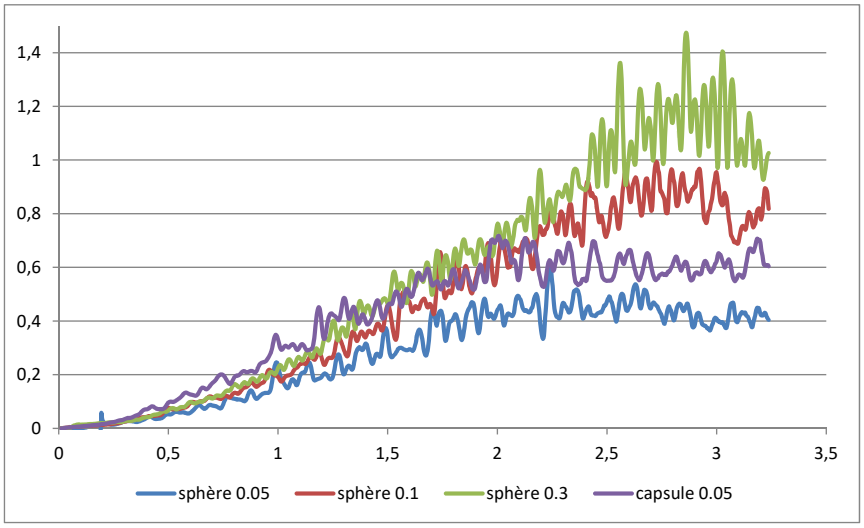



**Figure 6.8.:** Évolution en fonction de la vitesse de la résultante des forces appliquées par le fluide lagrangien sur un solide en déplacement linéaire. La légende indique le type d'objet (sphère ou capsule) utilisé par chaque courbe avec la valeur numérique indiquant le rayon de l'objet.

que pour la capsule correspondant au tibia du personnage le rapport est proche de 0.6. Hors une capsule verticale a vraisemblablement un coefficient de forme proche de celui correspondant à un câble cylindrique, soit environ 1.0 1.3. Par conséquent il est probable que les forces appliquées par le fluide simulé soient environ deux fois trop élevées. Le problème associé à ce constat est que la poussée d'Archimède est correcte, comme nous l'avons vu dans la section 6.2.2, par conséquent nous ne pouvons pas simplement appliquer un facteur de réduction sur les forces calculées par le fluide sans contrepartie.

Il serait intéressant d'étendre cette expérience pour analyser le coefficient de forme. En particulier, il serait intéressant de déterminer le rapport minimal entre les dimensions d'un objet et la taille des particules pour lequel le fluide produit toujours des forces correspondant au modèle théorique. Pour ce faire il serait nécessaire d'étendre notre expérience à plusieurs types d'objets dont le coefficient est connu, par exemple des cubes et des cônes. Également nous remarquons que notre expérience présente un niveau de bruit élevé, il est possible de modifier notre méthodologie pour améliorer la stabilité des résultats. Au lieu d'augmenter de manière continue la vitesse de l'objet, il serait intéressant de réaliser plusieurs simulations séparées où l'objet est maintenu à une vitesse constante différente pour chaque simulation. Ceci permettrait de pouvoir calculer une moyenne sur plusieurs pas de simulation réduisant ainsi le bruit observé. Ces nouveaux résultats plus stables pourraient également être utilisés pour tenter de déterminer la relation entre la vitesse de déplacement et la proportion de drag turbulent et de drag laminaire observée dans le fluide simulé. Nous ne disposons malheureusement pas du temps nécessaire à la réalisation de ces expérimentations plus étendues, cependant, celles-ci permettraient d'obtenir une meilleure compréhension des interactions entre un fluide lagrangien et des solides permettant ainsi d'obtenir un meilleur contrôle sur des systèmes mixtes.

Retournons à notre étude des différences entre les forces appliquées sur le tibia par le modèle simplifié (figure 6.6a) et la simulation lagrangienne (figure 6.6b). Si nous faisons abstraction du fait que l'intensité des forces de friction n'est pas la même pour les deux modèles, nous


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

pouvons noter quelques similarités entre les deux configurations. Durant la phase de vol, c.à.d. la première partie du cycle, le comportement global des forces dans le plan horizontal (axes x et z) sont similaires, et durant la phase d'appui, les forces sur les axes x et y sont identiques en comportement et en intensité. Cependant, nous remarquons deux différences dans l'évolution des forces :

— Lors de la seconde partie du cycle, qui correspond à la phase d'appui, nous pouvons observer une large force positive sur la composante z en début de cycle dans les forces produites par le fluide simulé alors qu'aucune force n'est observée dans les forces calculées par le modèle simplifié. Cette force est probablement due à la combinaison de deux phénomènes. D'une part, en début de cycle la jambe d'appui se redresse, ce qui fait que sa vitesse relative aux particules adjacentes est négative dans l'axe z bien que sa vitesse absolue soit positive. D'autre part, durant la phase de vol, la jambe a entraîné des particules dans son sillage. Au moment du contact du pied avec le sol, le mouvement de la jambe s'arrête brusquement, mais les particules entraînées conservent leur vitesse. Par conséquent, en début de phase d'appui, le fluide simulé "pousse" sur la jambe d'appui d'où l'observation d'une force positive sur l'axe z.

— Lors de la première partie du cycle, qui correspond à la phase de vol, la composante en y diminue puis augmente pour le modèle simplifié alors que la simulation lagrangienne produit une augmentation suivie d'une diminution. Au cours de cette phase, nous observons un mouvement ascendant de la jambe du personnage suivi d'un mouvement descendant. Par conséquent, les forces appliquées par le modèle réduit semblent mieux correspondre au mouvement de la jambe que celles produites par la simulation lagrangienne. Notre supposition est que l'évolution des forces observée dans le cas de la simulation lagrangienne est due à l'impact du déplacement de la jambe dans le fluide qui aurait pour conséquence de créer des turbulences trop faibles pour être visualisée ce qui impacterait les forces calculées par la simulation.

### Interaction entre fluide et solide multiples

Maintenant que nous avons comparé en détail les forces appliquées sur un solide en mouvement par les deux modèles de fluide dans le cas d'un corps rigide isolé, étudions l'impact de la présence d'autres corps rigides à proximité de l'objet étudié. Dans cette expérience nous ajoutons les particules correspondant à tous les membres de la seconde jambe du personnage en plus des particules correspondant au tibia étudié. Ces nouvelles particules de solide se trouvent assez proches pour être capables de générer des courants au niveau des particules en contact avec le tibia étudié mais suffisamment loin pour qu'elles n'empêchent pas certaines parties de la surface du tibia d'être en contact complet avec le fluide. La figure 6.9 présente l'évolution des forces appliquées par le fluide simulé sur le tibia. Nous comparons ces forces à celles observées lorsque le tibia se trouvait seul dans le fluide simulé (figure 6.6b) pour déterminer si la seconde jambe produit des perturbations significatives.

Nous observons une unique différence significative entre les forces obtenues par les deux configurations étudiées. Lors de la seconde partie du cycle de marche, c.à.d. durant la phase d'appui, nous observons une force sur l'axe x, l'axe coronal, atteignant un maximum de 25N en milieu de pas dans la configuration comprenant la seconde jambe du personnage alors que la force sur l'axe x est nulle lorsque le tibia est seul dans la simulation. La force sur ce même axe semble également légèrement plus élevée lors de la phase de vol. Ce phénomène est facilement explicable par le fait que le déplacement de la seconde jambe, qui est en phase de vol durant cette partie du cycle, déplace des particules. Cette seconde jambe étant modélisée



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

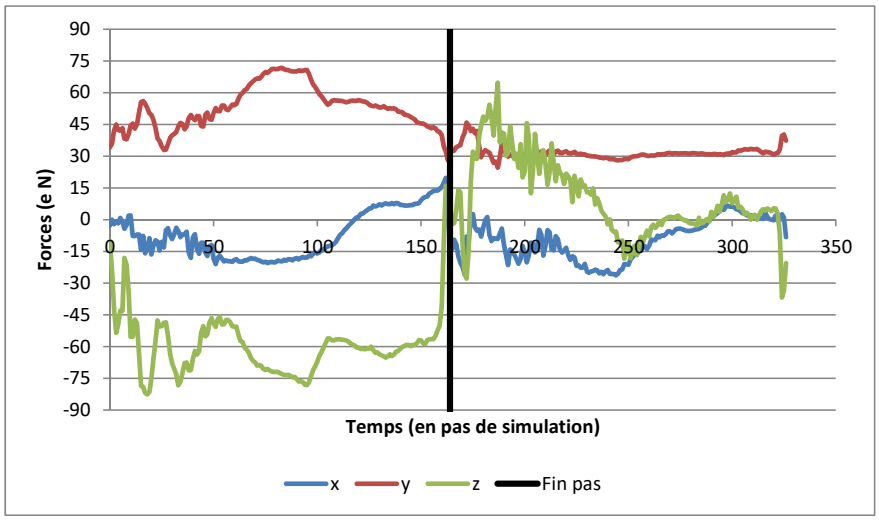



**Figure 6.9.:** Évolution de l'estimation de la résultante des forces appliquées par le fluide sur le tibia du personnage au cours d'un cycle de marche pour un fluide utilisant une simulation lagrangienne. Le corps étudié est le tibia qui se trouve dans la jambe en phase de vol au début du cycle de marche.

par des capsules verticales, les particules déplacées n'ont pas d'autre choix que de se déplacer selon l'axe x. Nous observons donc une compression du fluide entre les deux jambes ce qui augmente l'intensité de la force sur l'axe x durant la phase de vol et provoque l'apparition d'une force sur cet axe durant la phase d'appui.

### Interaction entre fluide et solide proches

Notre dernière expérience est focalisée sur l'impact du fluide sur des corps rigides. Nous cherchons à déterminer l'impact du manque de particules en contact avec la surface des membres de la jambe du personnage, en particulier au niveau des articulations. Pour cette expérience nous utilisons deux configurations utilisant un fluide simulé par le modèle lagrangien. La première configuration utilise une simulation, que nous appellerons simulation complète dans la description des résultats de cette expérience, dans laquelle toutes les membres de la jambe sont représentés par des particules. La deuxième configuration met en place une simulation séparée pour chaque membre de la jambe. Notre supposition est que la somme des forces sur la totalité des membres du personnage devrait être la même mais que la distribution des forces sur les différents membres de la jambe devrait être différente (section 6.2.3). Les figures représentant les résultats de cette expériences se trouvent dans l'annexe A.

Commençons par comparer la somme des forces sur la jambe complète pour les deux configurations (figure A.1). Lors de la seconde partie du cycle de marche, c.à.d. durant la phase d'appui, nous n'observons pas de différence significative dans l'impact du fluide. Cependant, durant la phase de vol, nous pouvons observer des différences entre les deux configurations. Nous observons que la somme des forces sur l'axe y, axe vertical, varie beaucoup moins lorsque l'on considère la totalité des parties de la jambe dans la simulation complète. Sur l'axe y, axe correspondant à la direction du mouvement (axe sagittal), nous remarquons que la somme des forces est identique au début de la phase de vol mais elle diminue plus vite dans la configuration utilisant la simulation complète jusqu'à être deux fois moins importante à la fin du pas. Le fait que les forces correspondent pour la phase d'appui mais diffèrent pour


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

la phase de vol semble indiquer que la présence d'objets très proches les uns des autres a une influence sur la résistance du fluide au mouvement des objets. Avant d'étudier les différences au sein des forces de résistance au mouvement, vérifions que le phénomène de redistribution de la poussée d'Archimède que nous avons supposé précédemment est effectivement observé. Nous sommes supposés observer une force sur l'axe y plus élevée sur le pied qui devrait s'équilibrer par une réduction de la force observée sur la cuisse. Le détail des forces sur la cuisse (figure A.2) et le pied (figure A.4), nous montre qu'en effet, la force observée par le pied est presque deux fois plus élevée sur l'axe y dans la simulation complète (20N) que lorsque les parties de la jambe sont simulées séparément (11N). Une réduction d'environ 7N de la composante en y de la force appliquée sur la cuisse est bien observée lorsque l'on utilise la simulation complète. Nous observons également un écart sur les forces de l'axe y du tibia d'environ 2N (figure A.3). Ce dernier écart est vraisemblablement du bruit mais pourrait être dû au fait que les particules manquantes du fait de la présence du pied ont une pression plus élevée que celles manquantes du fait de la cuisse, ce qui expliquerait que la force ne s'équilibre pas exactement. Des expériences étudiant des objets avec une plus grande taille relative aux particules étudiées seraient nécessaires pour valider cette hypothèse.

Étudions maintenant les différences sur les forces au cours de la phase de vol. La réduction de la résistance au mouvement sur l'axe y est simple à expliquer. Lorsque la jambe à un mouvement ascendant, la cuisse empêche le haut du tibia d'être en contact avec des particules de fluide. Un comportement similaire devrait être observable au niveau de la cheville car le tibia se trouve au-dessus du pied. Inversement lorsque la jambe a un mouvement descendant, le tibia devrait empêcher le bas de la cuisse d'être en contact avec des particules de fluide et le pied devrait empêcher le tibia d'être en contact avec des particules de fluide. Par conséquent en fonction du mouvement de la jambe, les différentes parties de la jambe seront moins affectées par la résistance du fluide au mouvement, ce qui explique l'amplitude plus faible des forces dans la configuration utilisant la simulation complète. La réduction des forces sur l'axe z que nous avons observé peut être expliquée par le fait que le pied se trouve devant le tibia et par conséquent diminuera l'amplitude des forces observées sur le tibia lorsque la jambe se déplace vers l'avant (environ à partir de $1/3$ de la phase de vol). Regardons le détail des forces sur la cuisse (figure A.2), le tibia (figure A.3) et le pied (figure A.4). Nous pouvons voir que sur la phase ascendante, la résistance du fluide sur le pied est moins élevée en début de pas, qui correspond à la phase ascendante de la jambe. Notons que contrairement à nos attentes la présence de la cuisse durant cette phase ne semble pas impacter de manière significative les forces appliquées par le fluide sur le tibia car nous n'observons pas de réduction significative. À partir de la moitié de la phase de vol nous pouvons voir que conformément à nos attentes, les forces sur l'axe y sur la cuisse et le tibia ainsi que la force sur l'axe z du tibia sont moins importantes dans la configuration utilisant la simulation complète.

### Bilan

Récapitulons les conclusions de nos trois expériences :

— L'intensité de la poussée d'Archimède est identique entre le modèle simplifié et le fluide lagrangien. Cependant, le manque de particules au niveau des articulations fait que celle-ci est redistribuée au sein de la jambe dans le fluide simulé.

— Les forces de résistance au mouvement ont une intensité bien plus élevée, entre deux et trois fois, dans le fluide lagrangien. Ceci est dû au fait que la taille des particules est trop proche des dimensions des formes géométriques représentant les membres



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

du personnage. Notons que le manque de particules dû aux articulations diminue légèrement cette différence.

— Les forces de résistance au mouvement générées par le fluide lagrangien ne correspondent pas à la loi décrivant un mouvement dans un milieu turbulent. Les forces semblent correspondre à la loi d'un flux laminaire pour les vitesses très faibles (jusqu'à $0.25m.s^{-1}$) puis à un flux intermédiaire (entre $0.25m.s^{-1}$ et $2.5m.s^{-1}$) ,dont la loi correspond probablement à une combinaison d'une loi d'un milieu laminaire et celle d'un milieu turbulent, avant d'atteindre finalement un flux strictement turbulent (à partir de $2.5m.s^{-1}$.

— Les perturbations introduites dans le fluide par le déplacement des particules ont pour conséquence l'apparition de nouvelles forces qui ne sont pas modélisées par le modèle simplifié. Nous avons noté l'apparition d'une force sur l'axe x qui a tendance à écarter les jambes lorsque la jambe en phase de vol passe à côté de la jambe en phase d'appui et une force sur l'axe z qui a tendance à pousser la jambe en phase d'appui vers l'avant en début de phase d'appui.

### 6.3.2 Contrôle du personnage

Comme nous venons de le voir l'une des difficultés majeures associées à notre fluide lagrangien est qu'il génère des forces de résistance au mouvement trop élevées. Pour se rapprocher de forces plus réalistes il est nécessaire d'appliquer un facteur de réduction sur les forces appliquées par le fluide. Contrairement au modèle simplifié, le fluide lagrangien ne permet pas d'obtenir séparément la poussée d'Archimède et la résistance au mouvement. Par conséquent, si nous appliquons un facteur de réduction celui-ci sera également appliqué sur la poussée d'Archimède. or nous avons vu que, lorsqu'un objet solide est immobile, ce qui veut dire qu'il n'y a pas de forces de résistance au mouvement, la force générée par le fluide correspond bien à la poussée d'Archimède pour le tibia et la cuisse de chaque jambe (section 6.3.1). Nous voudrions donc une solution n'appliquant pas de facteur de réduction sur la partie des forces correspondant à la poussée d'Archimède. Lors de notre expérience considérant une simulation dans laquelle toutes les parties de la jambe sont présentes dans le fluide, nous avons vu qu'à la fin d'un cycle de marche, c.à.d. à la fin de la phase d'appui, la seule composante des forces qu'applique le fluide qui n'est pas nulle est celle sur l'axe y (axe vertical). Cela nous indique que la poussée d'Archimède est toujours strictement verticale pour chaque membre de la jambe malgré le manque local de particules au niveau de certaines parties de la surface de chaque membre. Cela veut dire que les composantes sur les axes x et z correspondent uniquement aux forces de résistance au mouvement. La composante sur l'axe y contient une combinaison de la poussée d'Archimède et des forces de résistance au mouvement. Si on regarde l'évolution des forces sur cet axe au cours d'un cycle, on remarque que pour la cuisse (figure A.2b) la composante y reste proche de la valeur correspondant à la poussée d'Archimède. Cela veut dire que l'intensité des forces de résistance au mouvement sur l'axe y est bien plus faible que l'intensité de la poussée d'Archimède. Nous avons donc décidé de ne pas appliquer de facteur de réduction sur la composante y des forces appliquées par le fluide sur la cuisse. Pour le tibia (figure A.3b), l'intensité des forces de résistance au mouvement sur la composante y varie sur une plus grande amplitude, jusqu'à atteindre une intensité proche de la poussée d'Archimède. Au cours de la phase de vol, l'intensité moyenne des forces de résistance au mouvement est environ de $35\%$ de la poussée d'Archimède. Notons cependant que la force totale est toujours strictement supérieure à la poussée d'Archimède. Donc elle aura tendance à aider le mouvement ascendant de la jambe pour la majorité de la phase de vol. Lorsque la jambe a un mouvement descendant, nous savons que la force de


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

résistance au mouvement sur l'axe y est quasi nulle à cause de la présence du pied dans la simulation. Par conséquent, nous avons décidé de ne pas appliquer de facteur de réduction sur la composante y des forces appliquées par le fluide sur le tibia. Le cas du pied du personnage est plus complexe. Premièrement nous avons précédemment expliqué que le modèle du pied possédant des dimensions plus faibles que la taille des particules, la poussée d'Archimède, due au volume du pied, est plus élevée que la valeur théorique (environ trois fois plus élevée). Si nous regardons l'évolution des forces sur le pied au cours d'un cycle de marche (figure A.4b), nous pouvons voir que l'intensité des forces de résistance au mouvement dépassent largement l'intensité de la poussée d'Archimède (jusqu'à $200\%$). De plus le signe de ces forces change au cours de la phase de vol. Vu que la poussée d'Archimède est plus élevée que la valeur théorique et que les forces de résistance au mouvement sont prédominantes relativement à la poussée d'Archimède, nous avons décidé d'appliquer le facteur de réduction sur la totalité des forces appliquées par le fluide sur le pied.

Pour résumer nous utilisons un facteur de réduction sur :

— les axes x et z de la cuisse

— les axes x et z du tibia

— les axes x, y et z du pied

Nous avons précédemment déterminé que les forces de résistance au mouvement étaient environ deux fois trop élevées pour une capsule dont les dimensions correspondent aux objets de notre simulation, nous appliquerons donc un facteur de réduction de 2.

### Compensation de champs de force externe

Comme nous l'avons présenté dans la vue d'ensemble de notre contrôleur (section 4.1), nous utilisons l'extension du système de compensation externe que nous avions proposé dans notre contrôleur précédent [CPB15]. Cette extension ajoute à la compensation de la gravité proposée par Coros et al. [CBV10] la compensation de la poussée d'Archimède quand le personnage est en interaction avec un fluide. Cependant, l'utilisation de cette extension présente deux difficultés. Premièrement, comme nous l'avons discuté, il n'est pas possible de déterminer la poussée d'Archimède pour un solide en mouvement lorsque le fluide est simulé par une méthode lagrangienne. La solution à cette difficulté est d'utiliser les hypothèses que nous venons de poser pour déterminer sur quelles composantes des forces nous devons appliquer un facteur de réduction. Par conséquent, la compensation de gravité ne compense que la composante y des forces appliquées par le fluide sur le tibia et la cuisse.

La seconde difficulté associée à cette extension est que les valeurs des forces appliquées par le fluide sur le personnage sont nécessaires au calcul des moments à appliquer sur le personnage. Par conséquent, cela nous oblige à utiliser une implémentation séquentielle pour notre simulation dans laquelle la simulation du fluide précède les calculs du contrôle du personnage. Notons tout de même que seuls deux composants du contrôleur dépendent des forces appliquées par le fluide. Le premier est la compensation de la poussée d'Archimède et le second est le stabilisateur des contacts car il utilise les valeurs de moments et forces calculés par tous les autres composants du contrôleur. Le reste des calculs liés au contrôleur (suivi du mouvement de référence, IPM, contrôle de la direction, compensation de la gravité et contrôle de la vitesse) peuvent être réalisés en parallèle de la simulation de fluide. Cependant, le composant du contrôle utilisant le plus de ressources de calcul est le stabilisateur des contacts. Si l'on n'utilise pas la compensation de la poussée d'Archimède le stabilisateur des contacts



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

ne dépend plus du fluide et peut également être réalisé en parallèle de la simulation du fluide lagrangien. Par conséquent, l'utilisation de la compensation de la poussée d'Archimède limite fortement la capacité à réaliser une implémentation parallèle de la simulation de fluide et du contrôleur.

Nous avons réalisé une expérience évaluant si la compensation de la poussée d'Archimède améliore significativement le contrôle du personnage. Rappelons que le but de la compensation de champs de force externe est de mettre le personnage "en apesanteur", pour faire en sorte que l'impact des moments appliqués par les régulateurs PD sur les articulations soit plus contrôlable. L'objectif initial est de faire en sorte que les régulateurs PD ayant pour but le suivi du mouvement de référence n'aient pas à compenser la présence de forces externes de manière à pouvoir utiliser des gains plus faibles et ainsi rendre l'animation moins robotique. Cependant, si la force externe déplace le membre du personnage dans la direction voulue par le mouvement de référence, alors l'utilisation d'un système ajoutant des moments compensant cette force externe force les régulateurs PD à utiliser des moments plus élevés. Nous avons ajouté une règle assurant que si le moment calculé par la compensation de force externe pour une articulation est opposé au moment calculé par les régulateurs PD pour une articulation alors la compensation de la force externe n'est pas appliquée sur cette articulation. Avec cette nouvelle règle, le système de compensation a bien comme objectif explicite la réduction des moments calculés par les régulateurs PD. Par conséquent, nous pouvons évaluer l'importance de la compensation de la poussée d'Archimède en calculant la réduction des moments observés lorsque cette compensation est appliquée sur le personnage.

Pour ce faire, nous avons comparé les valeurs des moments calculés par les régulateurs PD sur la hanche et le genou de la jambe en phase de vol. Ce test n'est effectué qu'au cours de la phase de vol car au cours de la phase d'appui la jambe étant quasiment verticale, le moment dû à l'application d'une force verticale (la poussée d'Archimède) est quasiment nul. Nous avons observé une réduction d'environ $2\%$ du moment appliqué sur le genou et de $5\%$ du moment appliqué sur la hanche. La compensation de la poussée d'Archimède n'apporte donc pas une amélioration significative au contrôle du personnage pour le fluide lagrangien.

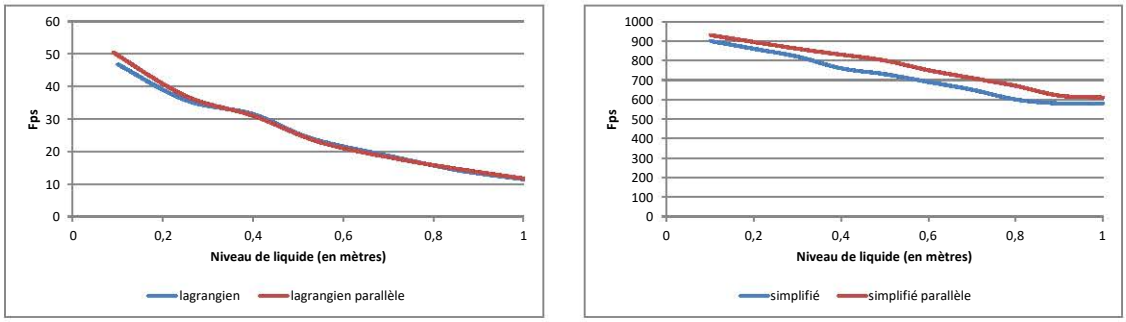
Suite à ces deux constatations, nous avons décidé de ne pas compenser la poussée d'Archimède dans nos expériences utilisant une implémentation parallèle du contrôle et de la simulation de fluide.

### Temps d'exécution

Pour comparer les nombres d'images par seconde (fps) entre les deux modèles de fluide nous relevons les fps moyens pour une durée correspondant à 25 pas du personnage. Vu que la quantité de calculs réalisés dépend du volume de fluide simulé, nous relevons l'évolution du temps moyen en fonction de la hauteur de liquide simulé. Pour notre comparaison nous considérons 4 configurations. Pour chacun des deux modèles de fluide (simplifié et lagrangien) nous relevons les temps d'exécution lorsque la simulation de fluide lagrangien sur GPU (ou fluide simplifié sur CPU) est réalisée en parallèle du contrôle et lorsque la simulation du fluide et le contrôle du personnage sont réalisés séquentiellement. Les résultats sont présentés dans la figure 6.10a pour le système utilisant le fluide lagrangien et dans la figure 6.10b pour le système utilisant le fluide simplifié.

Si nous comparons les résultats obtenus pour les deux modèles de fluide, la vitesse d'exécution du système utilisant le fluide lagrangien est beaucoup plus faible que celle de celui utilisant


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés




**(a)** **(b)**

**Figure 6.10.:** Évolution du nombre d'images par seconde (fps) moyen pour un pas de simulation en fonction de la hauteur de liquide simulé (en mètres). Nous comparons les fps pour des implémentations séquentielles et parallèles (a) du fluide lagrangien et (b) du fluide simplifié.

le fluide simplifié même pour des quantités de liquide faibles. Pour 10cm de fluide, le fluide lagrangien limite la vitesse d'exécution à environ 60 fps alors que le fluide simplifié permet d'obtenir environ 900 fps pour un niveau de liquide équivalent. Notons que la fréquence de simulation est de 300Hz. Le système utilisant le fluide simplifié ne permet d'obtenir que des temps d'exécution correspondant à $1/5$ du temps réel pour 10 cm de liquide et $1/30$ du temps réel pour un mètre de liquide. Si nous regardons les temps d'exécution pour 70000 particules, ce qui correspond à 1 mètre de fluide avec la configuration de notre expérience, que nous avions obtenus lors de notre étude du fluide (section 5.5.3) nous pouvons voir que nous aurions dû obtenir des temps d'exécution proches de $1/10$ du temps-réel. L'écart entre les $1/10$ du temps réel attendu et les $1/30$ observés s'explique par trois raisons : le temps de rendu, la présence de corps rigides en mouvement en interaction avec le fluide et la fenêtre dynamique de simulation. La prise en compte du rendu dans cette expérience réduit les performances observées. Cependant, notre rendu est simple, chaque particule est représentée par une sphère bleue dont l'opacité dépend de sa vitesse de déplacement, et pour 1 mètre de fluide, nous observons moins d'1 fps supplémentaire lorsque nous le désactivons. La raison expliquant cette différence de performance est que le fluide est moins stable dans cette expérience que le fluide au repos utilisé pour obtenir les temps d'exécution lors de notre étude du fluide. En effet, bien que le déplacement de la fenêtre dynamique requière un temps négligeable, elle perturbe l'état du fluide, en particulier sa divergence. Par conséquent, après chaque déplacement de la fenêtre dynamique, nous observons une courte durée au cours de laquelle le simulateur de fluide doit utiliser plus d'itérations des processus assurant une divergence nulle et une densité constante. Ceci augmente fortement le temps d'exécution des pas de simulation suivant chaque déplacement de la fenêtre dynamique. Ces résultats illustrent bien le coût élevé associé au meilleur réalisme physique du fluide lagrangien. En effet, le fluide simplifié permet d'obtenir des performances entre 2 et 3 fois supérieures au temps réel, soit entre 15 et 60 fois les performances du fluide lagrangien.

Si nous regardons en détail la distribution du temps d'exécution pour la simulation utilisant un mètre de fluide lagrangien (environ 90ms), nous pouvons voir que la quasi-totalité du temps d'exécution est utilisée pour la simulation du fluide (89.2ms) et seule une très faible partie est due au contrôle du personnage (0.35ms) et à la simulation des corps rigides (0.45ms). C'est ce qui explique que nous n'observons pas de différence entre les fps obtenus avec une implémentation séquentielle ou parallèle.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

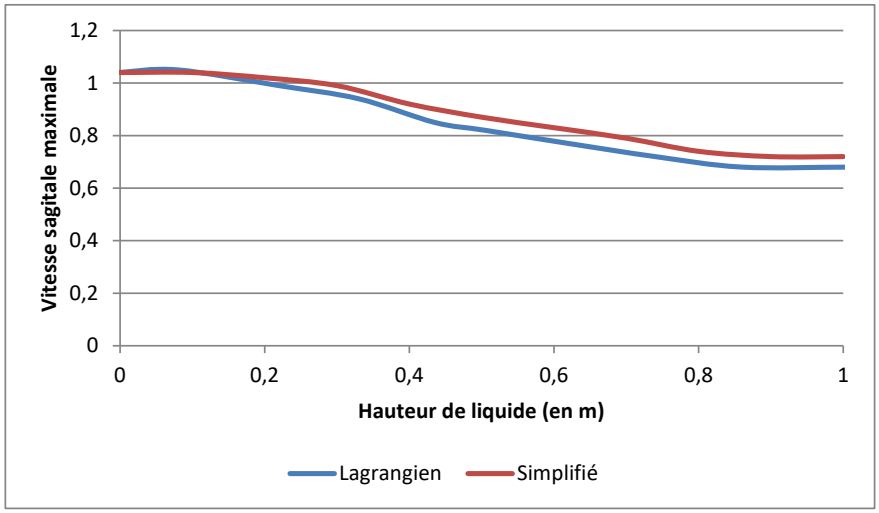



**Figure 6.11.:** Évolution de la vitesse sagittale maximale atteignable par notre contrôle pour un mouvement de marche en fonction de la hauteur de liquide présente dans la simulation pour chacun des deux modèles de fluide.

Le modèle simplifié nécessite 0.3ms pour 1 mètre de fluide. Donc nous pourrions espérer des gains de performance élevés avec l'implémentation parallèle par rapport à l'implémentation séquentielle. En effet, les calculs du fluide étant réalisés en parallèle des calculs du contrôleur nous supposions environ $30\%$ de gains de performance. Cependant, nous n'observons qu'entre $5\%$ et $10\%$ de gain sur la figure 6.10b. Nous supposons que cette différence est due au fait que le système de parallélisation interne au moteur physique (ODE) semble bloquer l'exécution de threads en parallèle de l'exécution d'un pas de simulation. Notre système d'amélioration des contacts repose sur l'utilisation du moteur physique pour estimer les contacts futurs ce qui limite l'intégration parallèle du fluide et du contrôle du personnage.

### Impact sur le contrôle du personnage

Dans cette section nous allons évaluer l'impact de chacun des deux modèles de fluide sur le contrôle du personnage. Notre objectif est de mesurer à quel point la présence du fluide entrave la capacité de notre contrôleur à atteindre de manière exacte une vitesse cible. Notre expérience a pour objectif de déterminer la vitesse maximale atteignable en fonction de la hauteur de liquide pour chacun des deux modèles de fluide. Le processus utilisé est de fixer une vitesse de déplacement faible pour laquelle nous sommes sûrs que le contrôleur est capable produire un mouvement stable, puis nous augmentons la vitesse cible jusqu'à observer la chute du personnage. Pour chaque hauteur de liquide nous répétons l'expérience 5 fois et relevons la moyenne des vitesses désirée maximale avant échec du contrôle. Lorsque nous nous approchons de la vitesse maximale, les incréments de la vitesse cible sont fixés à $0.02m.s^{-1}$ et le contrôleur dispose de 20 pas personnage entre deux incrémentations de la vitesse cible.

Comme nous pouvons le voir sur la figure 6.11, les deux modèles de fluide ont un impact similaire sur la vitesse maximale atteignable par notre contrôleur. Nous observons que le modèle lagrangien restreint légèrement davantage la vitesse maximale, environs $0.03m.s^{-1}$. Cependant cette différence est négligeable car les vitesses atteintes sont au minimum de $0.7m.s^{-1}$ (soit une différence inférieure à $5\%$. Notons tout de même que cette différence est


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

probablement due au fait que comme nous pouvons le voir sur les figures relevant l'intensité de l'impact du fluide de chacun des deux modèles au cours d'un pas du personnage dans notre comparaison de l'impact des deux modèles (section 6.3.1), la force sur l'axe sagittal (axe z) atteint plus rapidement son intensité maximale avec le modèle lagrangien. La raison est que, comme nous l'avons vu, la force appliquée sur un solide par le fluide simulé ne correspond pas exactement à un flot turbulent mais à une combinaison d'un flot turbulent et d'un flot laminaire. Hors pour les vitesses faibles observées lors du début de la phase de vol d'une jambe, la force due à un flot laminaire (c-à-d à une loi linéaire) est plus élevée que la force due à un flot turbulent (c-à-d une loi quadratique). Pour des liquides pour lesquels la viscosité du fluide limiterait fortement les vitesses maximales l'écart entre les deux modèles augmenterait probablement. Cependant, il serait nécessaire que le modèle simplifié prenne en compte la formule réaliste de l'impact de la viscosité du fluide pour pouvoir réaliser une telle expérience.

Notre conclusion est que l'utilisation du modèle lagrangien ne semble pas avoir un impact significativement différent sur le contrôle du personnage. Le modèle simplifié du fluide que nous utilisons permet d'obtenir un impact tout aussi réaliste sur la vitesse de déplacement du personnage et permet d'obtenir des temps d'exécution temps-réel. Rappelons tout de même que nous avons appliqué un facteur de réduction sur les forces de résistance au mouvement du fluide lagrangien pour compenser le fait que les parties du personnage n'ont pas une taille suffisamment grande relativement à la taille des particules. Il est possible que des expériences utilisant des particules suffisamment petites pour ne pas nécessiter la présence d'un tel facteur de réduction permettent d'obtenir des résultats différents. Cependant, que nous l'avons noté précédemment, une réduction de la taille particules n'est pas une solution envisageable si l'on désire que le contrôle du mouvement reste interactif.

## 6.4 Limitations

La première limitation de notre système que nous voulons évoquer est liée à la taille des particules. Nos expériences comparant les forces produites par le fluide simulé et le modèle simplifié semblent indiquer qu'un rapport d'au moins 4 entre le diamètre des particules et les dimensions des objets solides est nécessaire pour que les forces de résistance au mouvement correspondent aux formules théoriques (figure 6.8). Hors dans nos expériences les capsules représentant les membres des jambes du personnage ont un rayon correspondant à deux fois le rayon des particules. Ceci nous force à mettre en place un facteur de réduction sur les forces appliquées par le fluide. La raison pour laquelle nous avons choisi de ne pas réduire davantage le rayon des particules est que si l'on divise par deux le rayon des particules de fluide il est nécessaire d'utiliser environ cinq fois plus de particules pour représenter le même volume. Ceci augmenterait fortement les temps de simulation nécessaires pour chaque pas de simulation.

Une seconde limitation de notre système est probablement également liée à la taille des particules. Nous avons observé que les forces appliquées par le fluide sont extrêmement bruitées. Nous pouvons proposer deux solutions pour diminuer le bruit présent dans la simulation. La première est de diminuer la taille des particules de façon à ce que la résultante des forces appliquées par le fluide sur chaque solide soit basée sur un plus grand échantillon de forces de pression. Une seconde solution serait d'améliorer la stabilité des particules de fluide au niveau des contacts avec les solides en calculant les valeurs de pression pour les particules de solide au lieu de considérer qu'elles ont la même pression que les particules



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

de fluide voisines. Cette solution a permis à Band et al. [Ban+18] d'améliorer fortement la stabilité du IISPH en particulier pour des simulations avec des bordures dynamiques.

Une autre limitation est liée à la stabilité du moteur physique utilisé (ODE). Comme nous l'avions évoqué dans notre section présentant les travaux sur le contrôle du personnage, le moteur ODE n'offre pas des forces de contact stables pour des fréquences de simulation proches de 300Hz. Malgré l'ajout de notre composant visant à stabiliser les contacts entre les pieds du personnage et le sol, nous avons observé de nombreuses simulations échouer dues à un manque de friction. En particulier, si les forces de résistance au mouvement sont suffisamment élevées sur la jambe en phase de vol alors celle-ci sera plus "ancrée" dans le fluide que le pied du personnage n'est "ancré" au sol. La conséquence est que les moments appliqués sur la hanche de la jambe en phase d'appui visant à contrôler l'orientation du pelvis causent une rotation de la jambe en phase d'appui. Cette limitation est importante car elle limite fortement les possibilités d'application de notre contrôleur. En effet, nous observons ce problème si nous n'appliquons pas le facteur compensant les dimensions faibles de nos objets relativement à la taille des particules. Cela veut dire que notre système ne serait pas capable de supporter un fluide deux fois plus dense que l'eau ou bien un courant opposé au personnage augmentant les forces de résistance au mouvement. Une solution plus robuste au problème des contacts est nécessaire ; probablement l'utilisation d'un moteur plus stable tel que MuJoCo ou l'utilisation d'un modèle mou pour le pied profitant du temps libre du CPU en parallèle des calculs GPU du fluide.

## 6.5 Bilan

Nous avons proposé dans cette section un système permettant le contrôle basé sur la physique d'un personnage bipède en interaction avec un fluide simulé par une approche lagrangienne. Nous avons montré que pour que les forces appliquées par le fluide sur le personnage soient réalistes il est nécessaire d'étendre la simulation du fluide en dessous du sol physique ainsi que de placer les particules représentant les objets solides dans la simulation à l'intérieur de la surface de l'objet qu'elles représentent.

Nous avons comparé les forces appliquées par notre fluide lagrangien sur des solides à celles produites par un modèle simplifié de fluide. Le modèle simplifié utilisé correspond au modèle de fluide proposé dans notre contrôleur précédent et calcule l'impact du fluide à travers les formules classiques de la poussée d'Archimède et celle de la résistance au mouvement du fluide. Ces expériences nous ont permis d'établir plusieurs conclusions. Premièrement, le fluide lagrangien permet bien d'ajouter un niveau de réalisme supplémentaire à la simulation comme le montre l'apparition de nouvelles forces dues aux déplacements du fluide causé par le mouvement du personnage. Ce meilleur niveau de réalisme est également visible par la présence de phénomènes réalistes comme la redistribution de la poussée d'Archimède au sein d'une chaîne de corps rigides articulés ainsi qu'un phénomène d'aspiration qui diminue l'intensité des forces de résistance au mouvement sur un objet si une autre objet se trouve devant dans la direction du mouvement. Ces expériences nous ont également permis de voir que pour qu'un fluide lagrangien produise des forces correspondant aux formules théoriques il semble nécessaire que les objets présents dans la simulation aient des dimensions au minimum quatre fois plus grandes que le diamètre des particules de fluide. Également elles nous ont permis de déterminer que les forces produites par le fluide lagrangien ne correspondent pas exactement aux formules pour un régime turbulent mais probablement à une combinaison des formules pour un régime laminaire et un régime turbulent.


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

Nous avons comparé l'impact des deux modèles de fluide sur le contrôle du mouvement du personnage. Le fluide lagrangien permet l'observation de phénomènes, courants interne au fluide du au mouvement du personnage et effet d'aspiration pour des corps rigide proches, qui correspondent au comportement attendu d'un fluide réel. Cependant, ces phénomènes additionnels ne semble pas impacter significativement le contrôle de vitesse du personnage. De plus, le fluide lagrangien ne permet d'obtenir que des performances relativement interactives contrairement au fluide simplifié qui permettait l'obtention de performances temps-réel. Notons tout de même que nous avons dû réaliser une approximation, liée à l'utilisation de particules trop larges nécessaires pour obtenir des temps d'exécution interactifs, peut avoir limité l'impact des phénomènes additionnels présents dans le modèle lagrangien.

Mentionnons est qu'il aurait été intéressant de comparer les résultats avec des études du mouvement bipède humain en milieu aquatique. Il nous semble que certaines thérapies de rééducation utilisent ce type d'exercices. Il existe donc probablement des données sur ce type de déplacement, notamment sur l'intensité des forces musculaires ce qui aurait pu nous servir de point de comparaison avec les moments que nous appliquons sur nos articulations. Cependant, nous n'avons pas disposé du temps nécessaire à la réalisation d'une telle comparaison.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

# 7 Conclusion

Dans le cadre de cette thèse, nous nous sommes intéressé au contrôle, basé sur la physique, d'un personnage virtuel bipède en locomotion en interaction avec un milieu complexe. Nous nous sommes focalisés sur les interactions entre un personnage en locomotion bipède et un fluide. Pour ce type d'interactions, la littérature propose de représenter un liquide par un modèle simplifié appliquant des forces sur le personnage pour représenter les forces de résistance au mouvement. Cependant une telle représentation ne prend pas en compte le caractère dynamique du fluide. En particulier, un modèle simplifié ne prend pas en compte l'impact des courants dus au déplacement du personnage. Dans notre cas, pour augmenter le réalisme de l'animation du personnage, nous avons opté pour une simulation physique du fluide.

Notre objectif est la création d'un système mettant en interaction un personnage contrôlé par une méthode basée sur la physique ainsi qu'une simulation physique d'un fluide. Nous voulons également que notre système propose des performances interactives. Pour simplifier l'étude et la conception, nous avons, dans un premier temps, considéré chacune des deux parties indépendamment en cherchant à maximiser les performances en temps d'exécution ainsi que la robustesse.

La partie du système sur laquelle nous avons consacré le plus de temps est le contrôle basé sur la physique du personnage virtuel. Nous avons choisi d'utiliser un contrôleur de type SIMBICON, basé sur l'approche du contrôle dans l'espace des articulations. Ce choix a été motivé par la faible quantité de ressources de calcul que nécessite ce type de contrôleur à chaque pas de simulation. Dans un premier temps, nous avons amélioré la robustesse et la flexibilité du contrôleur. Pour ce faire, nous avons, d'une part, amélioré le contrôle de la vitesse du personnage en améliorant la stabilité et la vitesse de convergence du *velocity tunning* ainsi qu'en ajoutant des règles qui organisent l'utilisation des différents composants du contrôleur ayant un impact sur la vitesse du personnage. En particulier, nous avons cherché à favoriser l'ajout de moments par le *velocity tunning* et n'utiliser l'altération d'IPM pour déformer le mouvement du personnage que lorsque l'ajout de moments ne permet plus d'atteindre la vitesse désirée. D'autre part, nous avons modifié le système de contrôle de la direction du déplacement pour rendre possibles des changements brusques dans la direction du personnage. Notre solution met en place des périodes de transition permettant au personnage de changer de direction progressivement au cours de deux pas au lieu de forcer une modification instantanée de la direction du mouvement. Suite à ces améliorations, nous avons montré que notre contrôleur était capable d'atteindre exactement une vitesse cible en moins de 7 pas du personnage et qu'il était capable de supporter des changements d'orientation allant jusqu'à 2.4 radians.

Nous avons ensuite cherché à diminuer les ressources de calcul nécessaires au contrôle du personnage pour pouvoir simuler un fluide en parallèle. L'une des alternatives permettant d'obtenir des gains de performance importants est de diminuer la fréquence de simulation. Une telle démarche permet de diminuer les calculs requis à la simulation d'un intervalle de temps et augmente le temps disponible pour réaliser des calculs entre deux pas de simulation. L'un des problèmes soulignés dans la littérature est un manque de stabilité de la simulation



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

physique. Ce phénomène mène à un échec du contrôle du personnage lorsqu'une fréquence de simulation inférieure à 750Hz est utilisée. Notre étude nous a menée au constat que les instabilités au sein des forces de friction entre le pied d'appui et le sol sont le problème principal pour le contrôle du personnage. Pour cela, nous avons mis en place un système d'optimisation en ligne qui assure une bonne répartition des forces de contact entre le pied et le sol, sur la surface inférieure du pied, assurant ainsi une friction plus stable. Nous avons également remarqué que les valeurs des gains des régulateurs-PD sont spécifiques à une fréquence de simulation. Concernant ce second point, pour déterminer les valeurs adaptées à chaque fréquence, nous avons utilisé des optimisations hors ligne qui utilisent une fonction objectif spécifique pour l'évaluation des gains au lieu d'évaluer la qualité du mouvement du personnage. Nous avons proposé un processus utilisant plusieurs optimisations successives des optimisations successives pour déterminer les valeurs de gains permettant un contrôle stable pour une fréquence de simulation faible à partir des gains d'une simulation haute fréquence stable. Ces travaux nous ont permis de proposer un contrôleur robuste pour des fréquences de simulation jusqu'à 300Hz.

Dans une seconde partie, nous avons étudié les méthodes de simulation réalistes d'un fluide. Notre choix s'est porté sur l'approche lagrangienne car elle offre de meilleures performances vis-à-vis du temps d'exécution. Parmi les algorithmes existants, l'algorithme *divergence-free SPH* (DFSPH) permet d'obtenir le meilleur niveau de stabilité et les meilleures performances. Les simulations lagrangiennes de fluide sont facilement parallélisables. Par conséquent, l'une des méthodes les plus courantes pour améliorer les performances d'un simulateur de fluide lagrangien est de mettre en place une implémentation sur GPU. Le DFSPH ne disposant actuellement que d'une implémentation parallèle CPU, nous avons proposé une implémentation GPU, sous CUDA, de cet algorithme. Nous avons également étudié l'impact sur notre implémentation GPU de diverses optimisations présentes dans la littérature telles que l'utilisation d'un index de Morton et l'utilisation d'un démarrage à chaud. Notre implémentation GPU de l'algorithme DFSPH, a permis d'obtenir, suivant le nombre de particules simulées, un facteur d'accélération entre 10 et 20 par rapport à l'implémentation CPU existante. Enfin nous avons proposé un système qui permet à notre simulateur de représenter un océan plat infini en n'utilisant qu'un nombre relativement faible de particules. Ce système utilise une fenêtre dynamique de simulation qui permet de déplacer la zone de simulation du fluide. Notre approche déplace une tranche de fluide se trouvant vers l'arrière dans la direction du déplacement pour compléter l'espace vide vers l'avant du déplacement créé par le mouvement de la fenêtre. Cette solution permet un déplacement rapide et efficace de la fenêtre de simulation et assure une conservation du volume de liquide présent dans la simulation.

Enfin, nous avons proposé un système qui intègre le contrôleur et la simulation lagrangienne de fluide. Nous avons vu que pour que le fluide lagrangien applique des forces réalistes sur des solides, il est nécessaire que les particules représentant les solides soient placées à l'intérieur du solide et non pas sur la surface de celui-ci. La simulation de fluide doit également être étendue en dehors des bordures de la simulation des corps rigides (i.e. sous le sol) pour assurer la présence de forces réalistes pour tous les cas. Enfin, pour que l'intensité des forces soit réaliste, les dimensions des objets en contact avec le fluide doivent être au moins quatre fois supérieures au diamètre des particules. Cependant, une telle condition forcerait l'utilisation d'une très faible taille de particules, augmentant d'autant les ressources de calcul nécessaires. Pour nous permettre d'utiliser des particules plus larges, nous avons mis en place un facteur de réduction des forces de résistance au mouvement que le fluide lagrangien applique sur le personnage.
Nous avons aussi comparé les forces appliquées par le fluide lagrangien à celles calculées


Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

par un modèle simplifié basé sur les formules classiques de la résistance d'un fluide au mouvement et de la poussée d'Archimède. L'un des résultats de notre étude de l'évolution des forces en fonction de la vitesse de déplacement des objets nous a montré que les forces générées par le fluide lagrangien ne correspondent pas uniquement à un régime turbulent mais probablement à un régime intermédiaire entre régime laminaire et turbulent. Nous avons également remarqué que le fluide simulé permet d'observer la présence de forces additionnelles dues aux courants présents dans le fluide dus au déplacement du personnage dans le fluide.

Nous avons finalisé nos travaux en comparant l'impact des deux fluides sur le contrôle de la vitesse du personnage. Nous avons pu remarquer que les forces additionnelles présentes avec le fluide simulé ne semblent pas impacter le contrôle de la vitesse du personnage car l'évolution de la vitesse maximale atteignable en fonction de la hauteur de liquide est la même avec les deux modèles de fluides. Cependant, il est possible que ce manque d'impact soit dû à notre utilisation d'un facteur de réduction des forces de résistance au mouvement du fluide lagrangien qui sert à compenser l'utilisation de particules de fluide de tailles assez grosses pour permettre une exécution interactive. L'une des conclusions que l'on pourrait tirer est que l'utilisation d'un modèle simplifié pour déterminer l'impact d'un fluide sur le déplacement du personnage est justifié lorsque le contrôle du personnage soit réalisé en temps réel. Cependant, l'utilisation d'un fluide simulé par un modèle lagrangien peut être justifié si l'on veut prendre en compte des fluides avec une viscosité non négligeable.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

# Perspectives

Bien que notre contrôleur permette l'obtention d'un contrôle robuste pour des fréquences allant jusqu'à 300Hz, le contrôle n'est pas parfait à cette fréquence. La limitation principale de notre contrôleur est que notre système d'optimisation en ligne dont le but est de stabiliser les contacts ne permet pas d'obtenir une stabilité suffisante, en particulier lorsque les instabilités dues au moteur physique sont élevées. La résolution de ce phénomène par l'ajout de moments non liés au contrôle direct du personnage dans la jambe d'appui est une solution technique mais elle n'est pas satisfaisante d'un point de vue conceptuel. En effet, nous avons pu observer que ces moments additionnels perturbent fortement le contrôle du personnage, en particulier le contrôle de sa vitesse. Nous pensons que notre approche de la stabilisation des contacts du coté du contrôleur n'est pas l'approche la plus adéquate. En particulier il serait plus intéressant que le contrôleur n'ait pas à corriger les contacts lui-même. La solution la plus naturelle serait de disposer d'un moteur stable (à l'image de MuJoCo). Cependant, même de tels moteurs n'offrent pas des contacts parfaitement stables pour les fréquences de simulation faibles, habituellement utilisées dans les systèmes temps réel (60hz-120Hz). Au vu des travaux existant sur la stabilisation des contacts, l'utilisation d'un corps mou sur la partie inférieure du pied semble l'approche la plus prometteuse. Les travaux de Jain et Liu [JL11] montrent que cette approche permet d'obtenir de très bons résultats pour des fréquences de simulation élevée. Dans notre cas, cette solution a été étudiée et temporairement écartée. Il serait intéressant de reprendre cette étude pour mieux explorer une telle solution qui aurait l'avantage de rendre la stabilité des contacts indépendante du moteur physique. Notons que, d'autres axes de recherche sont possibles sur le contrôle du personnage tel qu'une étude plus avancée des gains en fonction de la fréquence de simulation et du type de mouvement. En particulier il serait intéressant de chercher à déterminer une relation entre la vitesse observée dans le mouvement de référence pour une articulation et le gain nécessaire dans la simulation, une telle relation dépendrait probablement de la masse des membres du personnage déplacés par l'articulation. Cependant, de tels travaux se retrouveraient probablement fortement limités si le problème de l'instabilité des contacts n'est pas résolu en premier.

Une comparaison plus approfondie des temps d'exécution de notre simulation de fluide obtenus avec ceux d'autres implémentations GPU de méthodes lagrangiennes serait intéressante. Il n'existe cependant pas de test "standard" en simulation de fluide. Par exemple, la publication proposant une implémentation GPU de l'algorithme IISPH ([Ihm+14a]) utilise des particules de 9cm de diamètre alors que nous utilisons des particules de 5cm de diamètre. Or, la taille des particules a une influence sur la stabilité du fluide, et donc le temps nécessaire à la simulation. Par conséquent, les comparaisons directes des temps d'exécution observés peuvent ne pas refléter une comparaison correcte des deux simulateurs de fluide. L'un des axes de recherche qui nous semble le plus intéressant en simulation lagrangienne de fluide serait l'étude des variations des nombres d'itérations des solveurs de densité et de divergence. Nous avons vu que ces variations rendent instables les interactions entre le fluide et les solides.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

Une amélioration de leur stabilité pourrait rendre l'utilisation d'une simulation de fluide dans un système interactif plus facile car plus déterministe. Comme nous l'avons montré, l'une des principales sources de ces variations est le système de démarrage à chaud. Par conséquent l'étude d'un nouveau système de démarrage à chaud serait un point de départ intéressant. Il serait également intéressant de calculer les valeurs de pression au niveau des particules de solides car les travaux récents en simulation lagrangienne ont montré que cela permet d'augmenter la stabilité d'un fluide IISPH ([Ban+18]). Une telle modification permet peut-être de résoudre le problème des variations du nombre d'itérations des solveurs que nous avons observé ce qui expliquerait l'important gain de stabilité visible dans les résultats présentés par Band et al. [Ban+18].

Avec l'utilisation de notre système de fenêtre dynamique le fluide devient instable au niveau des bordures et notre système est limité à un océan plat. Les instabilités sont principalement dues aux déplacements artificiels (i.e. qui ne dépendent pas des lois de physique) des particules au niveau des bordures ce qui provoque l'apparition de compressions et de dépressions dans le fluide. Le problème principal avec cette approche est que nous mettons en contact des particules qui se trouvaient éloignées au pas de temps précédent. Dans ce cas, non seulement il est possible que la disposition des particules ne corresponde plus à un fluide au repos, mais également les valeurs de démarrage à chaud ne sont plus correctes. Il existe théoriquement une solution à ce problème. Au lieu de déplacer les particules vers la surface du fluide au niveau des zones où nous savons que des compressions et des dépressions seront présentes, une solution consisterait à modifier la disposition des particules pour qu'elle corresponde à un fluide au repos. Un système capable de placer de manière procédurale les particules pour correspondre à un fluide au repos serait un axe de recherche très intéressant. En effet, un tel système permettrait également d'initialiser directement un fluide avec des particules qui correspondent effectivement à une disposition réaliste. Un tel procédé permettrait également d'initialiser un fluide dans un état où des vagues ainsi que des courants sont présents. La limitation de notre fenêtre dynamique à un océan plat est en partie due à l'absence d'une solution pour placer des particules à un état stable. Cependant, compte tenu du caractère isolé de notre système de fenêtre dynamique, l'existence d'un procédé permettant le placement de particules à un état proche du repos ne suffirait pas pour représenter des courants internes au fluide. Pour représenter de tels courants il serait nécessaire de pouvoir observer des échanges de particules entre la zone simulée et l'extérieur. Pour cela nous pouvons proposer l'utilisation d'une zone tampon au niveau des bordures de la zone de simulation. Cette zone pourrait être composée de particules disposées selon un état stable dont la position serait réinitialisée à la fin de chaque pas de temps. Ainsi il serait possible d'observer des échanges entre le fluide et cette zone tampon et si la fenêtre dynamique se déplace vers la zone tampon, celle-ci peut être intégrée au fluide simulé.

Nous soulignons qu'un aspect très intéressant de notre fenêtre de fluide dynamique actuelle est que chaque déplacement de la fenêtre ne requiert qu'une très faible quantité de calculs. Ces performances sont en partie dues au fait qu'il n'y a aucune allocation de mémoire lors du déplacement de la fenêtre. Bien que notre système introduise des perturbations vers les bordures dans la direction du déplacement ainsi qu'au niveau de la surface du fluide, "l'intérieur" du fluide est très peu perturbé. Une application intéressante de notre approche serait la simulation du déplacement de créatures aquatiques. En particulier, si l'on désire simuler simultanément le déplacement de multiples créatures pouvant s'éloigner les unes des autres.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

# Bibliographie


[AAT13] N. AKINCI, G. AKINCI et M. TESCHNER. “Versatile surface tension and adhesion for SPH fluids”. en. In : *ACM Transactions on Graphics* 32.6 (nov. 2013), p. 1–8 (cf. p. 36, 101).

[Aki+12] N. AKINCI, M. IHMSEN, G. AKINCI, B. SOLENTHALER et M. TESCHNER. “Versatile Rigid-fluid Coupling for Incompressible SPH”. In : *ACM Trans. Graph.* 31.4 (juil. 2012), 62 :1–62 :8 (cf. p. 36, 93, 101).

[Ban+18] S. BAND, C. GISSLER, M. IHMSEN et al. “Pressure Boundaries for Implicit Incompressible SPH”. en. In : *ACM Transactions on Graphics* 37.2 (fév. 2018), p. 1–11 (cf. p. 37, 140, 148).

[Bar+17] C. BARONEA, A. COJOCARU, M. FRANCU, A. MORAR et V. ASAVEI. “Comparison between Incompressible SPH Solvers”. In : *2017 21st International Conference on Control Systems and Computer Science (CSCS)*. Mai 2017, p. 271–278 (cf. p. 35).

[BB07] A. BOEING et T. BRÄUNL. “Evaluation of real-time physics simulation systems”. In : *Proceedings of the 5th international conference on Computer graphics and interactive techniques in Australia and Southeast Asia*. ACM, 2007, p. 281–288 (cf. p. 8).

[Ben] J. BENDER. *SPlisHSPlasH is an open-source library for the physically-based simulation of fluids. : InteractiveComputerGraphics/SPlisHSPlasH*. URL : %5Curl%7Bhttps://github.com/InteractiveComputerGraphics/SPlisHSPlasH%7D (visité le 30 oct. 2017) (cf. p. 39, 79–81).

[Ben+17] J. BENDER, D. KOSCHIER, T. KUGELSTADT et M. WEILER. “A Micropolar Material Model for Turbulent SPH Fluids”. In : *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*. SCA ’17. New York, NY, USA : ACM, 2017, 4 :1–4 :8 (cf. p. 36).

[Ber+18a] L. BERMUDEZ, J. TESSENDORF, D. ZIMMERMANN et V. ZORDAN. “Real-time locomotion with character-fluid interactions”. In : *Proceedings of the 11th Annual International Conference on Motion, Interaction, and Games - MIG ’18*. the 11th Annual International Conference. Limassol, Cyprus : ACM Press, 2018, p. 1–8 (cf. p. 117).

[Ber+18b] G. BERSETH, C. XIE, P. CERNEK et M. VAN DE PANNE. “Progressive reinforcement learning with distillation for multi-skilled motion control”. In : *arXiv preprint arXiv :1802.04765* (2018) (cf. p. 16).

[BET14] J. BENDER, K. ERLEBEN et J. TRINKLE. “Interactive Simulation of Rigid Body Dynamics in Computer Graphics : Interactive Rigid Body Simulation”. en. In : *Computer Graphics Forum* 33.1 (fév. 2014), p. 246–270 (cf. p. 7, 8).




Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

[BK15] J. BENDER et D. KOSCHIER. “Divergence-free Smoothed Particle Hydrodynamics”. In : *Proceedings of the 14th ACM SIGGRAPH / Eurographics Symposium on Computer Animation*. SCA ’15. New York, NY, USA : ACM, 2015, p. 147–155 (cf. p. 35, 37, 77, 79, 81, 89, 91, 103, 104, 114).

[BK17] J. BENDER et D. KOSCHIER. “Divergence-Free SPH for Incompressible and Viscous Fluids”. In : *IEEE Transactions on Visualization and Computer Graphics* 23.3 (mar. 2017), p. 1193–1206 (cf. p. 35).

[Bli89] R. BLICKHAN. “The spring-mass model for running and hopping”. In : *Journal of Biomechanics* 22.11 (jan. 1989), p. 1217–1227 (cf. p. 20).

[BM07] R. BRIDSON et M. MÜLLER-FISCHER. “Fluid Simulation : SIGGRAPH 2007 Course notesVideo Files Associated with This Course Are Available from the Citation Page”. In : *ACM SIGGRAPH 2007 Courses*. SIGGRAPH ’07. New York, NY, USA : ACM, 2007, p. 1–81 (cf. p. 29).

[BMM15] J. BENDER, M. MÜLLER et M. MACKLIN. “Position-Based Simulation Methods in Computer Graphics.” In : *Eurographics (Tutorials)*. 2015 (cf. p. 8).

[BN88] L. S. BROTMAN et A. N. NETRAVALI. “Motion Interpolation by Optimal Control”. In : *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH ’88. New York, NY, USA : ACM, 1988, p. 309–315 (cf. p. 15).

[BR86] J. U. BRACKBILL et H. M. RUPPEL. “FLIP : A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions”. In : *Journal of Computational Physics* 65.2 (août 1986), p. 314–343 (cf. p. 31).

[Bri15] R. BRIDSON. *Fluid Simulation for Computer Graphics, Second Edition*. en. A K Peters/CRC Press, sept. 2015 (cf. p. 30).

[BT07] M. BECKER et M. TESCHNER. “Weakly Compressible SPH for Free Surface Flows”. In : *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA ’07. Aire-la-Ville, Switzerland, Switzerland : Eurographics Association, 2007, p. 209–217 (cf. p. 35, 36).

[CBP09] S. COROS, P. BEAUDOIN et M. van de PANNE. “Robust Task-based Control Policies for Physics-based Characters”. In : *ACM SIGGRAPH Asia 2009 Papers*. SIGGRAPH Asia ’09. New York, NY, USA : ACM, 2009, 170 :1–170 :9 (cf. p. 14, 23).

[CBV10] S. COROS, P. BEAUDOIN et M. VAN DE PANNE. “Generalized biped walking control”. In : *ACM Transactions on Graphics (TOG)* 29.4 (2010), p. 130 (cf. p. 13, 14, 18, 20, 21, 24, 25, 42, 50, 66, 135).

[CM11] N. CHENTANEZ et M. MÜLLER. “Real-time Eulerian Water Simulation Using a Restricted Tall Cell Grid”. In : *ACM SIGGRAPH 2011 Papers*. SIGGRAPH ’11. New York, NY, USA : ACM, 2011, 82 :1–82 :10 (cf. p. 30).

[CPB15] S. CARENSAC, N. PRONOST et S. BOUAKAZ. “Real-time Gait Control for Partially Immersed Bipeds”. In : *Proceedings of the 8th ACM SIGGRAPH Conference on Motion in Games*. MIG ’15. New York, NY, USA : ACM, 2015, p. 177–182 (cf. p. 3, 14, 20, 22, 23, 41–44, 46, 50, 60, 62–64, 66, 67, 72, 76, 117, 118, 135).

[CPS92] R. W. COTTLE, J.-S. PANG et R. E. STONE. *The Linear Complementarity Problem*. en. Google-Books-ID : bGM80_pSzNIC. SIAM, jan. 1992 (cf. p. 7).



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

[DCG11] J. M. Domínguez, A. J. C. Crespo et M. Gómez-Gesteira. “Optimization strategies for parallel CPU and GPU implementations of a meshfree particle method”. In : *arXiv :1110.3711 [cs]* (oct. 2011). arXiv : 1110.3711 (cf. p. 38, 89, 102).

[DK01] R. A. Dalrymple et O. Knio. “SPH Modelling of Water Waves”. In : *Coastal Dynamics ’01*. Fourth Conference on Coastal Dynamics. Lund, Sweden : American Society of Civil Engineers, 18 mai 2001, p. 779–787 (cf. p. 36).

[ETT15] T. Erez, Y. Tassa et E. Todorov. “Simulation tools for model-based robotics : Comparison of Bullet, Havok, MuJoCo, ODE and PhysX”. In : *2015 IEEE International Conference on Robotics and Automation (ICRA)*. Mai 2015, p. 4397–4404 (cf. p. 8).

[GEF15] P. Goswami, A. Eliasson et P. Franzén. “Implicit Incompressible SPH on the GPU”. eng. In : *DIVA*. Eurographics - European Association for Computer Graphics, 2015 (cf. p. 38, 109).

[GM77] R. A. Gingold et J. J. Monaghan. “Smoothed particle hydrodynamics : theory and application to non-spherical stars”. en. In : *Mon Not R Astron Soc* 181.3 (déc. 1977), p. 375–389 (cf. p. 30).

[GP12] T. Geijtenbeek et N. Pronost. “Interactive Character Animation Using Simulated Physics : A State-of-the-Art Review”. en. In : *Computer Graphics Forum* 31.8 (déc. 2012), p. 2492–2515 (cf. p. 10–12, 14, 16).

[GPS12] T. Geijtenbeek, N. Pronost et A. F. van der Stappen. “Simple Data-driven Control for Simulated Bipeds”. In : *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA ’12. Goslar Germany, Germany : Eurographics Association, 2012, p. 211–219 (cf. p. 21).

[Gre16] D. A. Greer. “Physics-based character locomotion control with large simulation time steps.” PhD Thesis. Bournemouth University, 2016 (cf. p. 24, 25, 27, 52, 55, 56, 58).

[GT17] S. B. C. Gissler et M. Teschner. “Moving Least Squares Boundaries for SPH Fluids”. en. In : *Virtual Reality Interaction and Physical Simulation* (2017), p. 8 (cf. p. 36).

[GY11] S. Giovanni et K. Yin. “LocoTest : deploying and evaluating physics-based locomotion on multiple simulation platforms”. In : *Motion in Games* (2011), p. 227–241 (cf. p. 8, 25, 52).

[Häm+14] P. Hämäläinen, S. Eriksson, E. Tanskanen, V. Kyrki et J. Lehtinen. “Online motion synthesis using sequential Monte Carlo”. en. In : *ACM Transactions on Graphics* 33.4 (juil. 2014), p. 1–12 (cf. p. 15, 26).

[Han+16] D. Han, H. Eom, J. Noh et J. S. Shin formerly Sung Yong Shin. “Data-guided Model Predictive Control Based on Smoothed Contact Dynamics”. en. In : *Computer Graphics Forum* 35.2 (mai 2016), p. 533–543 (cf. p. 8, 15, 17, 26).

[Han06] N. Hansen. “The CMA Evolution Strategy : A Comparing Review”. en. In : *Towards a New Evolutionary Computation*. Studies in Fuzziness and Soft Computing. Springer, Berlin, Heidelberg, 2006, p. 75–102 (cf. p. 24, 58).



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

[HNS18] D. HAN, J. NOH et J. S. SHIN. “Physics-based trajectory optimization with automatic time warping : Daseong Han, Junyong Noh, and Joseph S Shin”. en. In : *Computer Animation and Virtual Worlds* 29.3-4 (mai 2018), e1813 (cf. p. 8, 15, 17, 26).

[Hod+95] J. K. HODGINS, W. L. WOOTEN, D. C. BROGAN et J. F. O’BRIEN. “Animating Human Athletics”. In : *Proceedings of the 22Nd Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH ’95. New York, NY, USA : ACM, 1995, p. 71–78 (cf. p. 13).

[HRL15] P. HÄMÄLÄINEN, J. RAJAMÄKI et C. K. LIU. “Online control of simulated humanoids using particle belief propagation”. en. In : *ACM Transactions on Graphics* 34.4 (juil. 2015), 81 :1–81 :13 (cf. p. 13, 15, 26).

[HW65] F. H. HARLOW et J. E. WELCH. “Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with Free Surface”. In : *The Physics of Fluids* 8.12 (déc. 1965), p. 2182–2189 (cf. p. 31).

[Ihm+11] M. IHMSEN, N. AKINCI, M. BECKER et M. TESCHNER. “A Parallel SPH Implementation on Multi-Core CPUs”. en. In : *Computer Graphics Forum* 30.1 (mar. 2011), p. 99–112 (cf. p. 33, 37, 38, 83, 89).

[Ihm+14a] M. IHMSEN, J. CORNELIS, B. SOLENTHALER, C. HORVATH et M. TESCHNER. “Implicit Incompressible SPH”. In : *IEEE Transactions on Visualization and Computer Graphics* 20.3 (mar. 2014), p. 426–435 (cf. p. 34, 91, 147).

[Ihm+14b] M. IHMSEN, J. ORTHMANN, B. SOLENTHALER, A. KOLB et M. TESCHNER. “SPH Fluids in Computer Graphics”. In : *Eurographics 2014 - State of the Art Reports*. Sous la dir. de S. LEFEBVRE et M. SPAGNUOLO. The Eurographics Association, 2014 (cf. p. 30, 34, 36, 37).

[IPN14] S. IVALDI, V. PADOIS et F. NORI. “Tools for dynamics simulation of robots : a survey based on user feedback”. en. In : *arXiv :1402.7050 [cs]* (fév. 2014). arXiv : 1402.7050 (cf. p. 8, 9).

[Jia+15] C. JIANG, C. SCHROEDER, A. SELLE, J. TERAN et A. STOMAKHIN. “The Affine Particle-in-cell Method”. In : *ACM Trans. Graph.* 34.4 (juil. 2015), 51 :1–51 :10 (cf. p. 31).

[JL11] S. JAIN et C. K. LIU. “Controlling physics-based characters using soft contacts”. In : *ACM Transactions on Graphics (TOG)*. T. 30. ACM, 2011, p. 163 (cf. p. 26, 53, 147).

[JLT11] JIE TAN, K. LIU et G. TURK. “Stable Proportional-Derivative Controllers”. en. In : *IEEE Computer Graphics and Applications* 31.4 (juil. 2011), p. 34–44 (cf. p. 12).

[Kaj+01] S. KAJITA, F. KANEHIRO, K. KANEKO, K. YOKOI et H. HIRUKAWA. “The 3D linear inverted pendulum mode : a simple modeling for a biped walking pattern generation”. In : *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No.01CH37180)*. T. 1. 2001, 239–246 vol.1 (cf. p. 13).

[Kwa+10] N. KWATRA, C. WOJTAN, M. CARLSON et al. “Fluid simulation with articulated bodies”. In : *IEEE Transactions on Visualization and Computer Graphics* 16.1 (2010), p. 70–80 (cf. p. 117).

[Len+11] M. LENTINE, J. T. GRETARSSON, C. SCHROEDER, A. ROBINSON-MOSHER et R. FEDKIW. “Creature control in a fluid environment”. In : *IEEE Transactions on Visualization and Computer Graphics* 17.5 (2011), p. 682–693 (cf. p. 117).



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

[Liu+10] L. LIU, K. YIN, M. van de PANNE, T. SHAO et W. XU. “Sampling-based contact-rich motion control”. In : *ACM Transactions on Graphics (TOG)*. T. 29. ACM, 2010, p. 128 (cf. p. 23–25, 27, 56).

[Liu+13] L. LIU, K. YIN, B. WANG et B. GUO. “Simulation and control of skeleton-driven soft body characters”. en. In : *ACM Transactions on Graphics* 32.6 (nov. 2013), p. 1–8 (cf. p. 26).

[LN04] R. LEINE et H. NIJMEIJER. *Dynamics and Bifurcations of Non-Smooth Mechanical Systems*. en. Lecture Notes in Applied and Computational Mechanics. Berlin Heidelberg : Springer-Verlag, 2004 (cf. p. 8).

[LPF96] J. LASZLO, M. van de PANNE et E. FIUME. “Limit Cycle Control and Its Application to the Animation of Balancing and Walking”. In : *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH ’96. New York, NY, USA : ACM, 1996, p. 155–162 (cf. p. 13).

[LPY16] L. LIU, M. V. D. PANNE et K. YIN. “Guided Learning of Control Graphs for Physics-Based Characters”. en. In : *ACM Transactions on Graphics* 35.3 (mai 2016), p. 1–14 (cf. p. 23, 26).

[Luc77] L. B. LUCY. “A numerical approach to the testing of the fission hypothesis”. en. In : *The Astronomical Journal* 82 (déc. 1977), p. 1013 (cf. p. 35).

[LYG15] L. LIU, K. YIN et B. GUO. “Improving Sampling-based Motion Control”. In : *Computer Graphics Forum*. T. 34. Wiley Online Library, 2015, p. 415–423 (cf. p. 24, 56).

[MCG03] M. MÜLLER, D. CHARYPAR et M. GROSS. “Particle-based Fluid Simulation for Interactive Applications”. In : *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA ’03. Aire-la-Ville, Switzerland, Switzerland : Eurographics Association, 2003, p. 154–159 (cf. p. 33, 34).

[MK09] J. MONAGHAN et J. KAJTAR. “SPH particle boundary forces for arbitrary boundaries”. en. In : *Computer Physics Communications* 180.10 (oct. 2009), p. 1811–1820 (cf. p. 36).

[Mon89] J. J. MONAGHAN. “On the Problem of Penetration in Particle Methods”. In : *J. Comput. Phys.* 82.1 (mai 1989), p. 1–15 (cf. p. 35).

[Mon92] J. J. MONAGHAN. “Smoothed particle hydrodynamics”. In : *Annual Review of Astronomy and Astrophysics* 30 (1992), p. 543–574 (cf. p. 30).

[Mon94] J. MONAGHAN. “Simulating Free Surface Flows with SPH”. In : *J. Comput. Phys.* 110.2 (fév. 1994), p. 399–406 (cf. p. 35).

[MPP11] U. MUICO, J. POPOVIĆ et Z. POPOVIĆ. “Composite control of physically simulated characters”. en. In : *ACM Transactions on Graphics* 30.3 (mai 2011), p. 1–11 (cf. p. 15, 26).

[Mui+09] U. MUICO, Y. LEE, J. POPOVIĆ et Z. POPOVIĆ. “Contact-aware nonlinear control of dynamic characters”. In : *ACM Transactions on Graphics (TOG)*. T. 28. ACM, 2009, p. 81 (cf. p. 15, 26).

[MW88] M. MOORE et J. WILHELMS. “Collision Detection and Response for Computer Animation”. In : *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH ’88. New York, NY, USA : ACM, 1988, p. 289–298 (cf. p. 8).

[NCX15] X. NIE, L. CHEN et T. XIANG. *Real-Time Incompressible Fluid Simulation on the GPU*. en. Research article. 2015 (cf. p. 37, 38).



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

[NF02] M. Neff et E. Fiume. “Modeling Tension and Relaxation for Computer Animation”. In : *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA ’02. New York, NY, USA : ACM, 2002, p. 81–88 (cf. p. 13).

[OK12] J. Orthmann et A. Kolb. “Temporal Blending for Adaptive SPH”. en. In : *Computer Graphics Forum* 31.8 (déc. 2012), p. 2436–2449 (cf. p. 92).

[Pan96] M. van de Panne. “Parameterized gait synthesis”. en. In : *IEEE Computer Graphics and Applications* 16.2 (mar. 1996), p. 40–49 (cf. p. 12).

[Pee+15] A. Peer, M. Ihmsen, J. Cornelis et M. Teschner. “An implicit viscosity formulation for SPH fluids”. en. In : *ACM Transactions on Graphics* 34.4 (juil. 2015), 114 :1–114 :10 (cf. p. 35).

[Pen+17] X. B. Peng, G. Berseth, K. Yin et M. Van De Panne. “DeepLoco : dynamic locomotion skills using hierarchical deep reinforcement learning”. en. In : *ACM Transactions on Graphics* 36.4 (juil. 2017), p. 1–13 (cf. p. 16).

[Pen+18] X. B. Peng, P. Abbeel, S. Levine et M. van de Panne. “DeepMimic : Example-Guided Deep Reinforcement Learning of Physics-Based Character Skills”. en. In : *arXiv :1804.02717 [cs]* (avr. 2018). arXiv : 1804.02717 (cf. p. 16).

[PF93] M. van de Panne et E. Fiume. “Sensor-actuator Networks”. In : *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH ’93. New York, NY, USA : ACM, 1993, p. 335–342 (cf. p. 16).

[PKF94] M. v. d. Panne, R. Kim et E. Flume. “Virtual Wind-up Toys for Animation”. In : *Proceedings of Graphics Interface ’94*. 1994, p. 208–215 (cf. p. 13).

[PL95] M. van de Panne et A. Lamouret. “Guided Optimization for Balanced Locomotion”. en. In : *Computer Animation and Simulation ’95*. Sous la dir. de D. Terzopoulos et D. Thalmann. Vienna : Springer Vienna, 1995, p. 165–177 (cf. p. 10).

[Pra+01] J. Pratt, C.-M. Chew, A. Torres, P. Dilworth et G. Pratt. “Virtual Model Control : An Intuitive Approach for Bipedal Locomotion”. en. In : *The International Journal of Robotics Research* 20.2 (fév. 2001), p. 129–143 (cf. p. 21).

[Rui+16] A. L. C. Ruiz, C. Pontonnier, N. Pronost et G. Dumont. “Muscle-Based Control for Character Animation”. In : *Computer Graphics Forum* 36.6 (2016), p. 122–147 (cf. p. 11).

[SB12] H. Schechter et R. Bridson. “Ghost SPH for Animating Water”. In : *ACM Trans. Graph.* 31.4 (juil. 2012), 61 :1–61 :8 (cf. p. 35, 36, 101).

[SC92] A. J. Stewart et J. F. Cremer. “Animation of 3d human locomotion : Climbing stairs and descending stairs”. In : *Eurographics Workshop on Animation and Simulation*. Citeseer, 1992, p. 152–168 (cf. p. 15).

[SG11] B. Solenthaler et M. Gross. “Two-scale Particle Simulation”. In : *ACM SIGGRAPH 2011 Papers*. SIGGRAPH ’11. New York, NY, USA : ACM, 2011, 81 :1–81 :8 (cf. p. 92).

[Si+14] W. Si, S.-H. Lee, E. Sifakis et D. Terzopoulos. “Realistic Biomechanical Simulation and Control of Human Swimming”. en. In : *ACM Transactions on Graphics* 34.1 (déc. 2014), p. 1–15 (cf. p. 117).



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

[Sil+17] D. B. da SILVA, R. F. NUNES, C. A. VIDAL et al. “Tunable Robustness : An Artificial Contact Strategy with Virtual Actuator Control for Balance : Tunable Robustness”. en. In : *Computer Graphics Forum* 36.8 (déc. 2017), p. 499–510 (cf. p. 14, 27).

[Sim94] K. SIMS. “Evolving Virtual Creatures”. In : *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH ’94. New York, NY, USA : ACM, 1994, p. 15–22 (cf. p. 16).

[SP09] B. SOLENTHALER et R. PAJAROLA. “Predictive-corrective Incompressible SPH”. In : *ACM SIGGRAPH 2009 Papers*. SIGGRAPH ’09. New York, NY, USA : ACM, 2009, 40 :1–40 :6 (cf. p. 34).

[SS17] A. STOMAKHIN et A. SELLE. “Fluxed animated boundary method”. en. In : *ACM Transactions on Graphics* 36.4 (juil. 2017), p. 1–8 (cf. p. 92, 93).

[Ste92] A. J. STEWART. “Beyond Keyframing : An Algorithmic Approach to Animation”. en. In : *Graphics Interface ’* (1992), p. 9 (cf. p. 15).

[Sto+13] A. STOMAKHIN, C. SCHROEDER, L. CHAI, J. TERAN et A. SELLE. “A material point method for snow simulation”. en. In : *ACM Transactions on Graphics* 32.4 (juil. 2013), p. 1 (cf. p. 31).

[Tan+11] J. TAN, Y. GU, G. TURK et C. K. LIU. “Articulated Swimming Creatures”. In : *ACM SIGGRAPH 2011 Papers*. SIGGRAPH ’11. New York, NY, USA : ACM, 2011, 58 :1–58 :12 (cf. p. 13, 117).

[TET12] E. TODOROV, T. EREZ et Y. TASSA. “MuJoCo : A physics engine for model-based control”. In : *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Oct. 2012, p. 5026–5033 (cf. p. 8).

[Tod14] E. TODOROV. “Convex and analytically-invertible dynamics with contacts and constraints : Theory and implementation in MuJoCo”. In : *2014 IEEE International Conference on Robotics and Automation (ICRA)*. Mai 2014, p. 6054–6061 (cf. p. 8, 26).

[TT94] X. TU et D. TERZOPOULOS. “Artificial fishes : physics, locomotion, perception, behavior”. en. In : ACM Press, 1994, p. 43–50 (cf. p. 117).

[TY09] J. TAN et X. YANG. “Physically-based fluid animation : A survey”. In : *Science in China Series F : Information Sciences* 52.5 (mai 2009), p. 723–740 (cf. p. 3).

[Van+10] H. VAN WELBERGEN, B. J. H. VAN BASTEN, A. EGGES, Z. M. RUTTKAY et M. H. OVERMARS. “Real Time Animation of Virtual Humans : A Trade-off Between Naturalness and Control”. In : *Computer Graphics Forum* 29.8 (déc. 2010), p. 2530–2554 (cf. p. 1).

[WZ10] C.-C. WU et V. ZORDAN. “Goal-directed Stepping with Momentum Control”. In : *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA ’10. Goslar Germany, Germany : Eurographics Association, 2010, p. 113–118 (cf. p. 15).

[XL16] X. XIA et Q. LIANG. “A GPU-accelerated smoothed particle hydrodynamics (SPH) model for the shallow water equations”. en. In : *Environmental Modelling & Software* 75 (jan. 2016), p. 28–43 (cf. p. 37, 38).

[YLS04] P.-F. YANG, J. LASZLO et K. SINGH. “Layered dynamic control for interactive character swimming”. en. In : ACM Press, 2004, p. 39 (cf. p. 117).



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

[YLV07] K. YIN, K. LOKEN et M. VAN DE PANNE. “Simbicon : Simple biped locomotion control”. In : *ACM Transactions on Graphics (TOG)*. T. 26. ACM, 2007, p. 105 (cf. p. 14, 17–19, 25, 27, 41, 62, 66, 67, 69).

[ZP05] P. ZHAO et M. van de PANNE. “User Interfaces for Interactive Control of Physics-based 3D Characters”. In : *Proceedings of the 2005 Symposium on Interactive 3D Graphics and Games*. I3D ’05. New York, NY, USA : ACM, 2005, p. 87–94 (cf. p. 14).



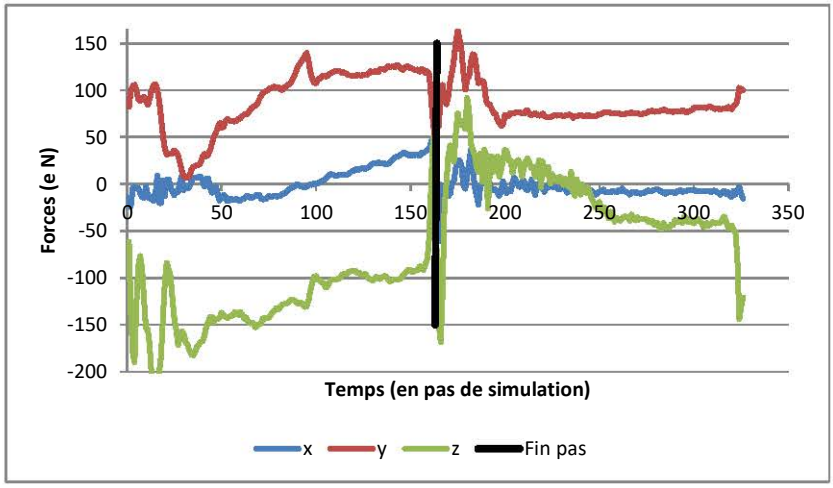
Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

# Comparaison forces pour une jambe simulée

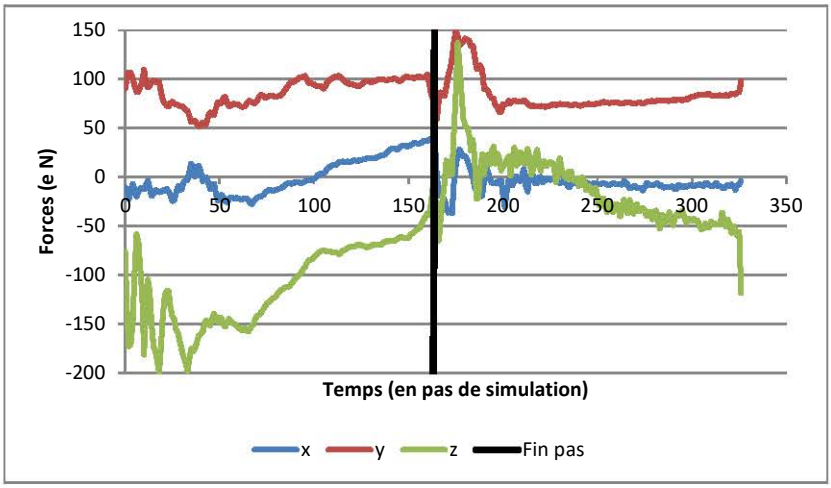
A



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés




(a)

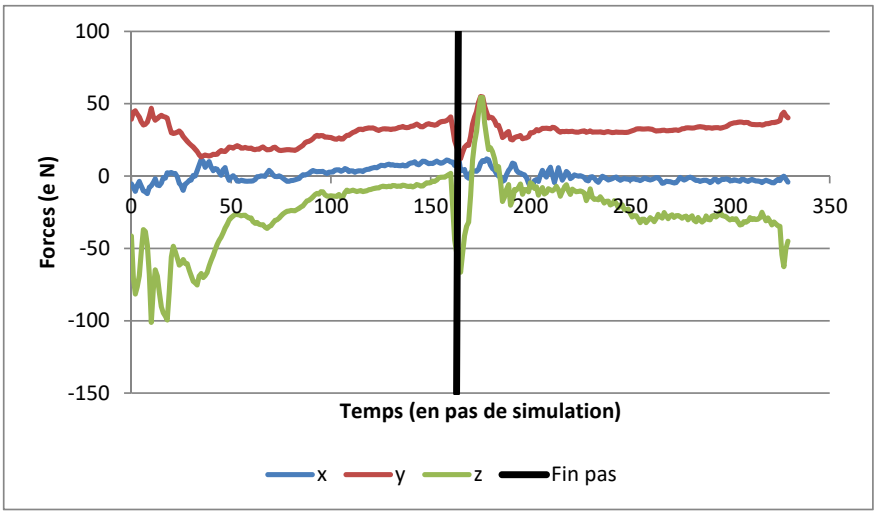



(b)

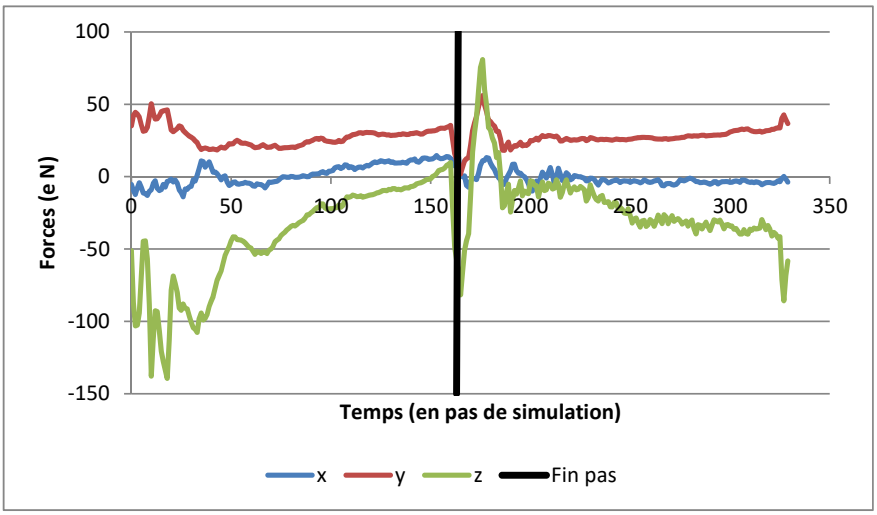
**Figure A.1.:** Évolution de l'estimation de la somme des résultantes des forces appliquées par le fluide sur les parties de la jambe du personnage (cuisse, tibias et pied) au cours d'un cycle de marche ; (a) : simulation séparées pour chaque partie de la jambe ; (b) : simulation unique pour toute la jambe. La jambe étudiée est celle qui se trouve dans la jambe en phase de vol au début du cycle de marche.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés




(a)

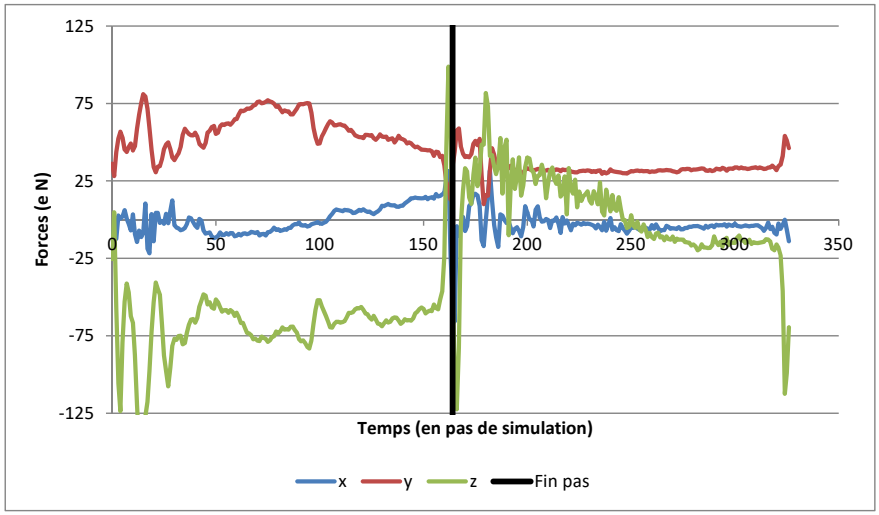



(b)

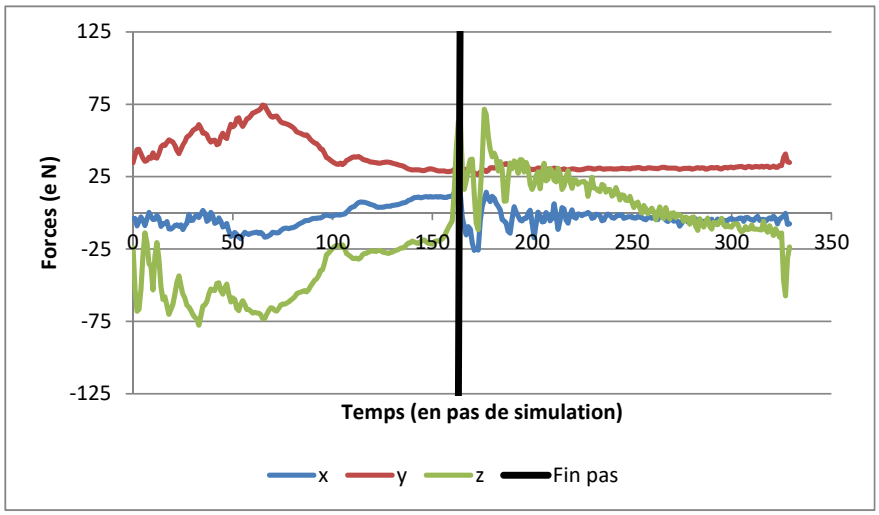
**Figure A.2.:** Évolution de l'estimation de la résultante des forces appliquées par le fluide sur la cuisse du personnage au cours d'un cycle de marche ; (a) : cuisse isolée ; (b) : jambe complète simulée. La cuisse étudiée est celle qui se trouve dans la jambe en phase de vol au début du cycle de marche.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés




(a)

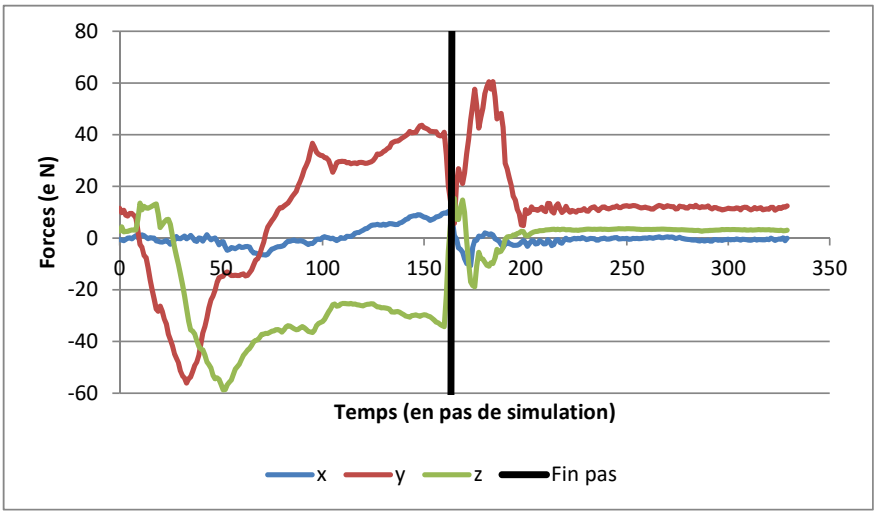



(b)

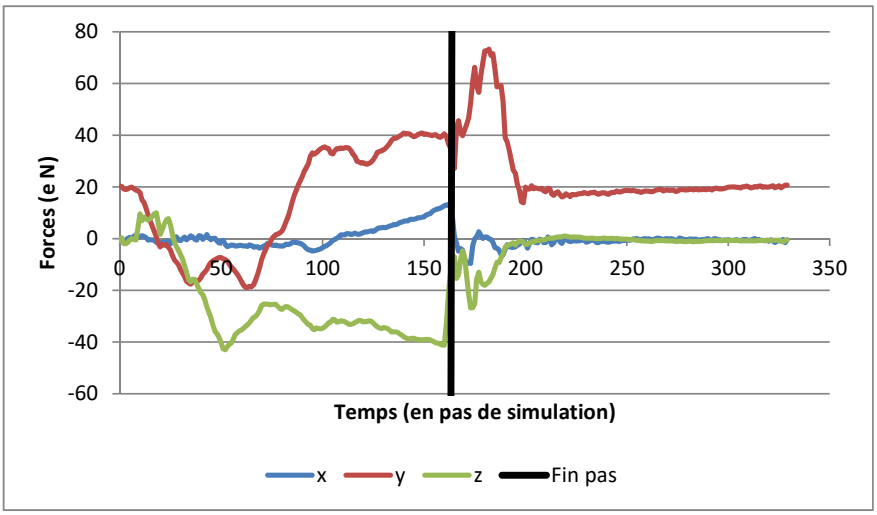
**Figure A.3.:** Évolution de l'estimation de la résultante des forces appliquées par le fluide sur le tibias du personnage au cours d'un cycle de marche ; (a) : tibias isolé ; (b) : jambe complète simulée. Le tibias étudié est celui qui se trouve dans la jambe en phase de vol au début du cycle de marche.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés




(a)




(b)

**Figure A.4.:** Évolution de l'estimation de la résultante des forces appliquées par le fluide sur le pied du personnage au cours d'un cycle de marche ; (a) : pied isolé ; (b) : jambe complète simulée. Le pied étudié est celui qui se trouve dans la jambe en phase de vol au début du cycle de marche.



Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés

# FOLIO ADMINISTRATIF

## THESE DE L'UNIVERSITE DE LYON OPEREE AU SEIN DE L'INSA LYON

NOM : CARENSAC DATE de SOUTENANCE : 05/07/2019

Prénoms : Samuel

TITRE : Contrôle physique de mouvement de personnages virtuels en environnement complexe

NATURE : Doctorat Numéro d'ordre : 2019LYSEI037

Ecole doctorale : InfoMaths

Spécialité : Informatique

RESUME : Cette thèse traite de l'animation de personnages virtuels composés de corps rigides reliés par des articulations et contrôlés par des interactions physiques (forces et moments). Le contrôleur est le système qui calcule dynamiquement ces interactions. Notre objectif est d'étudier et de réaliser un contrôleur pour la simulation de mouvements d'un personnage en interaction avec un fluide.

La complexité du comportement de tels milieux ne permet pas de prédire les interactions entre le personnage et le fluide. Il en découle que le contrôleur proposé doit être capable de réagir à celles-ci. Nous avons focalisé nous travaux sur la conception d'un contrôleur de type SIMBICON capable de s'adapter aux perturbations apportées par la présence d'un fluide simulé physiquement. Ce choix est motivé par notre contrôleur précédent qui proposait un contrôleur en interaction avec un fluide représenté à travers l'utilisation de formule de dynamique des fluides simples. L'utilisation d'une véritable simulation physique du fluide nous permettrait d'améliorer le réalisme physique de la simulation en prenant en compte l'impact du déplacement du personnage sur le fluide.

Ayant pour objectif un contrôleur interactif nous nous sommes focalisés sur deux axes principaux. Le premier est la conception d'un contrôleur capable de supporter des fréquences de simulation faibles tout en conservant la vitesse de calcul apporté par l'utilisatio du modèle SIMBICON. Nous proposons de réduire les instabilités introduites par l'utilisation de fréquences de simulation faibles par un système de feedback utilisant une optimisation en ligne permettant d'obtenir une meilleure stabilité des contacts. Ce système, associé à une étude des paramètres du système en fonction de la fréquence de simulation, nous a permis de proposé un contrôleur capable de supporter des fréquences de simulation allant jusqu'à 225Hz. Le second axe de recherche visait à proposer une implémentation entièrement GPU et interactive d'une simulation de fluide lagrangienne. Nous avons étudié l'impact sur les performances de notre implémentation GPU de plusieurs optimisations proposé par des travaux proposant des implémentations parallèles CPU. Nous proposons également une solution permettant de déplacer la zone de fluide simulé en cours de simulation pour limiter l'espace de simulation du fluide à la proximité immédiate du personnage au cours de son déplacement pour assurer une simulation du fluide en temps interactif.

MOTS-CLÉS : Animation basée sur la physique, Personnage virtuel, Contrôle de mouvement, Simulation lagrangienne, Calcul GPU

Laboratoire (s) de recherche : LIRIS

Directeur de thèse: M. BASKURT Atilla

Président de jury : --

Composition du jury :
M. BASKURT Atilla
M. MULTON Franck
M. REVERET Lionel
MME CANI Marie-paule
M. PRONOST Nicolas
MME BOUAKAZ Saïda
MME GIBET Sylvie

Cette thèse est accessible à l'adresse : http://theses.insa-lyon.fr/publication/2019LYSEI037/these.pdf
© [S. Carensac], [2019], INSA Lyon, tous droits réservés