# Université de Bordeaux

## Habilitation à diriger des recherches

Spécialité : Informatique

le 10 décembre 2012

*Un parcours de recherche des méthodes formelles aux robots humanoïdes.*

Olivier Ly

Après avis des rapporteurs :

Xavier Blanc
Pierre Blazevic
David Naccache

Soutenue devant le jury :

Michel Bauderon, Président
Xavier Blanc, rapporteur
Pierre Blazevic, rapporteur
Stephane Grumbach, examinateur
Yves Métivier, directeur de recherche
Féthi Ben Ouezdou, examinateur

# Préambule

J'ai consacré mes premiers travaux de recherche à des questions touchant à la théorie des langages et la géométrie, porté par un intérêt très marqué pour les mathématiques, mais également pour l'informatique.

Puis, je me suis intéressé progressivement à des considérations plus pratiques, poussé par un désir d'application de ces travaux et plus généralement des méthodes formelles. Un séjour dans l'industrie m'a permis de m'initier au plus près à ces problématiques. Je me suis alors passionné pour l'élaboration d'outils sophistiqués basés sur les méthodes formelles au travers d'applications liées à la sécurité informatique et embarquée.

A ma pratique de la programmation notamment embarquée et la conception de systèmes s'est adjointe alors une passion d'enfant pour les robots. Emprunte d'une naïveté agréable, elle m'a conduit à la robotique en m'initiant par une voie pragmatique consistant à concevoir et réaliser des robots. Je me suis alors plongé à cet art, notamment lors d'un séjour dans l'équipe Flowers. Le spectre large de cette discipline est extrêmement enthousiasmant.

Ces quelques années de recherche ont été pour moi comme un voyage naviguant entre théorie et pratique, jalonné de rencontres fascinantes avec des mystères non résolus. C'est une expérience de plaisir souvent, parfois de souffrance aussi, pris entre l'obscession d'aboutir et le besoin irréversible de se laisser aller à la curiosité. Mais toujours une expérience de rêves et d'apprentissages.

Je veux remercier mes collègues et amis H. Gimbert, L. Goubin et A. Vincent pour leur soutien, la collaboration quotidienne excitante, leur amitié sans faille. Je veux également exprimer ma reconnaissance à G. Sénizergues et Ph. Narbel qui m'ont initié à l'art et m'ont appris l'essentiel, à M. Bauderon, S. Chaumette, B. Courcelle, E. Fleury, C. Gavoille, M. Mosbah et P.-Y. Oudeyer pour toutes ces discussions motivantes et leurs conseils avisés qui ont guidé mon travail au fil de ces années. Je remercie Y. Métivier pour ses encouragements et son soutien bienveillant. Je remercie également mes élèves J. Andronick, R. Chen, C. Thuillet, Ph. Andouard et R. Tabary pour m'avoir supporté. J'ai enfin une pensée toute spéciale à l'égard des contributeurs du projet Rhoban : G. Passault, L. Hofer, et pour D. Lapoire. Merci à mon ami Loïc.

Je tiens également à remercier mes rapporteurs, X. Blanc, P. Blazevic et D. Naccache pour leurs remarques éclairées, extrêmement utiles.

Je dedie ces travaux à ma famille et à mes proches, à mes enfants Pierre et Héloïse, à Irène.

# Table des matières

# Introduction

En introduction, j'expose les principales questions qui ont guidé mes travaux durant ces dernières années. Je retrace les idées et leur enchaînement, et les raisons parfois de cohérence, parfois conjoncturelles, et parfois de curiosité qui m'ont conduit dans telle ou telle voie. Je souhaite donner ici une vue d'ensemble de mes contributions.

## Thèse : Topologie et théorie des langages

Thèse : *Etude algorithmique de complexes simpliciaux infinis*.
Directeurs de thèse : G. Sénizergues et Ph. Narbel.

Les travaux de ma thèse furent consacrés aux variétés topologiques définies par des moyens constructifs. Les moyens constructifs considérés sont les grammaires de remplacement d'hyperarcs (voir e.g. [27]). Ces grammaires définissent des hypergraphes infinis (les graphes HR-équationnels) qui, lorsque l'on interprète les hyperarcs par des simplexes, définissent à leur tour de façon naturelle des objets topologiques. Ce sont ces objets, en particulier lorsqu'ils définissent des variétés, qui font l'objet de cette étude. La première partie a consisté à étudier le cas de la dimension 2, et en particulier le cas des surfaces. Nous avons donné un algorithme pour décider si deux surfaces non compactes HR-équationnelles sont homéorphes ou ne le sont pas.

O. Ly. *On Effective Decidability of the Homeomorphism Problem for Non-Compact Surfaces*. Contemporary Mathematics 250, p89-112. AMS (1998).

En toute généralité, le cas des surfaces non compactes à bord demeure ouvert. Néanmoins, nous avons étendu le théorème de classification de Kerékjarto-Richards au cas des surfaces planaires non compactes à bord (cf. Thèse, chapitre "Topologie florale"). Dans une deuxième partie, nous avons étudié le cas des variétés de dimension 3. Nous avons mis en lumière la classe des variétés hyperboliques non compactes arborescentes, généralisant celles de volume fini. Nous avons montré que dans le cas de la dimension 3, ces variétés sont rigides, i.e., caractérisées par leurs groupes fondamentaux, en s'appuyant sur l'étude des espaces de Teichmüller associés.

O. Ly. *Construction of Pseudo-Isometries for Tree-Like Non-Compact 3-Manifolds.* Comptes Rendus de l'Académie des Sciences - CRAS 337 :7, p 457-460 (2003).

Dans le cas HR-équationnel, nous avons décrit ces groupes sous formes de produits amalgamés d'un nombre infini de groupes finiment présentés, en nombre fini à isomorphisme près.

De plus, la présentation d'un tel groupe peut être décrite en terme de langages rationnels (cf. Thèse). La dernière partie de ces travaux concerne les graphes automatiques, qui généralisent les graphes HR-équationnels. Nous avons montré que le problème de décider si un tel graphe possède un unique bout est indécidable. Nous avons également montré que ce problème est équivalent au problème de décider si tous les graphes engendrés par un D0L-système sont connexes ou non, qui est donc de même indécidable.

O. Ly. *Automatic Graph and D0L-Sequences of Finite Graphs*. Journal of Computer and System Sciences - JCSS 67 :3, p 497-545 (2003).

# Graphes et Langages

## Grammaires Pullback

L'étude des graphes infinis constructifs m'a conduit à m'intéresser à diverses méthodes de définition de tels graphes. Mon intérêt s'est porté en particulier sur les méthodes basées sur les catégories et en particulier basées sur l'opération de pull-back, en collaboration avec M. Bauderon, professeur à l'université de Bordeaux.

Ces travaux prennent leur source dans le contexte de la réécriture de graphe. En particulier, soulignons l'existence de deux directions correspondant à deux approches de la structure de graphe : soit des sommets liés par des arcs (réécriture de sommets), soit des arcs recollés sur des sommets (réécriture d'arcs et d'hyperarcs). Dans les deux directions, trois niveaux de description ont été explorés : le point de vue théorie des ensembles, le point de vue algébrique (algèbres universelles), ou encore le point de vue des catégories. Pour ce dernier, un effort important a été concentré sur le concept de réécriture d'arcs et d'hyperarcs, en utilisant le pushout comme opération élémentaire pour généraliser la substitution habituelle. Cela a conduit au développement d'un corpus conséquent basé sur le simple et double pushout (voir [99]).

Les travaux de [12] proposent une approche duale basée sur l'opération de pullback. Dans ce contexte, nous avons montré que grammaires de graphe basées sur le pullback (pullback graph grammars) sont hors-contexte. Conduisant plus avant l'exploration de cette notion, nous avons étudiés les graphes infinis définis par ces grammaires de façon projectives. Il s'est avéré que les graphes infinis définis par des grammaires pullback sont automatiques, et de ce fait, leur logique du premier ordre est décidable, en revanche la logique MSO est indécidable.

M. Bauderon, R. Chen, O. Ly. *Pullback Grammars are Context-Free.* In proc. of 4th International Conference on Graph Transformation (ICGT'08), LNCS 5214, p366-378. Leicaster (2008).

M. Bauderon, R. Chen, O. Ly. *Context-free Categorical Grammars.* In proc of 3rd International Conference on Algebraic Informatics (CAI'09), Thessaloniki (2009).

## Graphes Hyperboliques

Nous nous sommes intéressés également au cas des graphes hyperboliques. Cet intérêt est en particulier du au fait que les groupes hyperboliques sont automatiques (voir [17]), et cette propriété est fondamentale pour la preuve de décidabilité du problème de l'isomorphisme des

groupes hyperboliques (voir [96]). En rapport avec les variétés hyperboliques de dimension 3 arborescentes évoquées précédemment, sachant qu'elles sont définies par leurs groupes fondamentaux, nous avons exploré sans parvenir à le résoudre le problème de l'isomorphisme de groupe issus du produits amalgamés d'une infinité de groupes hyperboliques, mais en nombre fini à isomorphisme près.

Dans ce cadre, nous avons également étudié le problème de l'étiquetage de distance, problème largement étudié par C. Gavoille. Il s'agit de considérer une famille de graphe finis, et pour chacun de ces graphes d'assigner une étiquette telle que la donnée des étiquettes de deux sommets, et seulement de ces étiquettes, donne une information suffisante pour donner une approximation de la distance entre ces deux sommets. Nous avons décrit un schéma d'étiquetage comportant des étiquettes de taille $O(\log^2 n)$ pour les graphes à $n$ sommets d'hyperbolicité uniformément bornée permettant d'approximer la distance avec une erreur additive d'ordre $O(\log n)$. De plus, la taille de ces étiquettes est optimale pour toute erreur additive de $n^\epsilon$. Nous avons également montré une borne inférieure d'ordre $\Omega(\log \log n)$ sur le facteur d'approximation : tout schéma d'étiquetage de distance $s$-multiplicatif sur des graphes d'hyperbolicité bornée avec des étiquettes de taille polylogarithmique nécessite que $s$ soit d'ordre $\Omega(\log \log n)$.

C. Gavoille, O. Ly. *Distance Labeling in Hyperbolic Graphs.* In proc. of 16th Annual International Symposium on Algorithms and Computation (ISAAC'05). LNCS 3827, p1071-1081, Sanya (2005).

## Equations de langages

En marge de cela, je me suis intéressé aux équations de langages. Les équations de langages apparaissent naturellement en informatique. Il suffit de penser au lemme d'Arden par exemple, ou bien aux langages hors-contextes qui sont composantes de plus petites solutions d'équations polynomiales. Néanmoins dans ce contexte, même les questions les plus simples et les plus naturelles peuvent se révéler extrêmement compliquées à appréhender. Par exemple l'équation $XL = LX$ où $L$ est un langage rationnel et $X$ est un langage inconnu [1] est déjà très énigmatique, c'est en effet le problème de Conway qui consiste à se demander si le langage maximal qui commute avec un langage rationnel donné est rationnel ou non (voir [26, 22]).

Nous avons considéré l'inéquation $XA \subseteq BX$ où $A$, $B$ et $X$ sont des langages formels, et $X$ est inconnu. M. Kunc a prouvé dans [65] que si $B$ est rationnel alors la solution maximale est également un langage rationnel. Cependant cette preuve est basée sur le théorème de l'arbre de Kruskal, et ne fournit pas de construction effective de la solution. Nous avons montré qu'une telle construction est possible, et de complexité élémentaire, dans le cas ou $A$ et $B$ sont finis, et $\max_{b\in B} |b| < min_{a\in A}|a|$.

J'adresse mes remerciements à G. Sénizergues pour les discussions que nous avons pu avoir à ce sujet. Je poursuis cette étude en collaboration avec Z. Wu, chercheur à la "Chinese Academy of Science".

O. Ly. *A constructive solution of the language inequation* $XA \subseteq BX$. In proc. of Theory and Applications of Language Equations(TALE'07), General Publications series of Turku Centre for Computer Science, 44 p 76-84. Turku (2007).

1. Le produit des deux langages est simplement la concaténation.

# Méthodes Formelles et Sécurité logicielle

Ces travaux ont été fondés par une expérience de près de 3 ans dans l'industrie, initialement dans l'équipe fondée par Roland Moréno chez Bull CP8, puis au fil des fusions/acquisitions, dans la division carte à puce de la société Schlumberger.

## Vérification et certification de systèmes embarqués

La certification est un volet important de l'activité industrielle. Produire et faire état d'un certain degré de confiance et d'assurance dans une technologie est un préalable nécessaire à son déploiement à grande échelle.

Dans ce cadre, les méthodes dites "formelles" visent à offrir un complément aux techniques basées sur les tests. Complément que l'on peut comparer à la démonstration d'un théorème mathématique au regard de l'expérimentation de ses cas particuliers.

La carte à puce, et plus généralement l'embarqué, constitue un terrain très favorable à l'utilisation de ces techniques : Les cartes à puces sont déployées à très grande échelle, et leur domaines d'application sont souvent très sensibles. Eu égard à cela, l'enjeu est fort et justifie l'utilisation de moyens importants pour obtenir des degrés de confiance de très haut niveau. Par ailleurs, si leur complexité est grande, la masse des systèmes considérés reste relativement faible, en comparaison avec le reste de l'industrie informatique.

Dans ce domaine, nous avons expérimenté l'utilisation d'assistant de preuve pour la validation et la vérification de la machine virtuelle java embarquée dans la carte. En particulier, nous nous sommes attaché à produire une démonstration complète de l'isolation assurée par la machine virtuelle des différentes applications java embarquée (propriété de confidentialité). Cette démonstration se base sur le principe de non interférence. Précisément, nous montrons que la quantité d'information qu'une application donnée peut obtenir sur une autre est nulle, sauf autorisation explicite. Cette démonstration, assistée et vérifiée par le système Coq, a nécessité un développement de 30000 lignes de Coq. Notons que cette démonstration a détectée une faille (en fait connu par ailleurs) de la spécification JavaCard 2.1.1.

J. Andronick, B. Chetali, O. Ly. *Using Coq to Verify Java Card Applet Isolation Properties.* In proc of 16th Theorem Proving in Higher Order Logic (TPHOL'2003) LNCS 2758, p335-351 Roma (2003).

Nous avons également étudié les possibilités d'automatiser ce type de vérification d'un point de vue théorique. Un angle d'attaque a été de traduire les méthodes utilisées en certification (par exemple comme celle édictées par les Critères Communs) dans le formalisme de la théorie des graphes. Cela nous a conduit à considérer le problème consistant à vérifier une propriété globale d'un système (typiquement la machine virtuelle java) à partir de la spécification fonctionnelle de ses composants. Cela se rapproche de la vérification compositionnelle qui consiste à vérifier si un ensemble de propriétés structurelles locales implique une propriété globale du système. Nous avons considéré le cas où les propriétés sont exprimées dans la logique monadique du second ordre en ayant pour objet le graphe de flot de contrôle du programme et les appels de fonctions. Nous avons montré que le problème de la compositionnalité est décidable pour les programmes séquentiels dont le graphe de flot de contrôle est de largeur arborescente bornée. Formellement, nous prouvons la décidabilité de la théorie MSO des hypergraphes obtenus par

substitution uniforme d'hyperarcs par des hypergraphes eux-même spécifiés par des formules MSO.

O. Ly. *Compositional Verification : Decidability Issues Using Graph Substitutions.* In proc. of 29th Mathematical Foundations of Computer Science (MFCS'2004) LNCS 3153, p 537-549 Pragues (2004)

## Implantation automatique de contre-mesures

Dans un domaine connexe, nous avons travaillé à automatiser l'implantation de contre-mesures dans le logiciel embarqué dans la carte. Les attaques physiques constituent une menace très importante dans le domaine de la carte. Il s'agit d'attaquer le système par des moyens physiques (laser, power glitch, etc) permettant de modifier des données ou de provoquer des fautes en cours d'exécution, ce qui peut impliquer en modifiant par exemple le compteur de programme de faire sauter le flot contrôle et par exemple court-circuiter certaines vérifications, ou encore de provoquer des fautes de calcul dans un algorithme cryptographique, et de déduire de l'information du comportement obtenu.

Il s'agit de protéger le logiciel contre divers type d'attaque, et en particulier des attaques physiques (laser, power glitch, etc). Les contre-mesures logicielles (d'autres sont matérielles) étaient jusqu'à présent implantées à la main par les développeurs. Nous avons développé un pré-compilateur implantant un certain nombre de contre-mesures de façon automatique. Le pré-compilateur analyse l'ensemble du logiciel et implante par exemples des marqueurs d'exécution ainsi que des points de contrôle. Il assure également l'uniformité du temps d'exécution de certaines parties. Il permet à l'utilisateur de fixer le compromis efficacité / sécurité.

A noter que cet outil a nécessité environ 20000 lignes de code source ocaml, il a donné lieu à un brevet.

M.-L. Akkar, L. Goubin, O. Ly. *Procédé de sécurisation d'un dispositif électronique exécutant un algorithme quelconque contre les attaques par introduction d'erreur*. European Patent 032906885 (Schlumberger, 2003).

Par ailleurs, après avoir mis en œuvre des attaques par analyse différentielle de puissance ("Differential Power Analysis", DPA) sur des composants 8-bits, nous avons développé un outil permettant de simuler la consommation physique d'un exécutable destiné à la carte. L'objectif de ce travail est de fournir un environnement logiciel permettant de tester en amont (avant la phase de masquage) la résistance à certaines attaques (typiquement de type DPA).

Ph. Andouard, O. Ly, D. Rouillard. *VisAA : Visual Analyzer for Assembler.* In proc. of IEEE Int. Conf. on Risks and Security of Internet and Systems (CRISIS'08), Tozeur (2008)

Ph. Andouard, O. Ly, C. Thuillet. *A smartcard power analysis simulator.* In proc. of IEEE Int. Symp. on Trusted Computing and Communication, (TRUSTCOM'09) Vancouver (2009).

## Analyse statique de code exécutable

Les travaux précédents m'ont conduit à considérer plus généralement l'analyse statique de code exécutable.

L'objet des outils de vérification apparentés aux méthodes formelles est souvent de haut niveau. La vérification s'applique souvent sur les modèles de conception d'un système, ou encore sur le code source dans le cas d'un logiciel. Cela permet de détecter des erreurs en amont, tôt dans le cycle de développement. Cependant, certains cas d'application nécessitent de s'intéresser également au bas niveau, c'est-à-dire au code exécutable, la séquence d'instruction effectivement exécutée par le processeur. En effet, certaines attaques physiques par exemple s'attachent essentiellement à l'exécution réelle. C'est le cas des "timing attacks" par exemple qui se basent sur les variations du temps d'exécution. Dans un autre domaine, la production d'anti-virus doit se baser sur les virus eux-mêmes dont on ne dispose que de l'exécutable.

Or, du point de vue des méthodes d'analyse automatique, se focaliser sur le code exécutable comporte de nouvelles difficultés très profondes. Un premier trait terriblement problématique est la présence de branchements dynamiques. Au bas niveau, le code utilise des branchements de flot de contrôle (jump) calculés dynamiquement. En d'autre terme, on ne dispose pas de graphe de flot de contrôle. La seconde difficulté intrinsèque est l'absence de structuration des données. Dans la plupart des architectures, les types élémentaires (entier, chaines, etc) ne sont pas distingués à bas niveau, et encore moins les types structurés. Enfin, les différentes architectures (8-bits, ARM, x86, etc) comportent des variations importantes qui sont difficiles à abstraire.

Nous avons contribué au développement de l'environnement logiciel Insight qui fournit une base pour l'analyse de code binaire. Il fournit le décodage de plusieurs architectures (x86, ARM) en un langage intermédiaire (microcode). Il fournit également une bibliothèque d'analyse statique, et en particulier un interpréteur abstrait générique.

S. Bardin, P. Herrmann, J. Leroux, O. Ly, R. Tabary, A. Vincent. *The BINCOA Framework for Binary Code Analysis.* In proc. of 23rd Int. Conf. on Computer Aided Verification (CAV'2011) - SnowBird (2011).

L'environnement a été utilisé pour l'analyse virale, focalisé sur les virus polymorphes. Il s'agit d'extraire de l'exécutable d'un virus donné sa grammaire de réplication, préalable essentiel au développement d'un anti-virus. Le but de l'outil est de fournir une description de cette grammaire et de produire un détecteur de façon automatisée.

S. Chaumette, O. Ly, R. Tabary. *Automated Extraction of Polymorphic Virus Signatures using Abstract Interpretation.* In proc. of IEEE/IFIP 5th Int. Conf. on Network and System Security (NSS'2011) Milan. (2011).

Par ailleurs, nous avons contribué à développer un système de protection de logiciel consistant à externaliser une partie de son exécution. Nous nous sommes focalisé sur la protection d'application java. Il s'agit d'externaliser certaines parties sensibles du code à bord d'un système de confiance ("trusted device"). Le logiciel est également basé sur de l'analyse de code bas niveau, en l'occurrence de l'analyse le bytecode java, en extrait les parties sensibles et les formate pour une exécution externalisée. La difficulté provient du fait qu'on considère que le système de confiance (typiquement une carte à puce) dispose de très peu de ressources. Ainsi,

les parties à exécuter doivent être partitionnées en morceaux de petite taille. Et ces morceaux de code transitent entre l'hôte et le système de confiance sur un canal observable. S'ils peuvent être chiffrés un à un, leur ordre d'apparition durant l'exécution ne peut pas être occulté. Le système de protection doit donc réorganiser l'exécution de sorte que cette séquence ne révèle pas d'information sensible. Notons que le système de protection, opérationnel, est complet en ceci qu'il prend en compte l'intégralité du langage de bytecode java.

S. Chaumette, O. Ly, R. Tabary. *Automated Software Protection through Program Externalization on Memory-Limited Secure Devices.* In proc. of IEEE/IFIP Int. Symp. on Trusted Computing and Communication (TRUSTCOM'10) p777-784. Hong Kong (2010).

Inversement, nous avons contribué à étudier la possibilité d'exécuter du code sensible à bord de systèmes non garantis.

M. Ben MBarka, F. Krief, O. Ly. *Entrusting Remote Software Executed in an Untrusted Computation Helper*. In proc. of Int. Conf. on Network and Service Security (N2S'2009), Paris (2009)

# Robotique

Mon intérêt pour la programmation embarquée allié à une passion de longue date pour la robotique m'a conduit à concevoir des robots, d'abord dans une démarche privée et personnelle, puis dans le cadre professionnel de la recherche et de l'enseignement, en particulier lors d'un séjour dans l'équipe Flowers.

Mon travail s'est organisé essentiellement autour de la conception de robots humanoïdes, et en particulier autour de problèmes d'équilibre et de locomotion.

La participation à la RoboCup est un moteur très important. Cela m'a conduit à étudier également la robotique autonome, mais aussi les problématiques liées à l'ingénierie de ces systèmes, incluant notamment la sureté de fonctionnement.

Par ailleurs, cet investissement dans la robotique s'est très vite traduit dans mes pratiques pédagogiques au vu de son caractère pluridisciplinaire et de la motivation qu'elle suscite chez les étudiants.

## Robotique Humanoïde

**Rhoban.** Mon premier projet significatif a été le robot humanoïde autonome Rhoban. Ce robot de 27cm de haut est pourvu de 20 degrés de liberté. Il marche, peut se relever et maintient son équilibre. Un point à souligner est qu'il est conçu à partir de matériaux très bas coût. La conception de ce robot nous a conduit à envisager un projet d'entreprise, basé sur la robotique ludique et plus généralement de la robotique de divertissement (entertainment). Notre projet a été lauréat du concours d'entreprise innovante OSEO-ANVAR 2009 dans la catégorie projet émergents.

**Acroban.** La version suivante, baptisée Acroban, a vu le jour en 2009. Nous l'avons développé en collaboration avec P.-Y. Oudeyer, chef de l'équipe Inria Flowers. Acroban est un robot

humanoïde de 70cm doté de 28 degrés de liberté. Il a été conçu au travers de l'étude de plusieurs questions :

- Sa conception mécanique intègre plusieurs mobilités dans le buste, s'inspirant de la colonne vertébrale humaine. Elle intègre cinq degrés de liberté essentiels de la colonne vertébrale humaine (voir [18]). Cette conception ouvre de nouvelles possibilités motrices en terme d'équilibre postural. Par ailleurs, elle élargit de façon significative l'espace de travail (voir également [106]).
- La mécatronique et la mécanique utilisée a un caractère compliant (souple), à l'instar des animaux. Acroban est ainsi un robot semi-passif. Nous intégrons cela dans la conception des primitives motrices (en particulier liées à l'équilibrage et la ant à obtenir des comportements robuste vis-à-vis des variations d'environnement. Nous explorons en particulier les techniques basées sur l'apprentissage, et également l'apprentissage développemental en collaboration avec P.-Y. Oudeyer de l'équipe Flowers.
- Cette compliance permet d'envisager l'interaction physique homme / robot. La souplesse du robot lui permet d'absorber et d' "écouter" les sollicitations physiques dont il fait l'objet, en particulier de la part d'un utilisateur. Cette interaction physique homme/robot est une question naturelle au regard du fait qu'un robot est essentiellement un système informatique doté d'un appareil sensori-moteur.

FIGURE 1 – Acroban (photo P. Fudal ©)

O. Ly, P.-Y. Oudeyer. *Acroban the Humanoid : Playful and Compliant Physical Child-Robot Interaction*. In ACM SIGGRAPH'2010 Emerging Technologies. Los Angeles (2010).

O. Ly, P.-Y. Oudeyer. *Acroban the Humanoid : Compliance for Stabilization and Human Interaction.* In IEEE/RSJ Int. Conference on Intelligent Robots and Systems (IROS'2010) - Video Session - Taipei (2010).

M. Lapeyre, O. Ly, P.-Y. Oudeyer. *Modeling Maturational Constrains for Learning Biped Humanoid Locomotion.* In Proc of Int. Conf. on Development and Learning (ICDL) - Poster session - IEEE 2011 (Frankfurt).

O. Ly, M. Lapeyre and P.-Y. Oudeyer. *Bio-Inspired Vertebral Column, Compliance and Semi-*

*Passive Dynamics in a Lightweight Humanoid Robot.* In proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'2011) - 2011 (San Francisco) (également sélection pour le symposium special démonstration d'IROS'2011)

M. Lapeyre, O. Ly, P.-Y. Oudeyer. *Maturational constraints for motor learning in high-dimensions : the case of biped walking.* In proc. 11th IEEE-RAS International Conference on Humanoid Robots (Humanoid'2011) - 2011 (Bled).

P.-Y. Oudeyer, P. Rouanet, O. Ly. *Exploring robust, intuitive and emergent physical human-robot interaction with the humanoid robot Acroban.* In proc. of 11th IEEE-RAS International Conference on Humanoid Robots (Humanoid'2011) - 2011 (Bled).

Il faut souligner que la conception et la fabrication de cette plateforme a représenté un investissement très important en termes d'ingénierie que nous assumons en collaboration avec H. Gimbert, chercheur au CNRS, L. Hofer et G. Passault, étudiant à l'Enseirb-Matmeca. Outre l'intégralité du logiciel, nous avons conçu en grande partie l'électronique de contrôle, ainsi que la mécanique que nous avons conçue également, et usinée par nos propres moyens. Cette maîtrise globale de la chaine technologique nous a donné une grande marge de manœuvre pour la conception. Sigmaban. Nous avons récemment conçu le robot SigmaBan qui fait suite à Acroban. Il est plus petit qu'Acroban, ce qui lui donne un rapport poids/puissance plus favorable et lui confère plus de dynamisme. SigmaBan intègre des liaisons linéaires verticales comportant *des amortisseurs* dans les hanches. Cela permet d'amortir les chocs de la marche, qui s'en trouve plus robuste, et pousser plus avant le caractère semi-passif. Cela permet également de fournir une mesure empirique de la pression supportée par chaque jambe, en mesurant l'élongation de l'amortisseur. Cela donne notamment une approximation en temps réel du ZMP par un moyen empirique.

Sigmaban embarque une puissance de calcul élevé qui lui permet d'effectuer de l'analyse d'image en temps réel. Point crucial pour participer à la RoboCup. Outre le suivi de balle, nous avons développé plusieurs algorithmes de localisation notamment un basé sur l'apprentissage de lignes remarquables (features).

SigmaBan a été sélectionné et a participé à la RoboCup 2011, compétition mondiale de robotique. Nous avons également été qualifiés pour la RoboCup 2012. Cependant la charge très importante suscitée par notre participation à l'exposition internationale de Yeosu en Corée ne nous a pas permis de participer. Nous projetons de participer en 2013.

**Rhoban Project.** Cette activité a donné naissance à un projet du LaBRI (Rhoban Project) qui regroupe plusieurs permanents (Université Bordeaux 1, CNRS, Enseirb-Matmeca) ainsi que des étudiants, en collaboration notamment avec l'équipe MUSE mené par S. Chamette au LaBRI et l'équipe Méthodes Formelles.

Les principales plateformes robotiques que nous avons conçu sont les suivantes :

- Rhoban [Vidéo] : `http://www.youtube.com/watch?v=7_0YE2kOleY`
- Acroban [Vidéo] : `http://www.youtube.com/watch?v=wQ9xd4sqVx0`
- Sigmaban [Vidéo] : `http://www.youtube.com/watch?v=y8S0wQvJvXc` et [Vidéo] : `http://www.youtube.com/watch?v=H5OYCXZD-FI`
- Les bras anthropomorphes, et notamment la main à 12 degrés de libertés. [Vidéo] :

http://www.youtube.com/watch?v=F9W4NyD5XsI
et [Vidéo] : http://www.youtube.com/watch?v=g7vo01mBixc

En collaboration avec l'équipe Flowers, nous avons également participé à plusieurs manifestations grand public liées à la robotique, contribuant de ce fait à la dissémination scientifique. Nos contributions ont fait l'objet d'un relai médiatique significatif au plan national et international. Mentionnons :

- l'exposition "Mathématiques, un dépaysement soudain", organisée par la fondation Cartier à Paris, pour laquelle nous avons contribué au contrôle bas-niveau des robots.
- l'exposition internationale 2012 (Yeosu, Corée). Exposition de premier plan en Asie. Une salle entière du pavillon français a été dédiée à nos robots. *Environ 600000 visiteurs leur ont rendu visite*.

Au plan pédagogique, mentionnons l'ouverture cette année d'une option de 3ième année pour les élèves ingénieur de l'ENSEIRB-MATMECA pilotée par le projet Rhoban en collaboration avec l'IMS. Les membres du projet contribuent aussi à plusieurs cours centrés sur la robotique (Licence et master d'informatique, IUT d'informatique)

## Robotique Agricole

Outre les activités de recherche académique, Rhoban Project est très investi dans l'innovation et le transfert technologique.

Nous nous intéressons en particulier au marché de la robotique agricole. Depuis les années 50, la mécanisation des processus agricoles a provoqué un accroissement de la productivité inégalé dans l'histoire, avec de nombreuses conséquences, positives comme négatives. Pour autant, de nombreuses pratiques peuvent être améliorées, en particulier du point de vue écologique. Par exemple, les modes de traitements restent très approximatifs, et provoquent dans un but de productivité des dégats colatéraux très importants sur l'éco-système. L'arrosage également produit d'énormes pertes d'eau [2]. On peut penser que des techniques plus intelligentes pourrait apporter des améliorations dans ce domaine. Au vu de la pression croissante que va exercer le problème de l'eau dans les années futures, toute amélioration possible en la matière devra être considérée.

Par ailleurs, plusieurs techonologies issues de la robotique autonome (navigation, segmentation, apprentissage) sont actuellement en plein essort. Pour autant, elle n'offrent pas encore un degré de sureté de fonctionnement suffisant pour être déployées sereinement dans des industries comme le transport, ou encore l'avionique. Cependant, l'agriculture ne nécessite pas le même degré d'assurance. Si les pratiques peuvent être améliorées en moyenne, alors ce type de déploiement devient viable.

Nous collaborons en particulier avec l'entreprise Vitirover située à Saint-Emilion dans le cadre du projet européen VVINNER en collaboration notamment avec l'IMS (laboratoire d'électronique de Bordeaux 1). Il s'agit de concevoir un robot autonome pour l'entretien de la vigne (maintien et contrôle de la végétation). Mentionnons que le projet a reçu tout récemment le prix spécial du jury du salon Vinitech-Sifel 2012, principale manifestation mondiale dédiée aux

2. Plus de 70% de l'eau douce consommée par l'homme l'est dans le cadre de l'agriculture, et sur ces 70%, de l'ordre de 50% est perdue en évaporation ou ruissellement. A cela s'ajoute les traitements pour les maladies directement issues de techniques d'arrosage inappropriées.

technologies de la culture viticole.

Nous collaborons également avec la société ATH spécialisée dans les technologies hydrauliques pour la mise au point d'une machine de plantation viticole automatique, guidée par un système de positionnement satellite. Le premier prototype est depuis peu opérationnel.

La tête du bras est équipée d'un GPS centimétrique. Pour des raisons de contraintes budgétaires, nous avons développé notre propre système de positionnement centimétrique sur la base de techniques de triple différence en utilisant des puces GPS standard à bas coût. Le premier prototype est aujourd'hui opérationnel et sera également présenté au salon Vinitech-Sifel 2012.

Rhoban Project a donné naissance à une entreprise offrant des services de conception robotique dans le domaine de l'entertainment d'une part, et dans le domaine de la robotique industrielle agricole d'autre part.

# Chapitre 1

# Graphes et langages

J'ai choisi de ne relater qu'une partie de mes travaux dans ce domaine, essentiellement relevant d'une activité ou d'un questionnement actuel, bien que non exhaustif. J'y parlerai des équations de langages, pour y exposer la démonstration du théorème 1, page 19 qui, bien qu'incomplet, reste la seule solution effective à l'équation $AX \subset XB$.

## Le mystère des équations de langages

Les équations de langages apparaissent de façon naturelle en informatique. L'exemple du Lemme d'Arden en est une belle illustration, on peut également penser aux langages hors-contextes qui sont les composantes des solutions minimales d'équations polynômiales.

Cependant, même des questions très simples en la matière peuvent se révéler très difficiles, comme par exemple l'équation $XL = LX$ où $X$ est inconnu. C'est le problème de Conway qui pose la question de savoir si le langage maximal commutant avec un langage rationnel donné est aussi rationnel ou non ([26], voir aussi [60, 22, 58, 59]). Beaucoup d'avancées ont été accomplies ces dernières années ([60, 58, 59]). Mais ce problème n'a trouvé une solution que récemment, en fait négative : M. Kunc a prouvé dans [67] (voir aussi [57]) qu'il existe un langage *fini* $L$ tel que la solution maximale de $XL = LX$ n'est pas récursivement énumérable. Beaucoup de classes de langages, très naturelles, ont des caractérisations en termes d'équations (voir [88, 89]).

Par ailleurs, dans [65], M. Kunc a prouvé que la solution maximale de $XA \subseteq BX$ est un langage rationnel dès lors que $B$ est rationnel, et quelque soit $A$. Mais la situation est complexe : si l'on impose à $X$ d'être contenu dans un langage sans étoile, alors la solution maximale de $XA \subseteq BX$ peut devenir non récursivement énumérable (voir [64]). C'est une variation du résultat négatif de [67].

Cependant, la preuve que le solution maximale de $XA \subseteq BX$ est régulier est basée sur le Théorème de l'arbre de Kruskal (voir [63]). En particulier, cette preuve n'est pas constructive, i.e., elle ne donne pas de construction effective de la solution maximale.

*Notre but est de compléter ce manque.* Nous donnons une telle construction, i.e., un algorithme, dans le cas où $A$ et $B$ sont tous les deux finis et sont tels que $\max_{b\in B} |b| < \min_{a\in A} |a|$ : Plus précisemment, notre algorithme construit un automate reconnaissant la solution maximale de $XA \subseteq BX$. De plus, il est de complexité élémentaire.

Comme dans [65], notre preuve prend le point de vue des jeux. On considère un jeu à deux

joueurs : l'attaquant et le défenseur. Les positions du jeu sont des mots. Le jeux consiste en une succession de tours comme suit : premièrement, l'attaquant choisit un mot $a \in A$ et le concatène à la suite de $w$, la position courante du jeu. Si $w.a$ n'a aucun préfixe dans $B$ alors l'attaquant gagne le jeu. Sinon, le défenseur choisit un préfixe de $w.a$ appartenant à $B$ et le retire de $w.a$, cela donne la nouvelle position du jeu pour le tour suivant : $b\backslash w.a$. Le défenseur gagne si le jeu se poursuit une infinité de tours. L'appartenance d'un mot $w$ à la solution maximale de $XA \subseteq BX$ peut être traduite par l'existence d'une stratégie gagnante pour le défenseur sur $w$. (voir [65]).

L'ingrédient principal de notre preuve est le lemme de pompage sur les mots ayant une stratégie gagnante, il sera détaillé dans la section 1). C'est là en particulier que nous utilisons l'hypothèse sur la longueur des mots de $A$ et $B$.

**Préliminaires**

Dans la suite, $\Sigma$ est un alphabet fini. $A$ et $B$ sont des langages finis sur $\Sigma$ tels que

$$\max_{b \in B} |b| < \min_{a \in A} |a|$$

. Soit $w$ un mot, on note $|w|$ la longueur de $w$. Soit $v$ un préfixe (respectivement un suffixe) de $w$. On note $v\backslash w$ (respectivement $w/v$) le mot unique $v'$ tel que $w = vv'$ (respectivement $w = v'v$).

L'ensemble des suites finies d'éléments de $A$ est noté $T_A$, on l'appelle l'arbre complet $A$-déterministe. "A-deterministe" car chaque noeud possède un et un seul fils associé à chaque $a \in A$ ; l'arc associé à ce fils peut être considéré comme étiqueté par $a$. La suite vide, i.e., la racine, est noté $\rho$.

## Le jeu de $XA \subseteq BX$

De façon classique, l'équation $XA \subseteq BX$ peut être traduite dans le cadre des jeux de la façon suivante.

On se donne deux langages $A$ et $B$. On considère le jeu à deux joueurs *Alphonse* et *Bertrand* . Le jeu consiste en une suite éventuellement infinie de tours. Au début de chaque tour, la position du jeu est un mot. Un tour sur une position donnée $w$ est défini comme suit :

1. *Alphonse* choisit un mot $a \in A$ et le concatène à la suite de $w$.
2. Si aucun préfixe de $w.a$ n'appartient à $B$, alors *Bertrand* perd la partie, le jeu s'arrête. Sinon, *Bertrand* choisit un préfixe $b$ de $w.a$ qui est aussi un mot de $B$ et le supprime de $w.a$. Le jeu continue alors sur la position $b\backslash w.a$.

Ainsi *Alphonse* gagne s'il parvient à bloquer *Bertrand* ; et à l'inverse *Bertrand* gagne si le jeu se perpétue en une infinité de tours. On dit alors que *Bertrand* a une stratégie gagnante sur le mot $w$ s'il a une stratégie qui à partir de $w$ permet à *Bertrand* de continuer le jeu quelque soit les coups d' *Alphonse* .

**Lemma 1 (Correspondance jeu / équation)** *Un mot de $w$ appartient à la solution maximale de $XA \subseteq BX$ si et seulement si Bertrand a une stratégie gagnante.*

*Proof.* Voir [66]. □

## Un lemme de pompage pour les stratégies d' *Alphonse*

### Le lemme de pompage

Notons $S$ l'ensemble des préfixes stricts de mots de $B$, incluant le mot vide. On se donne un mot $w$. On dit qu'un mot $s \in S$ est *accessible à travers* $w$ par *Bertrand* si $w$ peut être décomposé sous la forme $b_1 ... b_n s$ où les $b_i$ sont tous des mots de $B$. L'ensemble de tous les éléments de $S$ qui sont accessible à travers $w$ est appelé la *visibility* de *Bertrand* à travers $w$. On le note $\text{Vis}(w)$.
*Remark 1.* Si $\text{Vis}(w) = \emptyset$ alors *Alphonse* possède une stratégie gagnante sur $w$. Mais l'inverse est faux. Un tel $w$ est dit *terminal*.

**Definition 1 (B-relation)** *Soient deux mots $w$ et $w'$. On dit que $w$ et $w'$ sont en $B$-*relation*, ce que l'on note $w \leftrightarrow_B w'$, s'il existe quatre mots $v_1, v_2, v_3$ et $v'_2$ tels que :*

- $w = v_1 v_2 v_3$ *et* $w' = v_1 v'_2 v_3$.
- *Pour tout* $s \in$ *Vis*$(v_1)$, *Vis*$(sv_2) =$ *Vis*$(sv'_2)$.
- $|v_1| \geq N_1$.

*où*

$$N_1 = \ulcorner 2 \frac{(\min_{a \in A} |a|)(\max_{b \in B} |b|)}{\min_{a \in A} |a| - \max_{b \in B} |b|} (2^{|S|^2} + 1) \urcorner$$

Notons que $v_3$ est en réalité superflu dans cette définition. Nous le conservons pour simplifier les notations.
*Remark 2.* Pour deux mots quelconques $w$ et $w'$, $w \leftrightarrow_B w'$ implique que $\text{Vis}(w) = \text{Vis}(w')$.

**Lemma 2** *La relation $\leftrightarrow_B$ est une congruence à droite d'index fini sur l'ensemble des mots de longueur supérieure à $N_1$.*

*Proof.* Pour voir que $\leftrightarrow_B$ est bien une congrence à droite, seule la transitivité n'est pas immédiate : Soit $w \leftrightarrow_B w'$ et $w \leftrightarrow_B w''$. On note (en reprenant les notations précédente) :

- $w = v_1 v_2$ et $w' = v_1 v'_2$ (ici, nous omettrons $v_3$ qui est supposé être ajouté à la droite de $v_2$).
- $w' = \bar{v}_1 \bar{v}_2$ et $w'' = \bar{v}_1 \bar{v}'_2$

Supposons que $|\bar{v}_1| > |v_1|$. Alors, $v_1$ est un préfixe de $w''$. Choisissons $s \in \text{Vis}(v_1)$. Alors $\text{Vis}(sv_2) = \text{Vis}(sv'_2)$. Soit $\bar{S} = \text{Vis}(s.(v_1 \backslash \bar{v}_1))$. Alors $\text{Vis}(sv'_2)$ est l'union de tous les $\text{Vis}(\bar{s}\bar{v}_2)$ pour $\bar{s} \in \bar{S}$. Par ailleurs, tout $\bar{s} \in \bar{S}$ appartient aussi à $\text{Vis}(\bar{v}_1)$. Ainsi, par hypothèse, $\text{Vis}(s)\bar{v}_2 = \text{Vis}(s)\bar{v}'_2$. Et $\text{Vis}(sv_2)$ est donc l'union de tous les $\text{Vis}(\bar{s}\bar{v}'_2)$ pour $\bar{s} \in \bar{S}$, qui est à son tour égal à $\text{Vis}(s(v_1 \backslash \bar{v}_1).\bar{v}'_2)$. Cela prouve que $w$ et $w''$ sont en B-relation avec $v'_2 = (v_1 \backslash \bar{v}_1).\bar{v}'_2$.

Pour vérifier que $\leftrightarrow_B$ est d'index fini, il suffit de remarquetr que tout mot suffisamment grand est équivalent à un mot plus petit de taille uniformément bornée, qui en est préfixe. Pour obtenir cela, on utilise un argument élémentaire de comptage, basé sur le fait que $S$ est un ensemble fini et donc n'a qu'un nombre fini de sous-ensembles.

Ajoutons pour finir que la congruence est calculable.

□

Soit $\sigma$ une stratégie pour $Alphonse$ (respectivement $Bertrand$) sur un mot $w$. Une *$\sigma$-séquence finie de coups* dans le jeu est une séquence finie de tours
$(a_1, b_1), (a_2, b_2), \ldots, (a_n, b_n)$ où $Alphonse$ (respectivement $Bertrand$) joue selon $\sigma$. Cela veut dire que chaque $a_i$ de la séquence est choisit selon les $b_j$ précédents pour $j < i$ en accord avec $\sigma$. Les $b_i$ ne sont pas spécifiés et sont variables.

Une *stratégie forte* pour *Alphonse* (respectivement *Bertrand* ) est une stratégie dans le jeu modifié de telle sorte que *Alphonse* (respectivement *Bertrand* ) peut jouer plusieurs mots de $A$ (respectivement $B$) au même tour. Formellement, une stratégie forte pour *Alphonse* (respectivement *Bertrand* ) est une stratégie dans le jeu définie par le couple$(A^+, B)$ (respectivement $(A, B^+)$) au lieu de $(A, B)$. Notons que si *Alphonse* dispose d'une stratégie forte gagnante, alors il dispose d'une stratégie gagnante, cela est vrai également pour *Bertrand* . En effet, étant donné une stratégie forte gagnante, on peut gagner le jeu normal en maintenant une file (FIFO) de coups. On utilise le concept de stratégie forte de telle sorte à alléger la description formelle des stratégies gagnantes.

**Lemma 3 (Lemme de pompage)** *Soient deux mots $w$ and $w'$ en $B$-relation, et une stratégie $\sigma$ pour Alphonse sur $w$. Alors il existe un entier $L$ et une stratégie forte $\sigma'$ pour Alphonse sur $w'$ avec la propriété suivante : Quelques soient les coups de $Bertrand$, en suivant $\sigma'$, en moins de $L$ tours :*

- *Soit $Alphonse$ gagne*
- *Ou bien il conduit le jeu de $w'$ jusqu'à un nouveau mot $v'$ tel qu'il existe une $\sigma$-séquence finie non vide de coups conduisant le jeu de $w$ à un nouveau mot $v$ qui est en $B$-relation avec $v'$.*

*L'entier $L$ et la stratégie $\sigma'$ ne dépendent que de $\sigma$, $w$ et de $w'$.*

Avant de rentrer dans la preuve, énonçons la conséquence principale de ce résultat :

**Theorem 1** *Soit $w$ and $w'$ deux mots en $B$-relation. Alors $Alphonse$ dispose d'une stratégie gagnante sur $w$ si et seulement si il en a une sur $w'$.*

*Proof.* En effet, on suppose que $Alphonse$ dispose d'une stratégie gagnante sur $w$. On peut alors construire une stratégie gagnante sur $w'$ de la façon suivante :

Le lemme 3 nous fournit une stratégie $\sigma'$ et un entier $L$. $Alphonse$ commence par jouer selon cette stratégie. Selon le lemme, après un nombre fini de tours, inférieur à $L$ :

- Soit $Alphonse$ gagne. C'est que que nous voulions et la stratégie $\sigma'$ s'arrête alors.
- Ou bien, il conduit le jeu à un mot $v'_1$ et le lemme nous donne une $\sigma$-séquence de coups conduisant de $w$ à un nouveau mot $v_1$ qui est en $B$-relation avec $v'_1$.

La stratégie $\sigma$ est encore gagnante sur ce nouveau mot $v_1$, on recommence alors le processus avec les mots $v_1$ et $v'_1$. Ainsi de suite.

En suivant cela, on construit une suite $v_1, v_2, \ldots, v_k, \ldots$ de mots qui sont les position des coups d'un jeu où $Alphonse$ suit $\sigma$. Notons que chaque couple $v_i$, $v_{i+1}$ est séparé par au moins un tour. Et en fait, plusieurs tours. En particulier, quand le jeu arrive au mot $v_k$, au moins $k$ tours ont été joués dans le jeu. Par ailleurs, observons que $\sigma$ est finie en tant que stratégie de $Alphonse$. Cela implique qu'il existe un entier $L_\sigma$ tel que $Alphonse$ gagne à coup sûr en moins de $L_\sigma$ tours à partir de $w$. Ainsi $k$ est aussi borné par $L_\sigma$. Cela implique que notre processus s'arrête après au plus $L_\sigma$ tours, ce qui veut dire que $Alphonse$ gagne en au plus $L_\sigma$ tours. □

Dans la preuve du lemme 3 nous avons besoin du concept suivant :

**Definition 2 (Boucle d'attente)** *Soit $w$ un mot donné. On définit* une boucle d'attente *comme une décomposition $w = w_1w_2w_3w_4$ de $w$ en 4 facteurs telle que pour tout $s \in$ Vis$(w_1)$, Vis$(sw_2) =$ Vis$(sw_2w_3)$ et $w_3$ est non vide.*

**Lemma 4 (B-Relations et boucles d'attentes)** *Soit $w$ un mot donné, et soit $w = w_1w_2w_3w_4$ une boucle d'attente telle que $|w_1| \geq N_1$. Alors $w$ est en $B$-relation avec tous les mots de $w_1w_2w_3^*w_4$.*

*Proof.* Soit donc $w' = w_1w_2w_3^kw_4$ pour un entier $k$. Selon les notations de la définition 1, définissons $v_1 = w_1$, $v_2 = w_2w_3$, $v_2' = w_2w_3^k$ et $v_3 = w_4$. Premièrement observons que $|v_1| > N_1$.

Nous prouvons cela pour tout $s \in \text{Vis}(v_1)$, $\text{Vis}(sv_2) = \text{Vis}(sv_2')$ par récurrence sur $k$. Pour $k = 0$, il n'y a rien à prouver : C'est l'hypothèse du lemme. Supposons donc la propriété vraie pour un $k \geq 0$. Par récurrence, nous avons prouvé que pour tout $s \in \text{Vis}(v_1)$, $\text{Vis}(sw_2w_3^k) = \text{Vis}(sw_2w_3^{k+1})$.

Soit donc $s \in \text{Vis}(v_1)$. Soit $s' \in \text{Vis}(sw_2w_3^k)$. Montrons que $s' \in \text{Vis}(sw_2w_3^{k+1})$. Soit $b_1, \ldots, b_n \in B$ tel que $b_1 \ldots b_n s' = sw_2w_3^k$. Soit $n'$ le plus grand index tel que $b_1 \ldots b_{n'}$ soit un préfixe de $sw_2$. Et soit $s'' = b_1 \ldots b_{n'} \backslash sw_2$ ; $s''$ appartient à $\text{Vis}(sw_2)$. Par ailleurs, par hypothèse, $\text{Vis}(sw_2) = \text{Vis}(sw_2w_3)$. Ainsi, il existe $b_1', \ldots, b_{n''}'$ tel que $b_1' \ldots b_{n''}' \backslash sw_2w_3 = s''$. Finalement, on obtient que

$$b_1' \ldots b_{n''}' b_{n'+1} \ldots b_n s' = sw_2w_3w_3^k = sw_2w_3^{k+1}$$

ce qui veut dire que $s' \in \text{Vis}(sw_2w_3^{k+1})$. Ce que nous voulions. La réciproque est similaire. □

L'existence de boucle d'attente est donnée par le résultat suivant, basé sur un argument simple de comptage utilisant le fait que les ensemble de visibilité sont des sous ensemble de $S$.

**Lemma 5 (Existence de boucle d'attente, Version 1)** *Pour tout mot $w$ et pour tout préfixe $w_1$ de $w$ tel que la longueur de $w_1 \backslash w$ est supérieure à $2^{|S|^2}$, il existe une boucle d'attente de forme $w = w_1w_2w_3w_4$.*

Nous énonçons ce lemme par soucis de clarté, mais en fait, nous utiliserons une version plus précise :

**Lemma 6 (Existence de boucle d'attente, Version 2)** *Soit $w$ un mot, et soit $w = c_1c_2 \ldots c_n$ une décomposition de $w$ en $n$ facteurs, où les $c_i$ sont des mots. Soit $n_1$ tel que $n - n_1 \geq 2^{|S|^2}$. Alors $w$ a une boucle d'attente $w = w_1w_2w_3w_4$ telle que $w_1 = c_1c_2 \ldots c_{n_1}$ et les autres $w_i$ sont des concaténations de $c_i$. Formellement : pour $i = 1, \ldots, 4$, $w_i = c_{n_{i-1}+1} \ldots c_{n_i}$, où $n_0, n_2, n_3$ et $n_4$ sont tels que $0 < n_1 < n_2 < n_3 \leq n$, $n_0 = 0$ et $n_4 = n$.*

*Proof of Lemma 3.*

On décrit une stratégie $\sigma'$ de $w'$, tours après tours. Pour faire cela, on décrit une partie $\mathcal{G}'$ où les coups de *Bertrand* sont génériques, et faisant cela, on décrit tours après tours comment *Alphonse* doit jouer. Pendant cette description, nous utiliserons $\sigma$ comme un *oracle* à qui l'on fournira des coups de *Bertrand* et qui nous indiquera ce que $\sigma$ suggère pour les coups d'*Alphonse* .

Avant tout, observons que $Bertrand$ doit jouer au moins $\llcorner \frac{N_1}{\max_{b \in B} |b|} \lrcorner$ tours avant d'effacer complètement $v_1$ (nous conservons les notations de la définition 1 où $w = v_1v_2v_3$ et $w' = v_1v_2'v_3$). Et par ailleurs, à partir de la définition de $N_1$ et du fait que $\min_{a \in A} |a| > \max_{b \in B} |b|$ on obtient :

$$\frac{N_1}{\max_{b \in B} |b|} \geq \frac{N_1}{\min_{a \in A} |a|} + 2(2^{|S|^2} + 1) \tag{1.1}$$

Définissons

$$N_1' = \ulcorner \max_{b \in B} |b|(2^{|S|^2} + 1) \urcorner$$

Il y a trois étapes dans la stratégie :

**1.** Informellement, la première étape commence au début et continue jusqu'à ce que le mot obtenu en concaténant tous les coups d' *Alphonse* soit suffisamment long, de taille supérieure à $N_1 + N_1'$.

Selon la remarque précédente, pendant cette phase, les coups de *Bertrand* restent dans $v_1$, car $v_1$ est supposé suffisamment grand (voir plus loin). Pour ces coups, ainsi, il n'y a pas de différence entre $w$ et $w'$ qui ont tous les deux $v_1$ comme préfixe commun. Alors $\mathcal{G}'$ peut être considéré comme un jeu $\mathcal{G}$ sur $w$ et *Alphonse* suit $\sigma$.

Ainsi, précisemment, la première étape consiste en les $n_1$ premier coups du jeu $(a_1, b_1), (a_2, b_2), \ldots, (a_{n_1}, b_{n_1})$ où $n_1$ est tel que $|a_1 a_2 \ldots a_{n_1 - 1}| \leq N_1 + N_1' < |a_1 a_2 \ldots a_{n_1}|$. Notons que $n_1$, i.e., le moment où l'étape 1 se termine, dépend des coups de *Bertrand* et de la stratégie $\sigma$ qui dit à *Alphonse* comment jouer. Cependant, on peut dire que $n_1 \leq (N_1 + N_1')/\min_{a \in A} |a| + 1$. Cela implique que

$$n_1 \leq \frac{N_1}{\min_{a \in A} |a|} + \frac{\max_{b \in B} |b|}{\min_{a \in A} |a|}(2^{|S|^2} + 1) \leq \frac{N_1}{\min_{a \in A} |a|} + 2^{|S|^2} + 1$$

Avec les équations 1.1 de la remarque préliminaire, on peut conclure que $v_1$ n'a pas totalement été effacé, et même plus : il subsiste au moins $2^{|S|^2} + 1$ tours avant cela, ce qui veut dire qu'il existe un mot $v$ de taille supérieure à $\max_{b \in B} |b|(2^{|S|^2} + 1)$ tel que $v_1 = b_1 b_2 \ldots b_{n_1} v$. En particulier, cela justifie le fait que *Alphonse* peut utiliser la stratégie $\sigma$ pour jouer pendant cette étape.

**2.** Dans l'étape 2, *Alphonse* cherche une boucle d'attente. Formellement : *Alphonse* joue selon $\sigma$ jusqu'au tour $n_2$ tel qu'il existe $n_2'$ tel que

$$[w a_1 \ldots a_{n_1}][a_{n_1 + 1} \ldots a_{n_2'}][a_{n_2' + 1} \ldots a_{n_2}]\varepsilon$$

soit une boucle d'attente. On choisit $n_2$ de telle sorte à ce qu'il soit minimal pour cette propriété. Grâce au lemme 6, car la longueur de $v$ est supérieure à $\max_{b \in B} |b|.(2^{|S|^2} + 1)$, nous somme certains que $n_2$ apparait avant que $v$ soit totalement effacé. En particulier, *Alphonse* peut encore utiliser $\sigma$ pour jouer pendant cette phase.

**3.** Pendant l'étape 3, *Alphonse* ne suit plus la stratégie $\sigma$. Il joue la séquence $a_{n_2' + 1}, \ldots, a_{n_2}$ en boucle jusqu'à ce que *Bertrand* ait été pratiquement effacé $v_2'$, i.e., jusqu'au tour $n_3$ qui est tel que $(b_1 \ldots b_{n_3}) \backslash v_1.v_2' \in \text{Vis}(v_1.v_2')$ où $b_{n_2 + 1}, \ldots, b_{n_3}$ sont tous les coups de *Bertrand* pendant l'étape 3. Dans ce qui suit, $(b_1 \ldots b_{n_3}) \backslash v_1.v_2'$ est noté $s'$. Notons qu'à ce moment là, *Alphonse* pourrait entrer dans une boucle, i.e., il pourrait jouer $a_i$ avec $n_2' + 1 \leq i < n_2$. Alors, quelques soient les coups de *Bertrand* , *Alphonse* finit la boucle courante. Cela conduit le jeu à un tour $n_4$ tel que $a_{n_4} = a_{n_2}$.

Notons que pendant que *Alphonse* termine sa boucle, ce qui prend au plus$2^{|S|^2} + 1$ tours de jeu, il peut arriver que *Bertrand* efface $v_3$ et commence à effacer les premiers coups de *Alphonse* , i.e., les coups de l'étape 1. Cependant, l'étape 1 assure que les $n_1$ premiers coups d' *Alphonse*

forment un mot de taille supérieure à $N_1 + N_1'$. Ainsi, *Bertrand* doit quitter au moins un mot de taille $N_1$ à partir de $a_1 a_2 \dots a_{n_1}$.

La succession des coups que nous avons décrit est la suivante :

$$(a_1, b_1), (a_2, b_2), \dots, (a_{n_1}, b_{n_1}), \dots, (a_{n_2'}, b_{n_2'}), \dots$$

$$\dots (a_{n_2}, b_{n_2}), \dots, (a_{n_3}, b_{n_3}), \dots, (a_{n_4}, b_{n_4}).$$

Soit $n_3'$ le plus grand entier tel que $n_2 \leq n_3' \leq n_3$ et $(b_1 \dots b_{n_3'}) \backslash v_1 \in \text{Vis}(v_1)$. Dans la suite, $(b_1 \dots b_{n_3'}) \backslash v_1$ est noté $s$. On a
$(b_{n_3'+1} \dots b_{n_3}) \backslash s.v_2' = s'$. Ainsi $s' \in \text{Vis}(s.v_2')$. Par ailleurs par définition de la $B$-relation, $\text{Vis}(s.v_2') = \text{Vis}(s.v_2)$. Ainsi $s' \in \text{Vis}(s.v_2)$, et donc il existe $\bar{b}_1, \dots, \bar{b}_m$ tel que $\bar{b}_1 \dots \bar{b}_m s' = s.v_2$.

Maintenant, considérons le jeu $\mathcal{G}$ sur $w$ défini comme suit : dans ce jeu, on considère la séquence de coups de *Bertrand* définie par

$$b_1, \dots, b_{n_2}, \dots, b_{n_3'}, \bar{b}_1, \dots, \bar{b}_m, b_{n_3+1}, \dots, b_{n_4}$$

Considérons la séquence de coups d' *Alphonse* correspondante selon la stratégie $\sigma$ : $\bar{a}_1, \dots, \bar{a_{m'}}$ où $m' = n_3' + m + n_4 - n_3 + 1$. Les $n_2$ premiers coups $\bar{a}_i$ sont exactement les $a_i$ que nous venons de définir pour $\sigma'$ aux étapes 1 et 2, puisqu'à ces étapes, *Alphonse* utilise en fait $\sigma$.

Revenons à la définition de $\sigma'$ sur le jeu $\mathcal{G}'$. Rappelons que nous en sommes au tour $n_4 + 1$, et *Alphonse* s'apprête à jouer. Nous définissons son coup par la concaténation de $\bar{a}_{n_2+1} \dots \bar{a}_{m'}$, notons le $a_{n_4+1}$. Rappelons que nous définissons $\sigma'$ comme une stratégie forte. Cela veut dire que *Alphonse* joue dans le jeu $\mathcal{G}'$ comme s'il était dans $\mathcal{G}$. Soit $b_{n_4+1}$ le coup suivant de *Bertrand* .

Observons maintenant que si $w'$ n'a pas totalement été effacé, il reste le même mot dans $\mathcal{G}$ et dans $\mathcal{G}'$ pour compléter la passe, qui est $(b_1 \dots b_{n_4} b_{n_4+1}) \backslash w'$, notons le $u$.

La description de $\sigma'$ s'arrête ici. Pour conclure, remarquons que si à un moment donné, *Bertrand* ne peut plus jouer car le mot courant n'a pas de prefixe dans $B$, alors *Alphonse* gagne le jeu et la stratégie $\sigma'$ se termine.

Il reste à montrer que les positions du jeu $\mathcal{G}$ et $\mathcal{G}'$ sont des mots en $B$-relation. Pour voir cela, on applique le lemme 4.

Supposons que $w'$ n'a pas été totalement effacé et qu'il reste un mot $u$ comme défini plus haut. Nous obtenons finalement une position pour $\mathcal{G}'$ de la forme :

$$\overbrace{u a_1 a_2 \dots a_{n_1}}^{w_1} \overbrace{a_{n_1+1} \dots a_{n_2'}}^{w_2} \dots$$

$$\dots \overbrace{(a_{n_2'+1} \dots a_{n_2}).(a_{n_2'+1} \dots a_{n_2}) \dots (a_{n_2'+1} \dots a_{n_2})}^{w_3^*} \overbrace{\bar{a}_{n_2+1} \dots \bar{a}_m'}^{w_4} \tag{1.2}$$

où $\bar{v}_2' \in a_{n_1+1} \dots a_{n_2'} (a_{n_2'+1} \dots a_{n_2})^+$. Dans $\mathcal{G}$ on obtient une position de la forme :

$$\underbrace{u a_1 a_2 \dots a_{n_1}}_{w_1} \underbrace{a_{n_1+1} \dots a_{n_2'}}_{w_2} \underbrace{a_{n_2'+1} \dots a_{n_2}}_{w_3} \underbrace{\bar{a}_{n_2+1} \dots \bar{a}_m'}_{w_4} \tag{1.3}$$

Les mots $w_1$, $w_2$, $w_3$ et $w_4$ définis plus haut satisfont par construction les conditions du lemme 4. Ils ont été choisi en effet pour construire une boucle d'attente. De plus, comme nous avons vu

pendant la description de $\sigma'$, si $w'$ a été totalement effacé, alors la partie restante de $a_1 a_2 \dots a_{n_1}$ est toujours de taille supérieure à $N_1$.

Pour conclure la preuve, il reste à donner une borne sur le nombre de tours accomplis. Observons que pendant les trois étapes de jeu, $w'$ peut être effacé et au plus $2^{|S|^2} + 1$ tours peuvent être joués après (pour terminer la boucle). Cela conduit à la borne suivante :

$$L = \frac{|w'|}{\min\limits_{b \in B} |b|} + 2^{|S|^2} + 1$$

□

## Construction effective de la solution

Pour revenir à l'équation, on considère la congruence droite du lemme 2 sur les mots de taille supérieure à $N_1$. Elle peut être étendue facilement à une congruence droite sur les mots en définissant chaque mot de taille inférieure à $N_1$ comme équivalent à lui même seulement. On obtient alors une nouvelle congruence droite qui est encore d'index fini. Par le théorème 1, tous les mots congruents appartiennent simulanément à la solution maximale de $AX \subset XB$ ou pas.

Considérons l'automate associé ; on choisit alors le plus grand ensemble d'états de cet automate tel que si l'on considère cet ensemble comme l'ensemble des états finaux, alors on obtient un langage solution de $AX \subset XB$.

# Chapitre 2

# Méthodes formelles et sécurité

## 2.1 Propriété de Confidentialité dans le contexte de JavaCard

Les plateformes ouvertes multi-applicatives ont constitué un changement important de paradigme pour le marché des smart cards. Le concept de plateforme *ouverte* consiste à pouvoir modifier les applications à bord de la carte après son émission. L'aspect *multi-applications* caractérise le fait que plusieurs applications peuvent être actives à bord d'une carte. Par ailleurs, la carte est le lien privilégié entre l'opérateur et l'utilisateur, et elle doit préserver la sécurité des transactions, la confidentialité en particulier, tout en ouvrant de nouveaux services.

L'architecture sous-jacente accroit la complexité du système. Les applications embarquées dans la carte peuvent provenir de fabriquants différents. Cela impose un nouveau modèle de sécurité où le fabriquant de la carte doit veiller à la protection de la plateforme contre les applications elles-même, et chaque application doit elle-même être protégée contre les autres applications. Par exemple, une carte peut contenir une application de porte-monnaie électronique, mais également des applications de fidélité provenant d'acteurs différents. Une application de fidélité ne doit pas pouvoir modifier ou encore accéder d'une quelconque façon aux données du porte-monnaie électronique.

Les plateformes ouvertes et les architectures basées sur des machines virtuelles comme JavaCard [1] constituent une solution raisonnable, et de plus offrent un standard d'interopérabilité et de portabilité intéressant. JavaCard en particulier bénéficie des avantages de la technologie Java, éprouvée par le temps, et offre un mécanisme de sécurisation de l'échange de données et de services.

Les travaux décrits ici s'inscrivent dans l'objectif général de démontrer la correction de l'architecture de sécurité de la plateforme JavaCard, en utilisant des méthodes formelles. L'usage de méthodes formelles est motivé par le but d'atteindre le plus haut niveau du certification des Critères Communs (EAL 7). Ce niveau impose l'usage des méthodes formelles pour la modélisation et la vérification du système visé.

Dans ce contexte, nous avons établit la preuve formelle du principe de l'*isolation des applets*. La production de la preuve est semi-automatique (voir plus loin) et sa vérification est automatique.

L'isolation des applets est un concept central pour la sécurité de JavaCard (voir [19]). Il

1. ou plus anciennement MultOS ou encore Windows for Smart Cards

repose sur le modèle classique de bac-à-sable. Cela consiste en deux concepts classiques : *intégrité* et *confidentialité* des données des applets, ces dernières devant être préservées de tout accès non-autorisé en lecture comme en écriture.

L'isolation des applet repose sur plusieurs mécanismes interdépendants : le pare-feu (firewall) de JavaCard, mais également sur l'arithmétique des pointeurs, interdite en Java. Cette interdiction, à son tour dépend de la vérification du typage. En effet, si le typage n'est pas respecté, toute la sécurité s'effondre. Or cette vérification n'est pas faite en temps réel, mais au chargement de l'applet, selon un mécanisme largement non-trivial de vérification statique. Ainsi la sécurité dépend en réalité de la validité d'une architecture complexe et globale, dépassant largement le cadre des quelques règles du pare-feu de JavaCard. Comment être certain qu'une telle architecture ne recèle aucun canal caché ?

A cela s'ajoute le fait que le pare-feu est limité à l'exécution du bytecode, l'implémentation de l'API y échappe totalement, tout comme les opérations assurées par le Card Manager.

Pour englober cette réalité, nous formalisons la propriété de confidentialité en terme de *non-interférence* (voir e.g. [36])) qui capture l'idée de fuite d'information. La formalisation consiste à dire que le comportement d'une applet donnée ne peut être influencé par la valeur d'une donnée à laquelle elle n'a pas accès. Remarquons que ce concept exclue la vérification du mécanisme d'interface partagée ("shareable interface") qui autorise l'interaction entre les différentes applets sous leur propre contrôle, et donc leur responsabilité. L'isolation des applets telle qu'elle est conçue dans JavaCard délègue à une applet sa propre protection dès lors que cette dernière utilise le mécanisme d'interface partagée.

Nous avons mené la formalisation, le développement et la vérification de la démonstration de la propriété d'isolation au moyen de l'assistant de preuve Coq (voir [9]).

Le principe de Coq est de décrire le formalisme dans un langage traduisant les différents concepts mathématiques et en particulier les preuves dans la théorie des types (plus précisemment sous forme de termes du $\lambda$-calcul), c'est l'isomorphisme de Curry-Howard. Le développement consiste en l'écriture d'un script guidant le système pour établir la preuve formelle recherchée. Notons que dans notre cas, le script décrivant la démonstration a nécessité environ 30 000 lignes de code. En effet, il s'agit d'établir une preuve complète, sans omettre aucun détail, l'essentiel du travail étant d'établir des stratégies de preuve pour guider le système dans sa recherche.

Notre preuve se base sur une modélisation formelle complète de Java Card établie dans le cadre du projet FORMAVIE [2] (voir [13]). L'ensemble des spécifications de JavaCard, a été traduite de façon formelle dans le langage de Coq (le *Calculus of (Co)Inductive Constructions*)

Ces travaux ont été conduit en collaboration avec J. Andronick, chercheuse au Nicta (Australie).

## 2.2 Compositionnalité

En parallèle à la construction de preuve semi-automatisée, nous avons étudié les moyens d'automatiser intégralement le processus de preuve.

Pour ce faire nous nous sommes intéressé à la *vérification compositionnelle* de programmes séquentiels. Il s'agit de décider de façon algorithmique si un ensemble donné de propriétés

2. FORMAVIE a été un projet OPPIDUM incluant notamment Schlumberger Systems et l'INRIA.

structurelles (i.e. statiques) des fonctions d'un programme implique bien une propriété comportementale (i.e. dynamique) globale du programme. Le but est de réduire la vérification d'une propriété globale du système à la vérification de plusieurs propriétés locales et indépendantes de ses composantes, *a priori* plus aisées à vérifier.

La vérification compositionnelle est un problème classique de la théorie des systèmes concurrents (voir *e.g.* [40, 1, 69]). Dans notre contexte, nous considérons des programmes séquentiels et nous restreignons notre étude à des propriétés comportementales et structurelles ayant trait au flot de contrôle du programme ainsi qu'aux appels de fonctions apparaissant pendant l'exécution.

Ce cadre de travail s'appuie sur le modèle de programme indépendant de tout langage introduit dans [52] pour les propriétés de sécurité classiques. Il a été étudié dans [11, 10, 107] (voir aussi [31]) pour le raisonnement compositionnel dans le cadre des cartes à puce multi-applications ouvertes.

Comme dans *e.g.* [109, 3, 85] pour les systèmes concurrents, un système de preuve basé sur le $\mu$-calcul modal a été mis au point dans [11] pour le cadre de travail que nous considérons. Mais la question de la décidabilité est restée ouverte.

Elle a reçu une réponse positive dans [107] en restreignant l'étude à la logique de simulation.

Nous avons considéré les propriétés exprimé dans la *logique monadique du second ordre* (voir *e.g.* [41]), et avons donné pour tout entier donné $k$ un algorithme de décision pour le problème de la compositionnalité pour les programmes dont les graphes de flot de contrôle sont de largeur arborescente bornée par $k$.

Ce résultat contribue aux travaux décrit précédemment sachant que la logique monadique du second ordre contient le $\mu$-calcul modal, et a fortiori la logique de simulation.

Formellement, la limitation de notre solution réside dans la condition sur la largeur arborescente [3]du graphe de flot de contrôle. Cependant, nous soutenons que cette limitation est raisonnable : les programmes "raisonnables" sont de largeur arbordescente bornée : il a été montré dans [114] que la largeur arborescente du graphe de flot de contrôle d'un programme en langage C sans "goto" est au plus $6$. Mentionnons également que la largeur arborescente des graphes de flot de contrôle des functions de l'API Java n'excède jamais $5$ et vaut en moyenne $2,7$.

Pour autant, l'algorithme que nous avons défini porte une complexité très forte qui rendrait un projet d'implémentation hasardeux.

La démonstration de ce résultat repose sur une variation du lien classique entre la logique monadique du second ordre (MSO) et les automates d'arbre (voir *e.g.* [112]). D'abord nous remarquons que le problème de la compositionalité peut être reformulé sous la forme d'un problème de satisfiabilité pour des familles d'hypergraphes infinis (en fait $HR$-équationnels) obtenue par substitution *uniforme* d'hyperarcs par des hypergraphes spécifiées par des formules MSO, que nous appellerons *expansions MSO uniformes* . Ces familles sont en fait des ensembles de systèmes de transitions qui encodent le comportement d'un programme fait de fonctions satisfaisant une ensemble donné de propriétés structurelles.

Ensuite, nous prouvons la décidabilité de la théorie MSO des expansions MSO uniformes. La première étape de cette démonstration consiste à traduire le problème en terme de langages d'arbres. Pour ce faire, nous introduisons un nouveau type d'automate sur les arbres infinis que nous appelons *automate composite*. Ces automates encodent les expansions MSO uniformes

3. Trés grossièrement la largeur arborescente est un entier mesurant à quel point un graphe est loin d'être un arbre.

via le concept d'expressions syntaxiques ce qui est possible grâce à la borne sur la largeur arborescente. Puis nous démontrons la décidabilité du problème du vide pour l'intersection des langages d'un automate composite et d'un automate de Rabin. Nous utilisons pour ce faire une variation du point de vue classique voyant les parcours d'automates sur des termes au moyen de jeux et de stratégies (voir *e.g.* [112]).

## 2.3 Outils pour la sécurité

### 2.3.1 Implémentation semi-automatique de contre-mesures

En collaboration avec M.-L. Akkar, à l'époque ingénieur de recherche chez Schlumberger et L. Goubin, professeur à l'universite de Versailles Saint-Quentin-en-Yvelines, je me suis intéressé à automatiser certaines tâches liées à la sécurité des cartes à puces. Nous nous sommes attachés ainsi à améliorer le processus de génie logiciel entourant la conception de carte.

Nous nous sommes focalisés sur la protection du système contre les attaques par injection de fautes, et en particulier à sécuriser le flot de contrôle.

L'injection de faute consiste à perturber la carte pendant l'exécution par des moyens physiques. Il s'agit d'appliquer des modifications ponctuelles de son environnement en utilisant par exemple des pics de courant sur l'alimentation, ou bien des perturbations électromagnétiques, ou encore utiliser les courants de Foucault. L'utilisation d'émission laser sur le microprocesseur est aussi possible. De telles perturbations introduisent des erreurs dans le calcul du microcontrôleur. De la sorte, on peut modifier le compteur de programme, les registres, etc. En particulier, une telle attaque peut perturber le flot de contrôle et faire en sorte que la carte court-circuite certains bouts de code exécutable, par exemple la vérification du code PIN.

La prise en compte de ces attaques est cruciale dans la pratique, elle représente une menace réelle. Nous utilisons une famille de contre-mesure consistant à exercer un contrôle dynamique de la cohérence du flot de contrôle. Le principe est de maintenir l'historique de l'exécution et d'en vérifier régulièrement la validité et la cohérence. Un compromis est fait entre la précision et la complétude de l'information stockée sur l'historique, et la dépense en espace nécessaire à ce stockage. Il peut conduire à stocker une liste complète de labels, ou à l'inverse ne stocker qu'une somme de nombres-repères faisant état du passage du contrôle par tel ou tel point du programme. Par ailleurs, la fréquence des contrôle de cohérence a également un caractère subjectif, variant suivant la sensibilité du code en court d'exécution.

Ces pratiques avait court de façon hétérogène et ad-hoc dans le développement du logiciel embarqué. L'intégration manuelle de telles contre-mesures implique une charge de travail importante, et une maintenance ardue du fait de leur dépendance très forte avec le code.

Notre contribution a été l'élaboration d'un outil semi-automatique pour l'implémentation de ces contre-mesures. L'utilisateur indique à l'outil les emplacements des points de programme à vérifier ; il indique également les emplacements des points de vérification ; enfin, il indique la nature des contre-mesures à intégrer (historique, sommes, etc). Ainsi, l'utilisateur maîtrise les compromis faits en terme d'utilisation d'espace, en terme de maillage des points de programme à contrôler ainsi que des points de contrôle. Puis, de façon entièrement automatisé, l'outil, sous la forme d'un pré-processeur de code C, intègre la contre-mesure. Pour ce faire, il calcule à partir du graphe de flot de contrôle, l'équilibre des points de vérification, et intègre au code source les bouts de code nécessaires.

Le déploiement de cet outil comporte ainsi plusieurs avantages :

- L'automatisation du processus apporte une sureté de fonctionnement significative de l'implémentation. L'implémentation manuelle ardue était souvent source d'erreur et de bug.
- L'automatisation apporte également une économie importante de charge de travail.
- L'utilisateur ne fournit que des informations qualitatives au système, et en particulier aucun calcul. Cela facilite beaucoup la maintenance des contre-mesures au fur et à mesure du développement logiciel. L'amélioration du cycle de développement en est d'autant plus importante.
- Enfin, il apporte une uniformité dans la pratique de l'implémentation de ces contres-mesures.

Notons que l'outil est opérationnel et prend en compte l'intégralité du langage C, il représente plusieurs dizaines de milliers de lignes de code ocaml.

### 2.3.2 Analyse de code bas-niveau

En collaboration avec C. Thuillet et Ph. Andouard, étudiant en thèse sous ma direction et celle de M. Mosbah, nous avons continué ces travaux vers des outils prenant en compte directement le code exécutable.

Certaines attaques physiques sur la carte sont extrêmement sensible à l'implémentation, jusqu'à très bas niveau. C'est le cas des attaques par mesure du temps d'exécution par exemple. Le détail du temps d'exécution d'un programme est totalement abstrait dans les langages de programmation, y compris dans le langage C par exemple.

Si certaines contre-mesures sont elles-même abstraites et agissent indépendamment du langage de programmation, c'est le cas par exemple de la méthode des masques pour se protéger contre la DPA typiquement (voir [38]) d'autres sont très sensibles à l'implémentation, et l'influence des modifications/optimisation de la phase de compilation n'est pas nécessairement maîtrisée.

Ainsi, nous avons mis en oeuvre un outil permettant l'analyse de code exécutable. Pour des raisons pratiques, nous nous sommes focalisé sur le langage 8-bits AVR. Cet outil permet de désassembler un code exécutable destiné à être directement flashé à bord d'un microcontrôleur, et notamment de relever le graphe de flot de contrôle.

A partir de là, nous avons forgé un ensemble d'outils d'analyse permettant par exemple de contrôler l'uniformité du temps d'exécution. Un autre but a été de simuler une attaque DPA (voir Figure 2.2). L'objectif étant dans ce travail d'améliorer le processus de génie logiciel, en permettant de relever une sensibilité à la DPA très tôt dans le cycle de développement, en l'occurrence, bien avant les tests réels. Pour ce faire, l'outil permet de simuler l'exécution de code et de donner une approximation de la consommation (voir Figure 2.1), consommation évaluée selon plusieurs modèles en toute généricité (poids de Hamming ou distance de Hamming), ce qui le différencie de l'outil Pinpas par exemple (voir [42]). Il permet également de simuler des attaques en faute, selon des scénarii décrits par l'utilisateur.

L'outil permet également de distinguer les diverses sources de fuite (bus de donnée, bus d'adresse, opérations particulière) en paramétrant le module de génération de consommation, celui-ci pouvant choisir d'ignorer ou de prendre en compte tel ou tel élément lié à la consommation.

A noter que, toujours dans un soucis d'améloration du génie logiciel, cet ensemble d'outils (écris dans le langage ocaml) est intégré à l'IDE Eclipse.

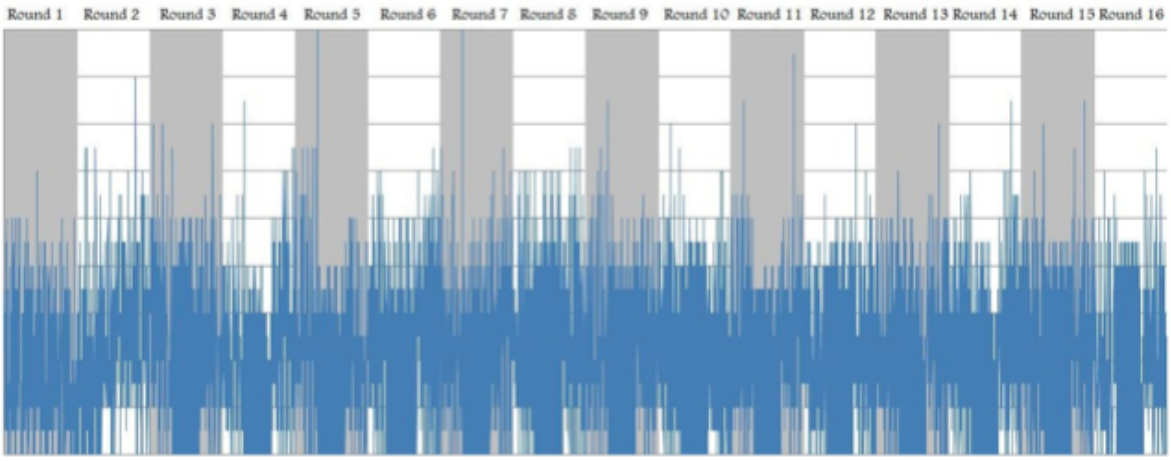


FIGURE 2.1 – On montre ici comme illustration la simulation par l'outil de la puissance consommée par un microcontrôleur exécutant le calcul d'un DES. On y distingue les 16 tours de l'algorithme. La simulation est produite par le modèle du poids de Hamming. On utilise ces données pour simuler une attaque DPA, voir Figure 2.2. L'échelle verticale est de 0,2 Watt/graduation.

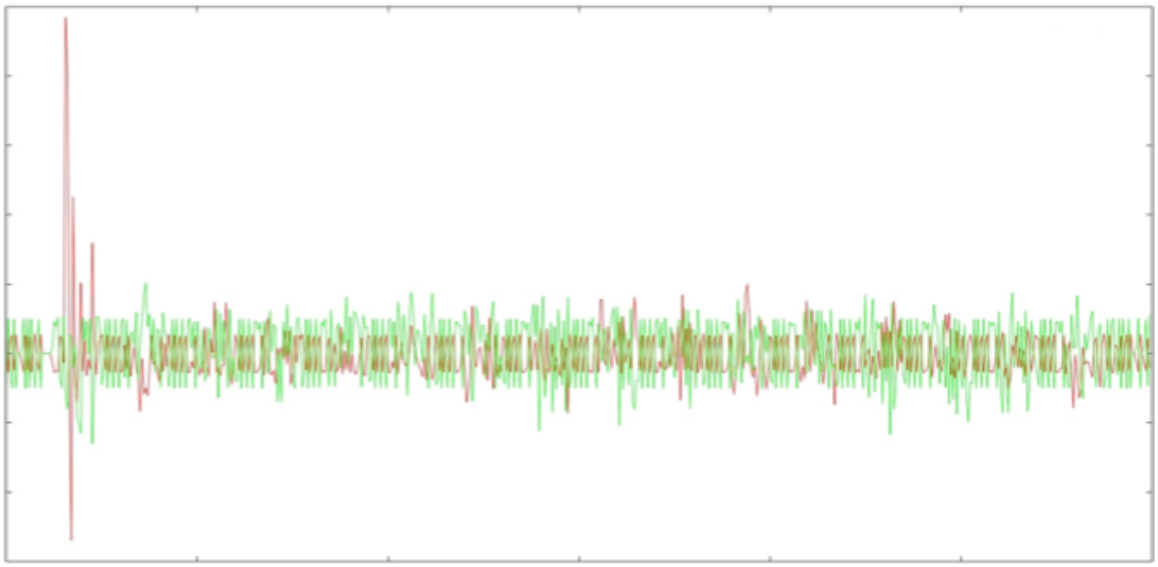

FIGURE 2.2 – On illustre ici un calcul typique d'attaque DPA, la courbe représente une différence de moyenne de consommation. On distingue le pic indiquant un choix valide d'un extrait de clé, très similaire aux résultats que l'on obtient par des mesures réelles. L'intérêt de cette méthode réside dans le fait que le test de résistance d'une implémentation peut être conduit en simulation. La limite reste la granularité du modèle de consommation utilisé, ainsi que son adéquation à la réalité.

## 2.4 Externalisation de code

Dans un autre contexte, mais lié à la carte, en collaboration avec R. Tabary, étudiant en thèse sous ma direction et celle de S. Chaumette, professeur à l'université de Bordeaux, nous nous sommes intéressés à la protection de logiciel contre le piratage et le vol de donnée, problème qui n'a jusqu'à présent pas reçu de réponse satisfaisante complète, et ce malgré des enjeux industriels considérables.

Les solutions de protection logicielle peuvent être divisées en deux catégories : protection logicielle uniquement, et protection assistée par des éléments matériels (du hardware).

Les solutions purement logicielles (obfuscation, watermarking, cryptographie en boîte blanche, etc.) n'ont aucune hypothèse préalable concernant l'environnement d'exécution et en particulier ne le considère pas de confiance. L'obfuscation de code [4] est centrale dans beaucoup de solutions proposées, grâce au fait qu'aucune infrastructure additionnelle n'est requise en général. Cependant cette méthode ne résiste pas réellement au piratage, et en particulier à l'analyse dynamique de code. En réalité, l'obfuscation a été montrée comme impossible pour une large classe de programmes ([6]).

Les solutions de protection utilisant des composants matériels améliore en général la sécurité. On entend par là les méthodes utilisant un token sécurisé en conjonction avec une machine non sécurisée a priori. Le composant matériel apporte des propriétés comme la tamper-résistance, ou encore la difficulté/impossibilité à être reproduit. Mais le prix à payer est un déploiement plus complexe impliquant des composants additionnels. Ce type de protection est classique aujourd'hui.

Plusieurs solutions ont été envisagées jusqu'à présent. Le composant sécurisé peut être utilisé pour cacher un secret comme des clés ou bien des licences (see e.g. [75, 5]). Ou bien de façon plus robuste : des bouts du code exécutable sont exécutée à bord du composant sécurisé. (voir e.g. [104]) ; mentionnons également *eXecute Only Memory* architecture [68]).

Dans ce sens nous proposons un système conçu de la façon suivante : un ordinateur non supposé de confiance et un token sécurisé à ressources limitées. La technologie consiste à externaliser l'exécution des bouts de code sensibles bloc par bloc auprès du token sécurisé, le reste du software étant exécuté par l'ordinateur.

L'externalisation de code exécutable a été étudiée dans [20] où elle est conduite instruction par instruction, et dans [70] où elle est conduite bloc par bloc.

Nous avons conçu ce système sous les contraintes suivantes : En premier lieu, la mémoire du token sécurisé est supposée réduite, ce qui a un effet direct sur la taille des blocs. En second lieu, on suppose que l'attaquant dispose d'un accès au canal de transmission. Ainsi, la suite des blocs utilisée pendant l'exécution doit être conçue de telle sorte à ne révéler aucune information. Chaque bloc est chiffré par un algorithme asymétrique, prévenant les fuites d'information au niveau du bloc lui-même.

Par ailleurs, dans ce contexte, nous nous sommes intéressés au processus de développement. Nous avons conçu un mécanisme générique de protection logiciel : le développeur pointe les données et le code qu'il souhaite protéger, et le système de protection, en boite noire, calcule l'ensemble du code qu'il faut externaliser pour protéger ces ressources par un calcul de dépendance de données ; il effectue les découpages en blocs et précalcule les séquences de blocs qui

4. L'obfuscation de code consiste à transformer un code exécutable en un autre fonctionnellement équivalent mais plus difficile à comprendre et en particulier plus résistant à la rétroingénierie. Il n'y a pas de définition formelle uniformément admise pour ce concept.

devront être suivies lors de l'exécution. Il modifie le logiciel initial pour y ajouter le chargement des blocs et l'ensemble du protocole de communication entre l'ordinateur et le token sécurisé.

Nous avons implémenté cette technologie dans le cadre du bytecode Java. Elle consiste en plusieurs outils : 1) *L'analyseur de code* qui détermine quelles sont les parties du code à externaliser à partir des informations données par le développeur. 2) *Le partitionneur* de code qui part du programme original en bytecode et découpe les blocs qui devront être externalisés. Il injecte dans le programme initial le protocole de communication nécessaire aux échanges avec le token sécurisé. 3) *L'infrastructure d'externalisation :* A l'exécution, les blocs de code sont externalisés vers une machine virtuelle java que nous avons modifiée pour accepter le chargement des blocs et la modification instantanée de son état à chaque nouveau bloc. Le logiciel lui-même est exécuté sur une machine virtuelle *standard* à bord de l'ordinateur. Notons que ce point n'est pas trivial sachant qu'il manque les blocs de code externalisé ; et malgré cela, le programme doit passer la vérification automatique de bytecode, point crucial de la sécurité de Java. C'est là un grand écart nécessaire pour rendre réaliste un scénario de déploiement.

Notre implémentation est opérationnelle, elle englobe l'ensemble des bytecodes Java, inclut le mécanisme des exceptions, du class loader, etc.

## 2.5 Analyse de code

Le développement d'outils, en rapport avec les attaques physiques mais également avec la protection de code, nous a conduit à nous intéresser à l'analyse de code exécutable dans sa globalité.

Après un premier prototype écrit en ocaml focalisé sur l'analyse de code x86, nous avons contribué, notamment avec A. Vincent et E. Fleury, à la mise en place et au développement de l'outil **Insight** , visant l'analyse automatique de code exécutable.

L'analyse de code binaire s'avère utile voire nécessaire dans plusieurs cas. On peut penser à la phase de débuggage dans le développement, ou encore la compréhension de code écrit partiellement en assembleur, comme par exemple certains drivers. D'une façon générale, ce type d'analyse est nécessaire lorsque l'on s'intéresse à des propriétés très bas niveau du programme ou bien lorsque l'on ne dispose pas du code source.

Un exemple de propriété très bas niveau est la résistance aux timing-attacks. Ce type d'attaque exploite le temps d'exécution comme vecteur de fuite d'information. Une contre-mesure classique est d'équilibrer les temps d'exécution sur tous les chemins du graphe de flot de contrôle. Or cette opération ne peut être réalisé ou vérifiée au niveau du code source, elle est totalement liée au programme assembleur issu du processus de compilation.

Un cas d'application appelé à se développer dans le futur est celui des magasins d'applications (AppStores) tels que ceux proposés par Apple ou Android par exemple. Suivant cette organisation, le détenteur du magasin centralise la distribution des logiciels auprès des utilisateurs finaux. Il se porte garant ce faisant d'une certaine qualité de fonctionnement, et en particulier de la sécurité. Or, ce type d'établissement reçoit des milliers de logiciels sans nécessairement disposer du code source. C'est ici que l'analyse automatique peut s'avérer extrêmement utile.

Enfin, un cas d'application évident à mentionner également est l'analyse de virus pour lesquels seul le code exécutable est disponible. Cette question fera l'objet d'une discussion plus approfondie plus loin (voir section 2.6).

L'analyse de programme représente un corpus très important de travaux, supposant en grande

majorité que le programme est donné dans un langage de haut niveau, structurant son flot de contrôle et son utilisation de la mémoire. Cela n'est pas le cas pour l'analyse de code exécutable :

- La mémoire n'est pas structurée. Seuls certains types élémentaires ont cours au niveau de l'exécution, attachés aux opérations et non aux données. Il n'y a pas de type structuré.
- L'adressage indirect (i.e., les pointeurs) y est un paradigme essentiel. En effet, l'adressage est constamment calculé de façon dynamique. Or l'aliasing est écarté des analyses classiques du fait notamment qu'elle élimine toute assurance donné par le typage.
- la possibilité de branchement dynamique est également une caractéristique essentielle (absente des langages de haut niveau). En effet, la structuration d'un programme, en blocs par exemple, est essentielle dans les langages modernes. Elle permet de considérer le graphe de flot de contrôle d'un programme comme une donnée statique. À bas niveau, cela est complètement remis en cause, le graphe de flot de contrôle a un caractère dynamique très embarassant.

Pour autant, certains progrès récents nous permettent de penser que l'analyse de code binaire peut être mise à l'ordre du jour. C'est le cas par exemple de l'émergence de solveurs SMT extrêmement efficaces (e.g. [77, 39]).

A l'heure actuelle les outils focalisés sur le code binaire sont peu nombreux. On peut mentionner osmose [7], mcveto [111], ou encore jakstab [62]. Les autres s'appuie sur d'autres outils (comme e.g. IDA Pro) pour obtenir le code assembleur. Or la phase de désassemblage est elle-même largement non triviale (on peut penser par exemple à du code obfusqué, décalé pour contrer les techniques classiques de type "linear sweep", ou encore du code chiffré dans le cas d'un virus polymorphe).

A l'instar de l'architecture classique d'un compilateur, au coeur de **Insight** se situe un langage intermédiaire simple, indépendant de toute architecture. L'outil propose aujourd'hui le décodage des langages x86-32 et ARM. Pour des raisons évidentes, cela donne plus d'indépendance aux couches d'analyse de l'outil.

**Insight** se base sur plusieurs techniques d'analyse. En tout premier lieu, nous avons développé un système d'interprétation abstraite générique (voir [28]). Dans notre contexte, il a fallu adapter les concepts de l'interprétation abstraite au fait que le graphe de flot de contrôle est dynamique et inconnu à l'avance. Cela a été possible du fait que les transitions de l'interpretation abstraite reposent sur des calculs locaux. Et le processus maintient ses données attachées aux points de programmes. Mais disposer de l'ensemble des transitions du programme à un moment donné n'est pas réellement nécessaire. Ces dernières sont découvertes au fur et à mesure du processus d'interprétation. On produit ainsi une suite croissante de graphes de flot de contrôle compatible avec la nature croissante des domaines attachés aux points de programme, et la convergence des valeurs abstraites des points de programme peut être menée en parallèle avec la convergence des transitions du programme.

En premier lieu, nous utilisons ce système pour effectuer l'interprétation concrète du programme, i.e. la simulation. L'interpréteur concret est central dans le système du fait qu'il définit la sémantique élémentaire du langage intermédiaire pour les autres domaines.

Ensuite, deux autres domaines ont été implémentés : les intervalles et les ensembles de valeurs (ou *K-sets*, voir e.g. [8]). Interpréter les valeurs par des intervalles, comme l'exemple historique de [28], est utilisé classiquement pour résoudre les conditions de test typiquement. Les K-sets consistent en des ensembles finis de valeurs de cardinalités uniformément bornées

(i.e. l'opération le "widening" est déterminée par la cardinalité). On utilise ce domaine typiquement pour collecter les cibles possibles pour un branchement dynamique ; on l'utilise également pour générer des cas de test.

Mentionnons que nous avons également implémenté un outil d'*analyse de dépendance de données*. Il est en particulier nécessaire pour les opérations de "slicing", i.e., l'extraction de l'ensemble des instructions dont dépend une $l$-valeur donnée à un endroit précis du programme (une $l$-valeur localisée). Etant donné une $l$-valeur localisée, l'algorithme calcule les dépendances par une analyse locale à rebourd de l'exécution. Ces dépendances sont codées par des formules sans quantificateur distinguant les ensembles possibles de $l$-valeurs dont dépend la $l$-valeur initiale. L'explosion combinatoire est contrôlée par un mécanisme de simplification des formules logiques d'une part, et d'autre part sur un mécanisme de widening opérant des unions entre les divers ensembles possibles.

Enfin, nous avons intégré un mécanisme de plus petite précondition ("weakest precondition" ou simplement "WP"). Il étend la méthode introduite dans [34]. Il n'est pas encore adapté à la nature dynamique du graphe de flot de contrôle. Cependant, il est utile pour certaines analyses locales.

## 2.6 Application à la virologie

Depuis la création des premiers virus, la technologie des malwares est devenue de plus en plus complexe et avancée, et décrire l'enjeu de la protection contre les virus est superflu. A l'heure actuelle, les virus polymorphes sont probablement les plus difficiles à détecter. Les deux types de virus peuvent muter en une infinité de copies fonctionnellement équivalentes mais syntaxiquement différentes. On notera $L_V$ l'ensemble de ces copies pour un virus $V$ donné.

On considère les virus polymorphes. Un tel malware est formé de trois parties : le corps du virus, le système de mutation et le décrypteur (voir Figure 2.3). A l'infection, le virus polymorphe $V$ se réplique en une nouvelle copie $V' \in L_V$ de la façon suivante : il génére une clé symétrique aléatoire et chiffre avec cette clé le corps du virus ainsi que le système de mutation ; puis il crée un nouveau décrypteur intégrant cette nouvelle clé, et se sert enfin du système de mutation pour faire muter le décrypteur. Le langage du virus est en fait le langage du décrypteur.

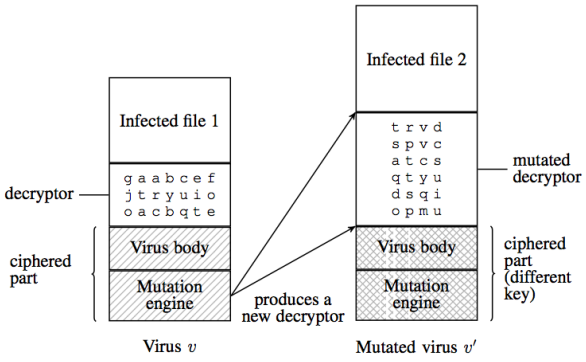


FIGURE 2.3 – Virus Polymorphe

Malgré de nombreux progrès ces dernières années, la détection des virus polymorphes reste

très difficile. Le principe le plus utilisé pour la détection est basé sur l'analyse de signatures définies par des expressions régulières (voir [61]). Le but étant de reconnaitre un sur-langage de $L_V$ approximant ce dernier le plus précisemment possible.

Cependant, cette technique doit faire face à deux problèmes :

- Le nombre de nouveau virus est très important (plusieurs milliers par jours). Or la création de nouvelles signatures repose sur des techniques à la fois manuelle et donc extrêmement couteuses et d'apprentissage automatique, et il est connu qu'en toute généralité, l'apprentissage de langages formels hors-contextes, et même rationnels n'est pas possible à partir de corpus d'exemples positifs seulement.
- La sophistication croissante des virus, et en particulier leur mode de mutation rend les techniques à base d'expressions rationnelles de moins en moins efficaces, tendant à accroître le nombre de faux positifs.

En collaboration avec R. Tabary et S. Chaumette, nous proposons une nouvelle approche consistant 1) à engendrer des signatures à base de grammaires hors-contextes, 2) à automatiser une grande partie du processus d'extraction de signature en se basant sur l'analyse statique du code exécutable du virus (et notamment sur la mise en oeuvre de l'outil **Insight** ). Cette technique comporte plusieurs avantages :

- La sur-approximation par un langage hors-contexte est plus prometteuse, car plus précise que par un langage rationnel.
- L'extraction de la signature est possible à partir d'un seul exemple du virus.
- Enfin, l'extraction semi-automatisée de la signature induit une économie substantielle.

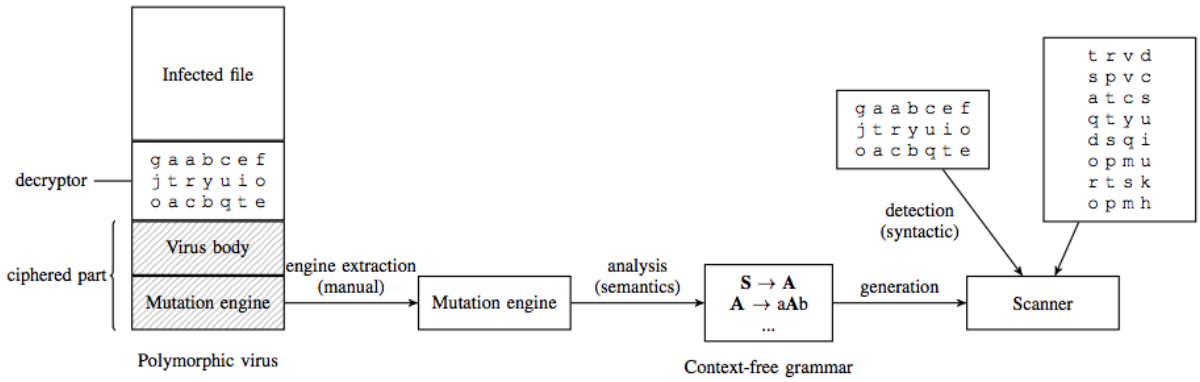


FIGURE 2.4 – Détection de virus polymorphe

La méthode de détection se déroule comme suit (voir Figure 2.4) :

- L'utilisateur doit extraire, par une analyse dynamique, le code source du module de mutation. C'est une opération courante, mais néanmoins non-triviale. Il faut noter que cette opération semble difficile à automatiser.
- Puis, l'analyse bas-niveau calcule le graphe de flot de contrôle, et y inscrit le graphe d'appel des fonctions. Il faut souligner qu'actuellement, le système de prend pas en compte une obsfuscation des appels de fonction. Pour la plupart des exemples connus, ce graphe d'appel de fonction, porte l'essentiel de la structure (récursive) de la grammaire de muta-

tion.

- Enfin, le système extrait de chaque fonction les règles de mutation, en définissant les terminaux de la grammaire comme les opcodes engendré par la fonction. Notons que les processus de mutation font souvent appel à des constantes engendrées de façon aléatoire. Dans notre contexte, nous assimilons toutes les nouvelles constantes à un terminal unique (ce qui implique une approximation).
- Une fois les règles produites, la signature est donnée par la grammaire. Le système calcule alors un automate à pile pour la reconnaissance du langage du virus.

La détection de virus est un problème largement indécidable. Pour autant, au vu de l'enjeu extrêmement important, le jeu du chat et de la souris entre les développeurs de virus et les développeurs de protections est très actif. Nous proposons un nouveau type de signature avec le potentiel apporté par le gain de précision séparant les langages hors-contextes des langages rationnels.

La conception d'un virus enrayant notre méthode est possible en développant un système de mutation contexte-dépendant. Pour autant, une large proportion des virus apparaissant chaque jour implique des développeurs de niveau moyen, utilisant des systèmes de mutation existants. Pour ceux-là, notre méthode peut s'avérer efficace et régler un nombre de cas significatif.

# Chapitre 3

# Robotique Humanoïde

## Robotique Humanoïde

La robotique humanoïde se caractérise par une morphologie similaire ou partiellement similaire à celle de l'homme. Un robot humanoïde est généralement doté d'un grand nombre [1] de degrés de liberté, lui conférant un champ d'action très étendu, à l'instar de l'homme.

La conception de robots humanoïdes s'inscrit dans la continuité d'un rêve ancestral qui fascine l'homme depuis l'apparition des premiers automates [2], ceux de Vaucanson au XVIIIe siècle, voire ceux de Léonard de Vinci au XVIe siècle (voir [32, 43] ). Les robots humanoïdes en tant que tels sont apparus il y a une quarantaine d'années, probablement avec le WABOT-1 en 1973. De nombreux projets ont vu le jour depuis. On peut mentionner les plus connus, et en premier lieu Asimo [3] de la société Honda (voir [44, 45]), ou encore HUBO 2/KHR-4 ([21]), capables de courir, le robot Lola ([115]), ou le robot européen ICub ([100]). Mentionnons également Sarcos ([108]) et PETMAN à technologie hydraulique, ou Justin ([15]) contrôlés en force, l'androïde HRP-4C ([56]), dernier né de la série HRP. Citons le projet ROMEO porté par la société Aldébaran. Citons également les robots semi-passif intégrant des articulations libres (voir e.g. [4]).

Il convient également de mentionner des robots comme le Manoï de la société Kyosho, ou encore les robots de la série KHR de la société Kondo, ces robots et quelques autres encore sont apparus dans un contexte différent, non nécessairement académique ou industriel, mais plutôt sur la base de la communauté du modélisme et grâce à la démocratisation des technologies embarquées [4]. S'ils n'intègrent pas la même complexité que les robots mentionnés ci-dessus, loin s'en faut, ils montrent des perfomances motrices très intéressantes.

Mentionnons le robot Nao ([37]) qui suit une voie médiane, il s'agit probablement du premier projet à inscrire la robotique humanoïde dans une perspective réellement industrielle pour un développement grand public, joignant robotique humanoïde et robotique personnelle. Du même gabarit, il convient de citer QRIO de la société Sony, et le Darwin OP, ce dernier étant focalisé sur la RoboCup.

1. bien qu'incomparable à celui de l'homme comportant plusieurs centaines de muscles et des dizaines de degré de liberté

2. Bien sûr, l'idée de concevoir un être humanoïde est bien antérieure si l'on pense au mythe du Golem.

3. "Advanced Step in Innovative MObility"

4. Peut-être à rapprocher des FabLab (FABrication LABoratory) qui incarnent un développement notable des capacités de prototypage d'individus non-nécessairement inclus dans une institution.

En effet, les robots humanoïdes s'inscrivent naturellement dans la perspective de la robotique personnelle : leurs possibilités d'action, très diverses, ajoutées à la locomotion, permettent d'envisager un grand nombre de tâches assumées par la même plateforme, tout comme l'homme. Ce caractère multitâche est à souligner d'autant plus qu'il implique un rapport plus favorable entre le coût et le service rendu. Leur morphologie les rend plus adaptés aux infrastructures conçues pour l'homme et par conséquent leur déploiement ne nécessite pas d'adaptation de l'environnement. Enfin l'interaction homme-robot peut être envisagée de façon plus naturelle.

Un certain nombre de travaux s'orientent vers l'utilisation de robots humanoïdes pour les relations publiques, pour des applications d'accueil ou de divertissement ("Entertainment"). Ces robots, des androïdes dont la morphologie mais également l'apparence sont extrêmement proches de l'humain, incarnent avant même d'être utiles toute une mythologie issue de la science-fiction et exercent de fait une fascination très forte, positive comme négative, mais dont l'intensité nous conduit à la rapprocher de la fascination plus ancienne évoquée plus haut. L'orientation de la série des HRP avec le HRP-4C ([56, 55]) en est une belle illustration. Il faut citer également les Geminoid d'H. Ishiguro dont le plus célèbre est une réplique exacte de lui-même (voir e.g. Geminoid F [50]).

## Locomotion

La locomotion est un point essentiel de la robotique. Elle est un prérequis à de nombreuses tâches, notamment dans le cadre de la robotique personnelle. Elle reste un thème majeur de la robotique moderne.

Les roues ou les chenilles offrent des solutions éprouvées dans le cas d'un terrain plat, un appartement par exemple, ou même accidenté. Elles rendent possible la conception de plateformes d'ores et déjà opérationnelles dans le quotidien, par exemple pour les tâches d'entretien. Ces solutions deviennent moins adaptées en présence d'obstacles, comme sur un site de catastrophe par exemple, ou simplement dans un cadre domestique (marches d'escalier, petit mobilier). Les robots munis de jambes ou de pattes, bipèdes, quadrupèdes, ou hexapodes peuvent alors être envisagés, d'autant plus que, comme évoqué précédemment, ce type de morphologie est davantage adapté aux infrastructures de l'environnement humain. Cela soulève des problèmes fondamentaux de robustesse et d'équilibre. D'importants progrès ont été accomplis durant les trois dernières décennies (voire e.g. [118, 87, 45, 54, 29]). Cependant le problème de la bipédie n'est pas encore résolu dans son intégralité ; aucun robot bipède n'a jusqu'à présent atteint la robustesse, l'agilité et la fluidité de l'être humain.

L'étude de ce problème fascinant est très active aujourd'hui et suscite l'exploration de nombreuses pistes nouvelles, que ce soit dans le contrôle et la planification, techniques basées sur le "Zero Moment Point" (ZMP) notamment (voir [54, 86]), ou la méthode "virtual leg" ([95]), ou encore dans la mécatronique avec le contrôle en force ([108]), citons [105], mais également dans le domaine de l'apprentissage, ou encore de la morphologie. L'exploration de nouvelles structures basées sur des matériaux souples et/ou flexibles est également très prometteuse. Un exemple intéressant est le robot RHEX ([102]). Ce robot n'emploie pas de roue ni de pattes. Son déplacement repose sur des membres flexibles accumulant et restituant l'énergie, et absorbant les perturbations. Une idée importante de ce type de conception est de laisser une partie du contrôle à la structure mécanique elle-même, qui par sa flexibilité et sa souplesse s'adapte directement à l'environnement. L'inspiration est biologique (les insectes). Il en résulte un ro-

bot extrêmement robuste dans ses déplacements. Mais dans le même temps, cela complexifie les algorithmes de contrôle qui doivent prendre en compte la déformation de la structure elle-même, déformation contrôlée de façon indirecte. Cette idée de déformation non contrôlée est poussée encore plus avant par le concept de robots passifs, mécanismes sans motorisation, où les mouvements sont engendrés par la gravité uniquement. Malgré leur simplicité apparente, leur conception produit des démarches saisissantes de réalisme ([24]) dans leur similarité avec la démarche humaine.

## Rhoban, Acroban, Sigmaban, ...

Notre activité essentielle est la conception et la réalisation de plateformes robotiques, et en particulier de robots humanoïdes. Nous intervenons sur un spectre large de la chaîne technologique menant à la plateforme :

- Nous concevons et réalisons nous-même *la structure mécanique* de nos robots. Nous avons pour ce faire mis en place plusieurs outils de prototypage, parmi lesquels en collaboration avec L. Gondry, expert indépendant en électro-mécanique, une machine d'usinage à commande numérique ("Commande numérique par calculateur", CNC), ou encore depuis peu une imprimante 3D. Cela nous permet d'avoir beaucoup de liberté dans la conception et de racourcir de façon très importante le cycle de conception/réalisation.
  Nous utilisons également des techniques de prototypage rapide. En particulier, nous utilisons des techniques à base d'un matériau plastique modelable (le polymorphe) à la suite d'échanges avec R. Knight ([73]). Par exemple, la main à 12 degrés de liberté des bras anthropomorphe (voir http ://www.youtube.com/watch ?v=F9W4NyD5XsI).
- Au niveau mécatronique, nous utilisons principalement des servo-moteurs standard en les modifiant. Dans Rhoban, nous avons modifié leur intégration, pour atteindre une structure de très petite taille (27 cm). Dans la version actuelle de Sigmaban, nous sommes en cours de remplacement du système embarqué de contrôle pour avoir une plus grande maîtrise de la boucle d'asservissement bas-niveau.
  En revanche, essentiellement du au gabarit de nos robots, nous n'utilisons pas de réducteurs sophistiqués comme des réducteurs harmoniques par exemple, utilisé dans beaucoup de plateformes (comme Asimo). Cependant, pour des raisons historiques mais également choisies, nous utilisons des composants à bas coût. En effet, un frein réel au déploiement de la robotique personnelle est le coût des composants qui interdit un déploiement grand public.
- Nous concevons l'ensemble de l'architecture embarquée à bord du robot. Il est à noter que nos robots sont autonomes. La motivation est la participation à la RoboCup. Ainsi, nous déployons une architecture basée autour d'une carte de contrôle de haut niveau comparable à un ordinateur standard, et d'une carte électronique de contrôle bas niveau que nous concevons nous-même architecturée autour d'un microcontrôleur de type ARM9. Cette carte est en charge de l'acquisition des données capteurs ainsi que du protocole de communication avec l'appareil moteur. Nous y développons l'ensemble du logiciel, pour assurer en particulier de bonne propriétés "temps réel".
- Enfin, nous développons l'ensemble du logiciel de contrôle. A cela s'ajoute une suite logicielle que nous avons également développé, destiné au développement et à l'expérimentation de primitives motrices, et au monitoring des plateformes en temps réel.

Cela implique un coût très significatif en terme d'ingénierie. Cependant, le bénéfice est également important. Cela nous permet d'intervenir rapidement à tous les niveaux de la plateforme.

Les principales plateformes [5] que nous avons conçues sont les suivantes :

- Rhoban [Vidéo] : `http://www.youtube.com/watch?v=7_0YE2kOleY`
- Acroban [Vidéo] : `http://www.youtube.com/watch?v=wQ9xd4sqVx0`
- Sigmaban [Vidéo] : `http://www.youtube.com/watch?v=y8S0wQvJvXc`
  et [Vidéo] : `http://www.youtube.com/watch?v=H5OYCXZD-FI`
- Les bras anthropomorphes, et notamment la main à 12 degrés de libertés. [Vidéo] : `http://www.youtube.com/watch?v=F9W4NyD5XsI`
  et [Vidéo] : `http://www.youtube.com/watch?v=g7vo01mBixc`

## Compliance et locomotion

### Compliance

On considère la *compliance* (ou *souplesse*) comme la capacité du système à absorber et accompagner les perturbations physiques extérieures. C'est l'inverse de la raideur. Le système peut être déformé sous l'action de forces extérieures. On entend par déformation la variation de position des ses articulations, mais pas seulement : nous utilisons également des matériaux flexibles ou déformables. Le contrôle de la compliance se traduit par les forces que le système met en jeu pour s'opposer/résister à ces déformations.

Nous explorons l'usage de la compliance dans le contexte de la robotique humanoïde, et en particulier pour le problème de la locomotion. Nous déclinons cela à deux niveaux : mécatronique/contrôle et mécanique.

Au niveau mécatronique, nous utilisons principalement des servo-moteurs contrôlés en position, mais dont les paramètres sont modifiables en temps réel. En particulier les paramètres de l'asservissement en position, mais également une borne maximum sur la puissance électrique consommée par le moteur. Cela permet d'avoir un contrôle en temps réel sur la compliance de l'articulation (voir Section 3.3). Par ailleurs, nous avons aujourd'hui la maîtrise du logiciel embarqué dans les servo-moteurs (sur une mécanique existante) et nous sommes en cours de déploiement de notre propre système de contrôle servo.

Cependant, nous n'employons pas de contrôle en force. En effet, il n'y a pas d'asservissement de la force exercée par l'articulation ; les réactions du système aux perturbations ne sont pas le fruit de calcul et de décisions du système de contrôle. Le robot n'est donc pas contrôlé en force. En revanche, il peut être qualifié de *semi-passif*. Un robot est usuellement qualifié de *passif* s'il ne comporte aucun moteur et la physique est son seul contrôleur, et que sa seule source d'énergie est l'énergie potentielle. Ce type de robot a été introduit dans [76], mentionnons également [90]. Dans notre cas, certaines articulations sont contrôlés par des amortisseurs et en ce sens sont passives. C'est pourquoi nous qualifions le système de semi-passif (voir Section 3.1.2).

### Contrôle

En premier lieu, le système de stabilisation d'Acroban et de Sigmaban implémente un contrôleur agissant sur les genoux, les hanches et le bassin pour maintenir l'équilibre. Ce

5. disponible à l'adresse suivante : www.rhoban-project.org

contrôleur accepte ses entrées d'un accéléromètre, d'un gyromètre, les erreurs positionnelles de chaque articulation, la position des liaisons linéaires passives, ainsi que la consommation des moteurs des chevilles. (voir Section 3.3.1).

Le système de stabilisation ne s'appuie pas seulement sur cela. Il repose également sur le caractère semi-passif du robot :

En premier lieu, la compliance de la partie haute du corps, et en particulier de la colonne vertébrale, agit comme un système couplant un pendule inversé (la colonne vertébrale) et deux pendules passifs (les bras). Le mouvement de ce système essentiellement passif est maintenu par les articulations actives : les jambes et le bassin. Durant la marche, ce système contribue également au transfert latéral de poids. Les expérimentations ont montré que ce processus diminue la consommation de courant ainsi que l'effet du choc inélastique à l'impact du pied. En second lieu, bien qu'à un degré moindre, les chevilles sont également compliantes, cela diminue également l'effet du choc décrit précédemment. Enfin, le matériau utilisé pour les pieds est lui même souple, il est également partiellement glissant (sur les parties externes du pied). Cela permet un ajustement naturel de la position du pied sur le sol dans le temps séparant l'impact en tant que tel et la stabilisation et la pleine adhérence du pied au sol.

**Marche Semi-Passive**

La locomotion est alors considéré comme une perturbation auto-engendrée par des primitives motrices périodiques, basées sur des sinusoïdes [6] actifs dans les jambes et le bassin.

Ainsi, la marche d'Acroban ne repose pas sur le calcul en temps-réel de la dynamique du robot. Elle ne repose pas non plus sur les techniques classiques basées sur le ZMP. La marche d'Acroban peut être qualifiée de semi-passive, inspirée par les mécanismes passifs auto-stabilisant introduit par [76], ou les robots semi-passifs motorisés (voir [25]). Le mécanisme d'équilibrage semi-passif d'Acroban peut être rapproché du marcheur passif motorisé de [110].

L'utilisation de la compliance, voire de la passivité, à la base du mécanisme de stabilisation transforme l'ensemble du système mécanique en un *système distribué* recherchant naturellement les minima locaux d'énergie potentielle. Il absorbe les chocs (en particulier les chocs inattendus) et ajuste constamment la position du robot sans avoir recours à un contrôle global centralisé à haute fréquence.

Notons que le caractère absorbant/amorti renforce la stabilité du système, nous l'avons observé expérimentalement à la lumière des indicateurs qualitatifs décris en Section 3.5. En particulier il atténue, au prix d'une déperdition d'énergie bien sûr, les phénomènes chaotiques pouvant apparaitre, y compris naturellement dans un système purement passif comme le classique double pendule.

Le système de contrôle global est d'autant soulagé, en particulier sur les réactions à haute-fréquence (essentielles en ce qui concerne les chocs). En ce sens, nous suivons les idées du concept de "calcul morphologique" (see e.g. [92], [94]) en laissant la structure mécanique assumer une large part du contrôle. La nature distribuée de cette méthode de contrôle peut se comprendre comme un système modulaire formé d'un mélange de réactions mécaniques indépendantes (mouvements, système de stabilisation, et mécanique elle-même). Cela étend l'architecture robotique classique intégrant un ensemble d'articulations, chacun étant actionné par un

6. inspirées de "central pattern generators"

servo-moteur indépendant lui-même contrôlé en position.

L'utilisation de la compliance n'est pas prépondérante dans le contexte de la locomotion des robots à patte. Cependant, plusieurs projets ont montré son intérêt. Cela peut être illustré par le robot Bigdog basé sur un système moteur hydraulique ([95]), voir également [51]). Certains robots bio-inspirés ont également recours à la compliance, par exemple le robot ECCE ([73, 74]), et les robots du projet PHRIENDS ([2, 16], [46]). L'usage de matériaux flexibles a également été explorée, notamment par le robot RHEX ([101], citons également [35], [103] [30], [78]). Enfin, la flexibilité/compliance matérielle est essentielle dans la conception de prothèses (voir *e.g.* [117]).

Dans tous ces exemples, la robustesse du déplacement est améliorée. En revanche, le prix à payer est souvent un manque de prédictabilité du système, voire de précision.

## Méthode de développement de contrôleurs

Beaucoup de contrôleurs de robots (citons les plus connus Asimo, ou HRP) sont basés sur une modélisation mécanique, cinématique et dynamique très précise. Cette modélisation est allié à une réalisation de la plateforme extrêmement précise et conforme au modèle. Cela permet de développer des primitives motrices sur la base de simulation, et de planification. Le succès de ces méthodes est très grand, cela a permis à Asimo à courir par exemple.

En revanche, le coût de l'adéquation de la plateforme au modèle est important (mécanique de précision, motorisation de grande qualité). Une préoccupation fondatrice de notre activité est la baisse des coût de réalisation de plateforme, notamment en vue du développement de la robotique personnelle. Un déploiement grand public entrainant à coût sûr des éconnomies d'échelle importante. Pour autant, ces dernières seront-elles suffisantes ? Nous explorons également une méthode relativement différente, fortement basée sur l'expérimentation.

Pour autant, nous avons implémenté plusieurs modèles très simplifiés. Nous avons étudié en collaboration avec D. Cavaille, étudiant en master 2 EEA, un modèle d'équilibre d'un pendule inversé à 5 segments (tibia, cuisse, bassin, colonne vertébrale et la tête), plus un segment horizontal symbolisant le pied, simplification du robot dans le plan sagittal. Ce modèle nous a permis d'expérimenter le comportement de contrôleurs simples, sous l'action de diverses perturbations. En revanche, la divergence entre ce modèle simplifié et le robot est importante, dûe au jeu, à la flexibilité de la structure notamment. Cette divergence rend impossible la transposition d'un contrôleur du modèle vers le robot.

Par ailleurs, nous avons développé un modèle de marcheur passif simplifié, intégrant 2 jambes sans genoux et une masse au niveau du bassin, avec également un modèle du pied sous la forme d'une courbe paramétrée. Ce modèle est en cours de développement, nous souhaitons à terme montrer l'influence de la forme des pieds sur la stabilité du cycle de marche (voir Section 3.1.2, voir également Figure 3.1).

M. Lapeyre a également développé une modélisation 3D d'Acroban permettant de le tester dans un simulateur physique dans le but d'expérimenter et de mettre au point des processus d'apprentissage.

Le développement de ces modèles nous a permis d'explorer la problématique de la marche, mais ne converge pas encore vers la réalité de la plateforme.

La nature du contrôleur d'Acroban ou de Sigmaban n'est pas issue d'un processus de synthèse numérique ou de planification s'appuyant sur un modèle. Elle est issue d'un processus

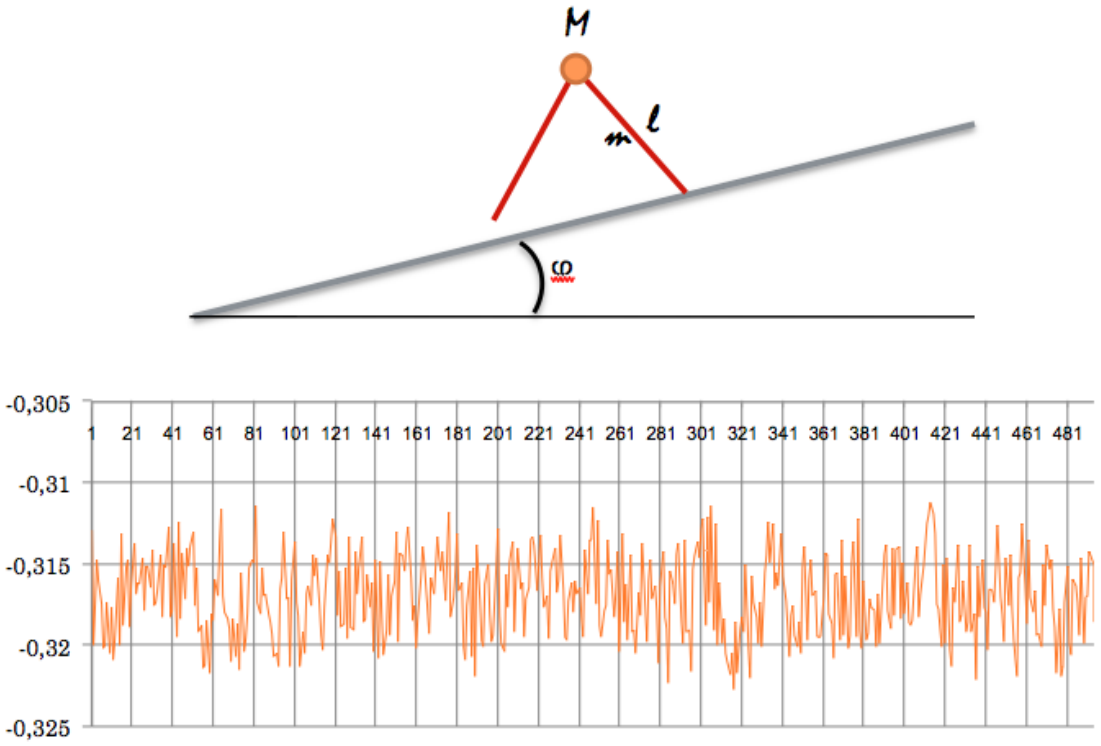


FIGURE 3.1 – Marcheur passif simplifié. Nous avons représenté ici l'angle des jambes (en radian), discrétisé à chaque impact de pas, pour un marcheur passif théorique réduit à sa plus simple expression : une masse ponctuelle au niveau du bassin, pas de genoux, pieds ponctuels se déplaçant sur une pente constante et parfaite. (La simulation est réalisé au moyen des équations d'Euler-Lagrange du système.)

itératif guidé par l'expérimentation. Typiquement l'architecture de contrôle de la primitive motrice d'équilibrage (voir Section 3.3) est construite de façon modulaire par la conjonction d'un ensemble de contrôleurs simples (souvent à base de PID) expérimentés et réglés directement sur la plateforme, selon des critères d'évaluation précis, les indicateurs de la marche (voir Section 3.5). Ce faisant nous tirons parti :

- du gabarit de nos robots qui facilite considérablement l'expérimentation. A l'instar de Nao, ou de l'Aibo, le robot peut être mis en oeuvre au moyen d'une infrastructure très légère, ne nécessitant qu'un seul opérateur et un espace de travail limité (une table).
- d'une maîtrise relativement étendue de la chaîne technologique qui permet d'intervenir rapidement au niveau logiciel, au niveau du système embarqué, au niveau mécatronique et enfin au niveau mécanique. Typiquement, l'ajustement de paramètres du contrôleur est mené en parallèle avec l'ajustement de la structure mécanique, par exemple de la nature des pieds, ou de la taille de tel ou tel segment.

Pour la marche, qui concentre l'essentiel de notre travail, l'idée sous-jacente à notre méthode est

de considérer la plateforme elle-même comme un système dynamique qui au lieu d'être évalué par simulation l'est directement par l'expérience. De façon plus précise, on considère le cycle de marche comme un système dynamique discret, faisant passer le robot d'un état de double appui à un nouvel état de double appui via un cycle de deux pas. Nous abordons l'etude de la marche comme l'étude du système dynamique sous-jacent dont nous avons une connaissance partielle de l'état au travers de l'ensemble des valeurs de capteurs ainsi que des positions et des vitesses des articulations (voir Section 3.5.1).

## 3.1 Morphologie, Structure

**Gabarit.** Acroban et Sigmaban ont une morphologie globale humanoïde (voir Figure 3.2 [7] et Videos) intégrant les articulations classiques des robots humanoïdes, une trentaine de degrés de liberté [8]. Ce sont des robots de petites tailles (entre 60cm et 70cm), comparables à celles de *NAO*, de *Qrio*, ou encore de *Darwin OP* ([84]). C'est un point à souligner à plusieurs titres. Cela contribue, avec probablement de multiple facteurs (incluant les matériaux utilisés, la nature des réducteurs de moteurs, la nature compliante des contrôleurs) au fait que le robot est pourvu d'une résistance mécanique importante, autorisant les chutes par exemple. Il est a souligner que la robustesse est un facteur déterminant si l'on pense à un déploiement grand public, cette question pourrait faire l'objet d'études plus approfondies dans le futur. Ce format facilite considérablement l'expérimentation. Il autorise notamment la mise en œuvre d'expérience d'apprentissage de durées significatives, sans avoir à recourir à une infrastructure lourde.

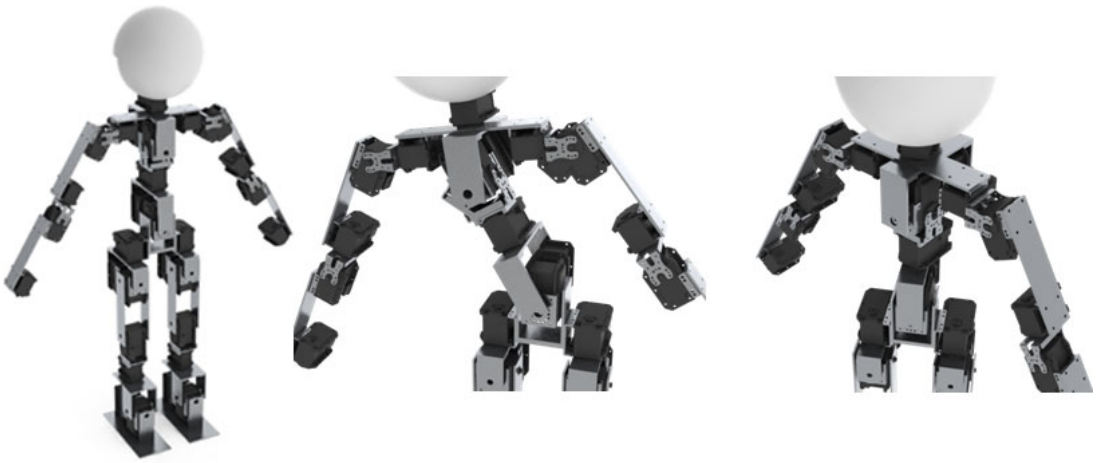

FIGURE 3.2 – Morphologie Globale

### 3.1.1 Tronc multi-articulé bio-inspirée

Acroban intègre *une colonne vertébrale multi-articulée* doté de cinq degrés de liberté (voir Figure 3.2). Cela permet de ne plus envisager le tronc comme un solide indéformable, mais

7. L'auteur remercie M. Lapeyre pour la réalisation du modèle 3D d'Acroban.
8. Plusieurs versions co-existent avec un cou comportant de zero à trois articulations.

comme un système à part entière reliant le bassin et les épaules. Cela ouvre de nouvelles perspectives en terme de mouvements. Cette colonne vertébrale est couplée à un *bassin multi-articulé*. La zone du bassin (intégrant le bas de la colonne vertébrale et les hanches) comporte ainsi 8 degrés de liberté (incluant les 2 articulations inférieures de la colonne vertébrale).

Relativement peu de robots intègrent des articulations dans le tronc, à des degrés divers. La plupart de ces robots utilisent cela pour élargir leurs espaces de travail (mentionnons l'iCub ([79]), ou encore le robot ECCE ([74]) bio-inspiré), mais peu d'entre eux en tirent profit pour la marche et l'équilibrage (citons le Wabian [14], pouvu d'un bassin multi-articulé), ou plus généralement pour la locomotion (citons la salamandre d'Ijspeert ([47])). Mentionnons Romeo, en cours de développement, pour lequel l'intégration de vertèbres a été étudiée ([106]). Mentionnons également le WBD-1 ([91]) qui explore les possibilités d'articulation du tronc pour la danse.

**Bio-inspiration.** La locomotion des vertébrés, et en particulier la marche chez l'homme constitue encore aujourd'hui un mécanisme inégalé par des machines, et un challenge très ouvert. Elle a fait l'objet de recherches importantes dans le domaine de la bio-mécanique (voir *e.g* [98] ou encore [116]). Dans ce corpus cependant, peu de travaux se sont focalisés sur le rôle du tronc dans la marche. Notons pourtant que le tronc constitue 60% de la masse totale chez l'homme, il est le lieu du centre de gravité du corps (voir *e.g.* [113, 33]). Il comporte un réseau complexe de muscles, utilisés intensivement durant la marche et en particulier pour le maintien de l'équilibre, comprenant mouvements d'anticipation et mouvements de réaction. La colonne vertébrale y est essentielle, en particulier en raison de son rôle dans le transfert latéral du poids durant la marche.

Il n'est pas clair que le mécanisme de marche de l'humain soit directement transposable dans le contexte de la robotique. Bien que sa morphologie globale soit similaire à l'homme, un robot humanoïde a considérablement moins d'articulations, d'éléments moteurs, ou encore de capteurs. Cependant, dans l'idée d'obtenir une marche robuste, intégrer une colonne vertébrale ou l'équivalent à un robot humanoïde semble sensé au vue des remarques précédentes.

Pour des robots de gabarit comparable à l'homme, il est envisageable de s'inspirer des vertèbres comme cela a été fait pour le ECCE robot ([74]), ou encore Romeo ([106], voir aussi le robot kenta [80, 81]).

Cependant, dans notre cadre de travail, l'intégration de vertèbres dans des robots tels que Acroban ou Sigmaban semble irréaliste à cause de leurs petits gabarits (pour des raisons d'intégration et de robustesse). Nous nous sommes plutôt attachés à simplifier et intégrer les degrés de liberté essentiels de la colonne vertébrale. Dans [18], Ceccato a étudié le rôle du tronc dans la marche humaine et les mouvements essentiels de la colonne vertébrale durant la marche. Il ressort de ses travaux que l'apparente complexité du tronc peut être réduite à peu de composantes essentielles.

D'abord, durant la marche, l'expérience a fait apparaître de petites oscillations du bassin et du thorax dans le plan sagittal. De façon plus précise, seules deux articulations semblent nécessaires pour approximer le mouvement de la colonne vertébrale dans le plan sagittal, l'un situé au niveau du bassin, l'autre au niveau du thorax.

Dans le plan frontal, le bassin et les épaules oscillent en opposition de phase alors que le centre reste immobile pendant le cycle de marche. De nouveau, seules deux articulations semblent nécessaires, l'un au niveau du bassin, l'autre au niveau des épaules.

Enfin, dans le plan horizontal, il y a deux rotations opposées du haut du tronc et du bas du tronc, soutenues par la torsion de la colonne vertébrale. Il semble donc qu'une articulation rotative au centre de la colonne vertébrale pourrait être suffisant.

**Structure mécanique du tronc.** En suivant les remarques précédentes, le tronc d'Acroban comporte 5 articulations (voir Figure 3.2) :

- 2 dans le plan sagittal respectivement au niveau du bassin et du thorax,
- 2 dans le plan frontal respectivement au niveau du bassin et du thorax,
- 1 dans le plan horizontal assurant le mouvement de torsion.

Cette conception produit une indépendance forte entre la partie haute du corps (épaules, tête et bras) et la partie basse (bassin et jambes). Cela permet par exemple aux jambes et au bassin de bouger en minimisant les déplacements de masse dans la partie haute du corps. Cela est illustré en particulier par la marche où la colonne vertébrale est utilisée pour le transfert latéral de masse, soulageant ce faisant les hanches et les chevilles.

Notons qu'en comparaison avec un corps ne comportant seulement que des jambes, la partie supérieure du corps (torse, bras, tête) élève le centre de gravité. Est accrue d'autant l'amplitude du mouvement du centre de masse dans le plan horizontal, ce qui facilite le transfert latéral du poids d'une jambe sur l'autre. La fréquence naturelle de la marche est également abaissée, ce qui soulage le système embarqué. De façon plus spéculative, remarquons que l'homme, lorsqu'il porte une charge, a tendance à la porter haut. Comme un sac à dos, voire les charges transportées sur le haut du crâne. Cela permet bien entendu de transposer l'effort des muscles vers le squelette, profitant de sa rigidité, mais le contrôle humain de la marche y est adapté. En revanche, il est clair que le système devient plus instable et plus sensible au déséquilibre.

### 3.1.2 Flexibilité, compliance et souplesse de la structure

**Flexibilité.**

Nous utilisons des matériaux flexibles à plusieurs niveaux. Nous utilisons ressorts et élastiques : 1) pour fournir du couple (vu comme accumulateurs d'énergie), et ainsi soulager la motorisation ; 2) pour atténuer les comportements non-linéaires occurrant du fait du jeu dans les réducteurs. Nous utilisons ainsi des ressorts de torsion dans les genoux, les chevilles et les hanches pour le plan frontal. Notons que nous utilisons des réducteurs standards qui comportent un jeu initial non négligeable (voir partie 3.2.1). Au vu du gabarit de nos robots, nous ne pouvons pas utiliser de réducteur de type harmonique. Cependant, l'usage de tels réducteurs est difficilement adaptable dans notre contexte car il ne sont pas réversibles, et difficile à rendre compliant.

D'autre part, nous utilisons également des matériaux flexibles ou déformables pour la structure du robot. De façon empirique, nous avons testé plusieurs types de plastiques pour la structure des jambes de Sigmaban (voir Vidéo http ://www.youtube.com/watch ?v=H5OYCXZD-FI).

La flexibilité de la structure, en particulier des jambes est significative. Dans le plan sagittal elle provient essentiellement des materiaux utilisés (plastique et aluminium). Quantitativement, la flexibilité du corps d'Acroban dans le plan sagittal, tout moteur raide, atteint 20°.

**Amortisseurs.**

L'un des buts de la conception de Sigmaban fut d'expérimenter l'usage d'amortisseurs. L'objectif était d'amortir les chocs occurrant lors de la marche, en particulier lors de l'impact du pied au sol. La méthode que nous avons suivie consiste à intégrer de nouvelles liaisons linéaires, au moyen de glissières, couplées à des amortisseurs de type hydrauliques telescopiques (typiquement utilisée dans les voitures). Nous avons expérimenté plusieurs conceptions :

- liaisons verticales dans les hanches (voir Figure 4.1. Il s'agit d'absorber le choc vertical occurrant à l'impact du pied.
- Une liaison verticale dans la colonne vertébrale. Il s'agit comme précédemment d'absorber le choc vertical occurrant à l'impact du pied.
- Une liaison verticale dans chaque épaule. Il s'agit comme précédemment d'absorber le choc vertical occurrant à l'impact du pied. Le choc du pied touche également le bras, dont le couple peut être important si le bras n'est pas disposé le long du corps.
- Une liaison horizontale dans le plan sagittal. Il s'agit là d'amortir le choc horizontal.

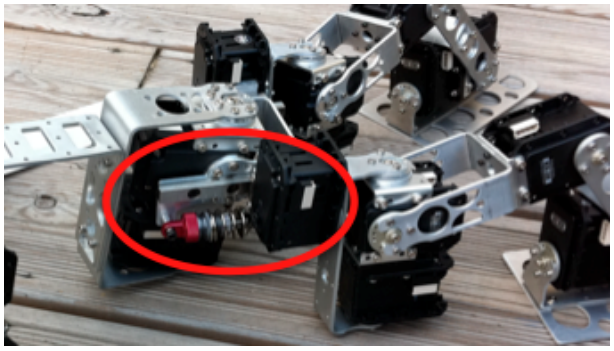

FIGURE 3.3 – liaison amortie dans la hanche

La vidéo suivante illutre l'usage de l'ensemble de ces liaisons : Vidéo http ://www.youtube.com/watch ?v FI.

A l'issue d'une série d'expérimentations, la version actuelle de Sigmaban n'intégre que les liaisons verticales linéraires amorties dans les hanches. Nous n'avons pas intégré les autres liaisons essentiellement pour des raisons de robustesse mécanique. Le résultat de la marche peut être observé dans la vidéo suivante video http ://www.youtube.com/watch ?v=y8S0wQvJvXc.

Notons que l'anatomie des vertébrés dispose également d'un mécanisme d'amortissement : le cartilage, omniprésent dans toutes les articulations. Dans un autre ordre d'idées, la plupart des véhicules, en particulier ceux destinés à un environnement accidenté, sont équipés d'amortisseurs. Outre la stabilité, cela améliore la robustesse de la structure elle-même qui subit moins de choc.

**Le pied.**

**Matériau.** Enfin, toujours dans l'objectif d'amortir les chocs de la marche, nous avons travaillé sur le matériau utilisé pour les semelles du robot, directement concernées par l'impact du pied au sol. Dans ce sens, nous avons expérimenté divers types de caoutchouc, et de silicone.

Nous utilisons dans la version actuelle de Sigmaban un silicone obtenu à partir d'un dosage déterminé de façon empirique [9]. Notons que l'étude de la texture des semelles de Sigmaban nous a conduit à utiliser plusieurs types de silicones parmi lesquels le RTV EC00 utilisé pour le moulage de précision, ou encore le platsil gel utilisé dans le monde des effets spéciaux notamment pour imiter la chair. Les paramètres de réglage que nous utilisons sont le dosage du catalyseur avec la base ainsi que l'ajout d'une proportion variable d'huile silicone. Cette technique est à rapprocher de l'utilisation classique de silentbloc comme par exemple pour les chevilles de HRP. Comme évoqué précédemment, l'utilisation de silicone permet de faire varier la souplesse du matériau.

Mentionnons également le caractère partiellement glissant des semelles. A l'impact du pied au sol, il réduit la composante horizontale du choc de façon significative. Grâce à cela, le pied trouve par lui-même un ajustement local de sa position. Durant la marche, la composante horizontale du choc du pied perturbe l'équilibre postural. Eu égard à cela, ce trait améliore la stabilité du robot. En revanche, la conception du pied est issu d'un compromis car l'adhérence du pied est nécessaire à la marche également. Ainsi, nous ajoutons à la semelle un revètement glissant sur l'extérieur seulement (plastique et/ou teflon), préservant l'adhérence au centre du pied.

**Morphologie.** En parallèle à l'étude du matériau utilisé pour le pied, nous conduisons des travaux sur la forme du pied en collaboration avec Q. Rouxel, étudiant à l'Enseirb-Matmeca. L'observation de la conception de prothèses pour le pied montre que la forme est primordiale.

Nous nous proposons d'évaluer de façon théorique les rapports entre la stabilité de la marche et la *forme* du pied. Dans [49], une étude physique a été menée pour déterminer les paramètres d'un marcheur passif stable, en prenant en compte dans le modèle une forme circulaire de pied. Le modèle, soumis à l'équation d'Euler-Lagrange, décrit le marcheur comme un système dynamique dont certaines caractéristiques de stabilité (le jacobien) peuvent être évaluées par la simulation. Nous proposons d'étendre cette méthode par un modèle prenant en compte une forme arrondie, mais *non nécessairement circulaire du pied*, i.e., à courbure variable. La courbe suivie par la forme du pied devient un nouveau paramètre du marcheur, que nous pensons significatif dans la stabilité du processus de marche. Ce paramètre peut être discrétisé en utilisant des familles de courbes comme des polynômes par exemple. Cette étude est en cours. Notons cependant que dans le modèle que nous considérons, nous ne prenons pas en compte la nature flexible ou déformable du pied.

## 3.2 Contrôle

### 3.2.1 Mécatronique

Nous utilisons principalement des servo-moteurs standards offrant des articulations rotatives : Dynamixel [10] RX-64 et RX-28. Le RX-64 (resp. RX-28) peut déployer 64kg.cm$^{-1}$ (resp. 28kg.cm$^{-1}$) de couple statique, valeur baissant à 30kg.cm$^{-1}$ (15kg.cm$^{-1}$) pendant un mouvement. Le jeu n'est pas négligeable et la précision est relativement faible, ce qui est dû au

9. Mais dont l'impact sur la marche est néanmoins évalué au moyen d'indicateurs quantitatifs précis (voir 3.5)
10. Dynamixel RX64 User's Manual : www.robotis.com © Dynamixel

réducteur classique (ni harmonique, ni planétaire). Cependant, le réducteur est réversible ce qui est un point crucial pour la compliance.

Les servo-moteurs sont contrôlés en position. Cependant, ils permettent un réglage en temps réel de la puissance électrique maximum consommée par le moteur, mesurée par la consommation électrique instantanée. Par ailleurs, ces servo-moteurs offrent un mode "torque null" dans lequel le servo-moteur simule une articulation complètement libre, compensant les frottements, et dans une certaine mesure l'inertie du réducteur. Nous en faisons usage pour simuler une articulation passive.

Par ailleurs, en collaboration avec G. Passault et Q. Lambert, étudiants à l'Enseirb-Matmeca, nous conduisons des travaux pour remplacer le contrôle de ces servo-moteurs afin d'avoir une maîtrise plus complète de la chaîne technologique tout en bénéficiant de l'intégration mécanique. Au plan de l'ingénierie, nous remplaçons le firmware du servo-moteur, conservant l'électronique en place. Nous implémentons un contrôleur de type PID, en ouvrant le réglage de tous les paramètres à l'utilisateur. De plus, nous intégrons un modèle élémentaire des frottements du réducteur pour permettre la mise au point de modes "compliants" paramétrables.

### 3.2.2 Système de contrôle

L'ensemble de ce système de contrôle (voir Figures 3.4 et 3.5) est conçu et développé en collaboration avec H. Gimbert et G. Passault.

**Boucle de rétro-action.** Les servo-moteurs sont contrôlés par une couche haute, un système centralisé implémentant la logique des mouvements. La boucle globale de rétro-contrôle d'Acroban et de Sigmaban est de basse fréquence (50 Hz). Cependant l'exemple des organismes biologiques et en particulier des vertébrés, et la lenteur relative de l'influx nerveux (entre 1 et 100 $m.s^{-1}$) montre qu'une boucle de rétro-action à haute fréquence n'est pas une nécessité pour résoudre le problème de la locomotion, et en particulier du point de vue de la robustesse. Notons également la nature hautement distribuée du système de contrôle moteur des vertébrés [11].

Chaque servo-moteur implémente à son tour une boucle de rétro-contrôle à plus haute fréquence (1 KHz) indépendante, pour le contrôle en position de l'articulation.

**Primitives motrices.** Les mouvements d'Acroban et de Sigmaban sont découpés en différents modules appelés *primitives motrices paramétrées* : pour la locomotion par exemple, la marche d'une part, et le système d'équilibrage d'autre part. Nous concevons chacune des primitives motrices comme une combinaison de "splines", de fonctions périodiques et de contrôleurs linéaires (notamment de type PID). Ces éléments évoluent en parallèle, exécutés par la couche haute centralisée de contrôle. Notre système peut-être comparé à un système de schémas-blocs classique, comme i.e., Simulink. En revanche, notons que nous en avons la maîtrise totale, l'ayant conçu de zéro. Notons également que le système est conçu pour être embarqué [12].

Les splines sont définies point par point, avec une interpolation linéaire, ou polynômiale (définie alors par des points et des demi-tangentes). La définition d'une spline peut se faire selon

11. Même si ce dernier reste pour l'essentiel relativement mystérieux, en particulier si l'on pense aux phénomènes d'apprentissage sous-jacents.

12. La version 1 du système était développée dans un système embarqué 8-bits disposant de quelques KO de ram, sans système d'exploitation. La version 2 du système est exécuté à bord d'un système à base ARM9.

2 modes : soit par l'utilisateur via l'interface homme/machine, soit par manipulation directe du robot. Dans ce dernier cas, les points sont enregistrés en temps réel. Notons que notre système ne fournit pas de système de synthèse de mouvement à partir de plusieurs enregistrements. Pour autant, c'est une direction privilégiée pour les développement à venir.

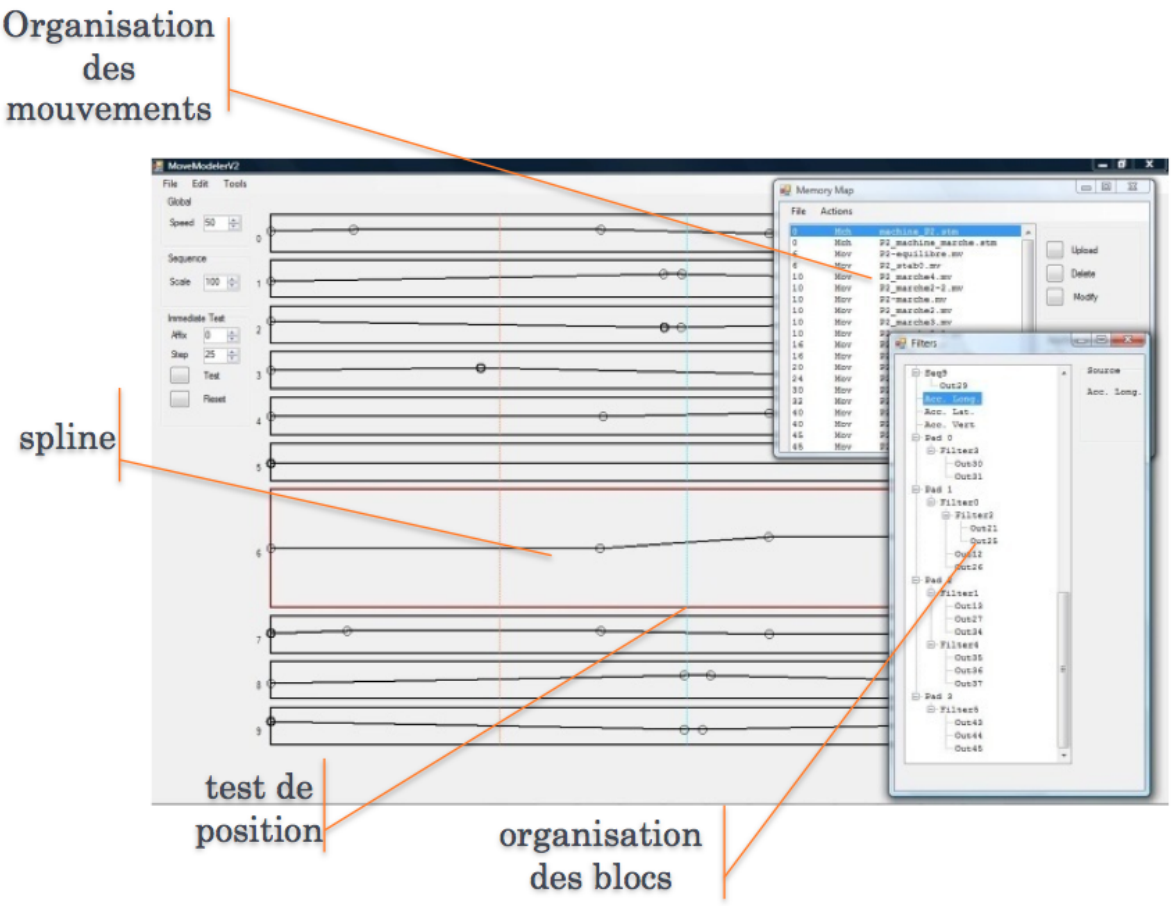


FIGURE 3.4 – Interface de conception de mouvement

Les fonctions périodiques sont implémentées par des sinusoïdes dont les périodes, phases et amplitudes sont définies par l'utilisateur.

A cela s'ajoutent des blocs linéaires, et en particulier des contrôleurs de type proportionnel-intégral-dérivée (PID) ; ainsi que des blocs de calcul élémentaires (seuils, etc).

Enfin, à chaque intervalle de temps et pour chaque sortie (voir plus loin), toutes les valeurs correspondantes dans chacun des modules sont simplement additionnées pour obtenir la sortie finale.

Les sorties sont

- l'ordre de position de chaque articulation
- la puissance électrique maximale consommée chaque articulation
- la position des pieds dans l'espace opérationnel (i.e., dans le repère cartésien associé au bassin).

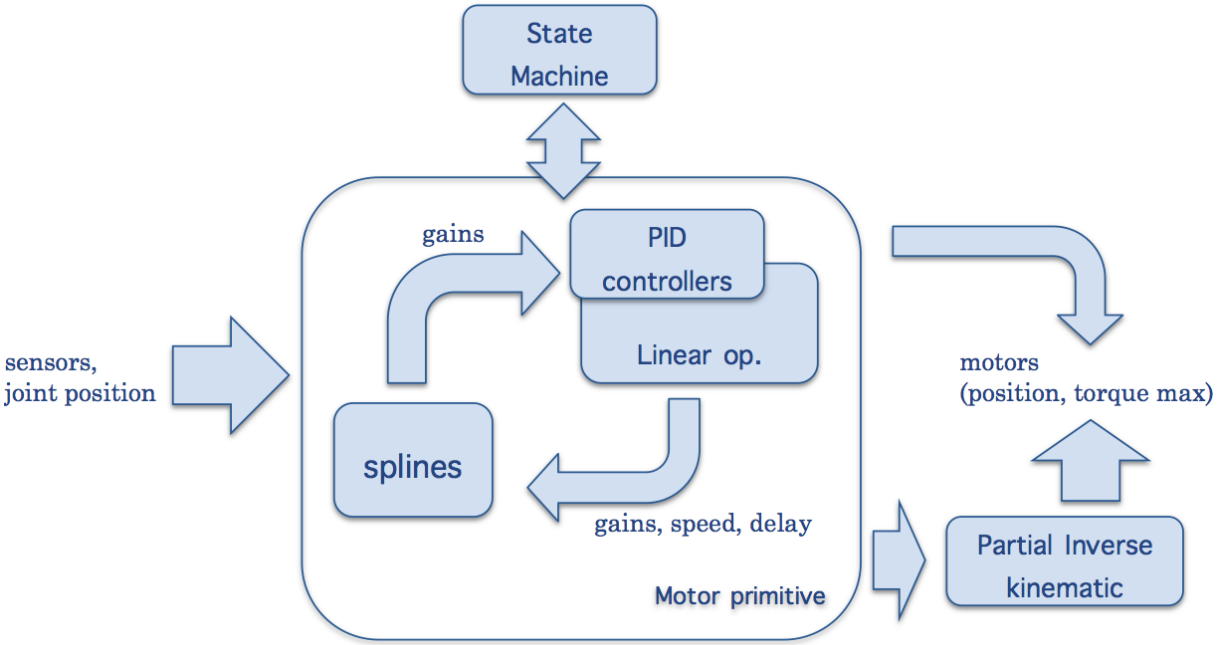


FIGURE 3.5 – Schéma global des primitives motrices. Notons en particulier l'usage d'une cinématique inverse partielle, qui permet de contrôler les pieds dans le repère cartésien lié au bassin et aux plans sagittal, frontal et transversal.

- les paramètres d'une primitive motrice peuvent également être des sorties, i.e., l'objet d'un calcul mené par la primitive motrice elle-même. Parmi ces paramètres, mentionnons le gain sur l'amplitude d'une spline ou d'un groupe de spline, les paramètres d'une sinusoïde, ou encore les paramètres d'un contrôleur PID. Par exemple, l'amplitude et la direction des pas peuvent être utilisées pour contrôler l'équilibrage.

**Moteur de comportements.** Mentionnons qu'au dessus de cela se trouve une couche de pilotage qui orchestre les primitives motrices. Ce module est basé sur l'utilisation de machines à états permettant de contrôler plusieurs primitives motrices en même temps : de les lancer, les stopper, etc.

Une illustration de ce module est son utilisation dans le cadre de l'exposition internationale de Corée 2012. Nous avons mis en scène 5 paires de bras jouant de la musique en playback. Les bras guitaristes n'ont fait qu'imiter le jeu. En revanche, le batteur était piloté par une partition de format midi. Il comprenait un certain nombre de primitives motrices correspondant aux divers mouvements d'un batteur, ainsi qu'à ses différentes positions. Chacun de ces mouvements était piloté par des événements midi.

Une autre illustration est donnée par le comportement d'interaction d'Acroban, qui déclenche plusieurs primitives motrices sous l'influence de l'utilisateur (voir section 3.5.4).

Enfin, cette couche est primordiale dans le cadre de la RoboCup. Elle orchestre les différents comportements du robot (poursuite du ballon, tir, se relever, recherche du ballon, etc).

**Vision.** Sigmaban est doté d'un appareil de vision. Le système a été développé en collaboration avec H. Gimbert, L. Hofer et P. Narbel. La vision intervient dans les indicateurs de la marche (voir 3.5.2). Elle est egalement cruciale dans le cadre de la RoboCup. En effet, elle sert à la poursuite du ballon ainsi qu'à la navigation sur le terrain. Le système de vision repose sur l'utilisation d'OpenCV pour les traitements de base des images.

Il faut mentionner que le système de vision doit fonctionner en temps réel, et embarqué à bord du système du robot (c'est à dire avec relativement peu de ressources). C'est une contrainte significative relativement aux algorithmes, mais aussi dans une moindre mesure techniquement : typiquement, nous sommes intervenus au niveau des pilotes (drivers) linux pour accélérer les traitements de base.

Nous l'utilisons classiquement pour la poursuite du ballon, en isolant par des techniques de "clipping" [13] et de reconnaissance de cercle. Néanmoins, des efforts constants sont nécessaires pour accroître la robustesse, et surtout la fréquence et le délai de traitement, sachant que la position du ballon est utilisée pour asservir le déplacement du robot (voir section 3.4.2).

Nous utilisons le système de vision également pour la navigation. En premier lieu, nous extrayons les lignes du terrain grâce à des techniques classiques (Hough). Nous extrayons également le cercle central, toujours sur la base de méthodes de Hough. Ensuite, nous avons expérimenté plusieurs techniques.

- Faire correspondre les lignes du terrain avec un modèle géométrique. L'opération est relavement ardue du fait du manque de robustesse de l'extraction des lignes. Ainsi, plusieurs lignes peuvent se retrouver confondues, certaines peuvent manquer, ou n'être obtenues que partiellement. La mise en correspondance est une opération combinatoire relativement coûteuse. Cette correspondance établie, elle permet via les paramètres de la caméra de retrouver la position de cette dernière. Nous avons expérimenté cette technique. Elle manque a priori de robustesse.
- Une seconde technique a été expérimentée en particulier par H. Gimbert et L. Hofer consistant à apprendre la fonction faisant correspondre les positions sur le terrain avec les éléments extraits de la caméra. Il s'agit de faire une base de couples (position / liste des éléments) en manipulant directement le robot sur le terrain. Puis, en temps réel, de chercher la correspondance avec les éléments extraits courant et la base de couple. Cette technique montre une robustesse satisfaisante et sera utilisée à la prochaine compétition.
- Enfin, nous avons également expérimenté une technique à base de filtre particulaire, intégrant certains éléments clé du terrain (buts, bornes latérales). A terme, nous projetons d'intégrer cette technique à la méthode précédent, en y intégrant les informations de pilotage du robot pour accroître la robustesse.

**Supervision.** Pour finir, mentionnons que notre suite logicielle intégre un système de monitoring à distance permettant en particulier de surveiller les paramètres des moteurs (taux d'erreurs, température, etc). Ce système nous permet en particulier d'anticiper les pannes. Il a été développé en collaboration avec P. Fudal, ingénieur de recherche dans l'équipe Inria Flowers.

Il faut souligner que la robustesse et la sûreté de fonctionnement sont un souci constant dans nos travaux. Cela représente un investissement non négligeable. L'exposition de Corée 2012 nous a permis de beaucoup progresser sur ce point. En effet, l'ensemble des robots (5 paires de bras à taille réelle, 2 robots humanoïdes) devait fonctionner 7j/7, 12h/j pendant 3 mois sans

13. Consistant à isoler le terrain du reste de l'environnement grâce à sa couleur unie, au marquage près.

aucun jour d'arrêt, les pannes ne devant pas impliquer plus de quelques dizaines de minutes d'arrêt partiel. La mise en place de procédures d'urgence de réparation, l'amélioration de la robustesse mécanique, électrique et électronique ont demandé beaucoup d'efforts. Cependant, l'expérience a montré qu'au final, nos robots pouvaient fonctionner plusieurs semaines en ne dénombrant qu'un nombre limité de pannes minimes (essentiellement électriques). Notons que la robustesse est également un élément crucial pour la participation à la RoboCup.

## 3.3 Equilibre postural

Nous décrivons ici l'architecture de contrôle implémentant la primitive motrice d'équilibrage, à la base de nos robots.

Pour diverses raisons évoquées précédemment, la réalisation et l'expérimentation sont relativement simplifiées dans notre contexte. Cela nous permet de donner à notre méthode un caractère expérimental très fort.

Le mouvement est issu de la superposition de plusieurs modules distincts.

### 3.3.1 Compliance

Nous avons vu que la compliance est de deux natures : contrôlée et structurelle.

**Compliance contrôlée.** Nous entendons par là l'ajustement en temps réel de la puissance électrique maximale utilisée par une articulation particulière pendant le mouvement. Nous l'utilisons de façon dynamique, pour simuler un ressort de torsion de raideur ajustable (voir *e.g.* [93]) : nous régulons la puissance maximale de façon proportionnelle à l'erreur positionnelle constatée. D'autre part, nous utilisons cet ajustement de façon statique, ce qui a pour effet de mettre la structure dans un état particulier de compliance. Par exemple, en rendant les épaules libres à certains moments du cycle de marche [14].

Le caractère passif de la partie supérieure du corps transforme le robot en un système à trois pendules couplés (la colonne vertébrale et les deux bras). L'importance de cela sera souligné plus loin (voir Section 3.4).

**Compliance dynamique.** Nous utilisons cette technique à différents niveaux du robot :

- *Chevilles.* Nous rendons les chevilles compliantes dans le plan frontal. Cela permet une meilleure adhérence du pied, plus robuste aux perturbations. On l'implémente en simulant un ressort ajustable, lui même contrôlé de façon proportionnelle-intégrale par l'erreur en position. Par ailleurs, la raideur varie également suivant l'instant dans le cycle de marche. Nous assurons cela en contrôlant la raideur au moyen d'une spline définie par un ajustement manuel point par point.
- *Bassin et colonne vertébrale.* Les articulations frontales de la colonne vertébrale et du bassin sont également rendus compliants. En particulier les articulations frontales supérieur de la colonne vertébrale (au niveau du thorax) est rendu complètement libre. L'articulation inférieure agit comme un ressort ajustable.

14. Une telle configuration du système peut-être interprétée comme produisant des propriétés de calcul morphologique particulier, ce qui peut-être rapproché du concept de morphosis (voir [23])

- *Les Epaules et les bras* sont également rendu compliants, voire libres.
- *Articulation sagittale du bassin.* Nous ajustons la résistance de l'articulation sagittale du bassin pour absorber les perturbations agissant sur l'ensemble de la structure comme rotations dans le plan sagittal.

Notons que le fait que la souplesse soit contrôlée dynamiquement par des contrôleurs linéaires ou des splines assure des transitions continues dans la raideur du robot. Nous soulignons ce fait élémentaire car c'est une des motivations pour concevoir notre propre version du contrôle bas-niveau des servo-moteur. En effet, le contrôle standard comporte des discontinuités dans l'ajustement de la raideur du servo. A terme, nous souhaitons corriger cela.

**Compliance structurelle.** La souplesse structurelle du robot est également paramétrable, ou plutôt ajustable. Bien sûr de façon beaucoup moins pratique puisque l'ajustement en question se fait concrètement par une intervention physique sur le robot. Néanmoins, le contrôle que nous avons sur la conception mécanique nous le permet.

D'abord, la raideur des ressorts peut-être changée et ajustée. De même, la souplesse des matériaux comme le silicone peut également être modifiée comme nous l'avons vu à la Section 3.1.2.

L'usage d'amortisseurs apporte une souplesse très significative dans la structure globale du robot. Quantitativement, la course de la liaison linéaire amortie de la hanche est de l'ordre [15] de 10% de la longueur totale de la jambe. Durant la marche, cette course est utilisée à 90%, ce qui montre une utilisation très significative de l'amortisseur. En revanche, l'ajustement de l'amortisseur est un processus fastidieux consistant à faire varier

- la raideur du ressort,
- la viscosité de l'huile,
- la surface de l'orifice permettant le transfert de fluide

L'introduction de ces liaisons amorties a amélioré la stabilité du robot de façon très significative.

L'usage d'amortisseur en revanche ne simplifie pas la compréhension du contrôle. Cependant, le robot mesure la position des liaisons linéaires verticales intégrées dans les hanches. Il s'agit d'une information substantielle : sachant que la liaison est contrôlé par un amortisseur, il donne une indication sur la force de réaction déployée par la jambe lorsqu'elle est au sol (voir Figure 3.7). Ainsi lorsque les deux pieds sont au sol, elle donne une approximation empirique de la position du ZMP dans le plan frontal. Au passage, elle indique également les moments ou le pied est flottant ou au sol.

Cette flexibilité joue un rôle important (voir http ://www.youtube.com/watch ?v=y8S0wQvJvXc). Elle absorbe les chocs et les perturbation, notamment à l'impact du pied durant la marche.

### 3.3.2 Contrôleurs actifs

La primitive motrice de maintien d'équilibre repose également sur un contrôle actif, via les moyens d'action sur la structure suivants :

- Mouvement de parallélogramme déformable sur la colonne vertébrale. Ce mouvement est utilisé dans le plan sagittal et également dans le plan frontal. Grâce aux articulations supérieures et inférieures de la colonne vertébrale. Il permet de modifier la position du centre de gravité tout en minimisant le moment subit par la partie supérieure du corps.

15. Nous utilisons plusieurs configurations

Cette action est utilisée par un contrôleur proportionnel-derivée sur le gyromètre.

- Position des chevilles et des genoux dans le plan sagittal. Nous utilisons ces articulations pour rétablir la verticalité au moyen de deux contrôleurs de type PID prenant en entrées le gyromètre ainsi que l'accéléromètre.
- La rotation sagittale du bassin. Nous utilisons cette articulation pour induire des mouvements précis du centre de gravité et pour maintenir le bassin horizontal. Cela est fait au moyen de deux contrôleurs de type PID prenant en entrées l'accéléromètre et le gyromètre.
- La position du centre de gravité. Pour ce faire, nous utilisons la cinématique inverse des pieds pour contrôler leur position relative au centre de masse (évaluée de façon approximative) dans un repère cartésien.
  - Nous utilisons des mouvements horizontaux du centre de masse pour corriger à basse fréquence l'équilibre du corps.
  - Nous utilisons des mouvements horizontaux à haute fréquence pour absorber les perturbations enregistrées par l'accéléromètre (toujours via un contrôleur de type PID).
  La selection de fréquence est faite au moyen de techniques de lissage des signaux d'entrées (moyenne "discounted", et/ou approximation linéaire du signal par des méthodes de moindres carrés).

Durant la marche, nous utilisons également un contrôleur de plus haut niveau : l'amplitude horizontale et verticale des pas. Les pas sont déterminés par des splines de base dont les paramètres (l'amplitude) sont ajustés en temps réel. Un point important est que les pas du robot ne sont pas issus d'un calcul planifié, mais de mouvement fixés à l'avance (au moyen de splines et de fonctions périodiques, en l'occurrence de sinusoïdes) dont les paramètres sont régulés en temps réel, les splines décrivant la trajectoire des pieds dans l'espace cartésien.

Un contrôleur linéaire de type PID est alors utilisé pour réguler la position du pied à l'impact de sorte à corriger la verticalité.

De façon résumée, le robot corrige son équilibre en régulant la longueur de ses pas.

*L'équilibrage dans le plan frontal* repose beaucoup sur la compliance : le système de triple pendule mis en mouvement par la base de la colonne vertébrale, additionné au système d'amortissement des liaisons linéaires verticales des hanches assure une stabilité importante dans le plan frontal. A cela s'ajoute un contrôleur linéaire actionnant les chevilles et les hanches, activé (au moyen d'une spline) dans les positions latérales extrêmes de la marche pour corriger la verticalité.

Enfin, nous travaillons actuellement plusieurs améliorations :

- L'intégration de capteurs de pression sous les pieds (4 par pied). Cela pourra nous donner une approximation plus précise du centre de pression. Il subsiste une difficulté d'intégration pour assurer une bonne adhérence du pied de sorte à distribuer de façon satisfaisante la pression sur le capteur. Les capteurs sont actuellement noyés dans une couche de silicone.

- Par ailleurs, nous souhaitons intégrer l'appareil de vision de Sigmaban dans le contrôleur d'équilibrage. Nous l'utilisons d'ores et déjà dans l'évaluation du cycle de marche, mais pour des raisons de temps de calcul, son utilisation est difficile dans l'état actuel à cause

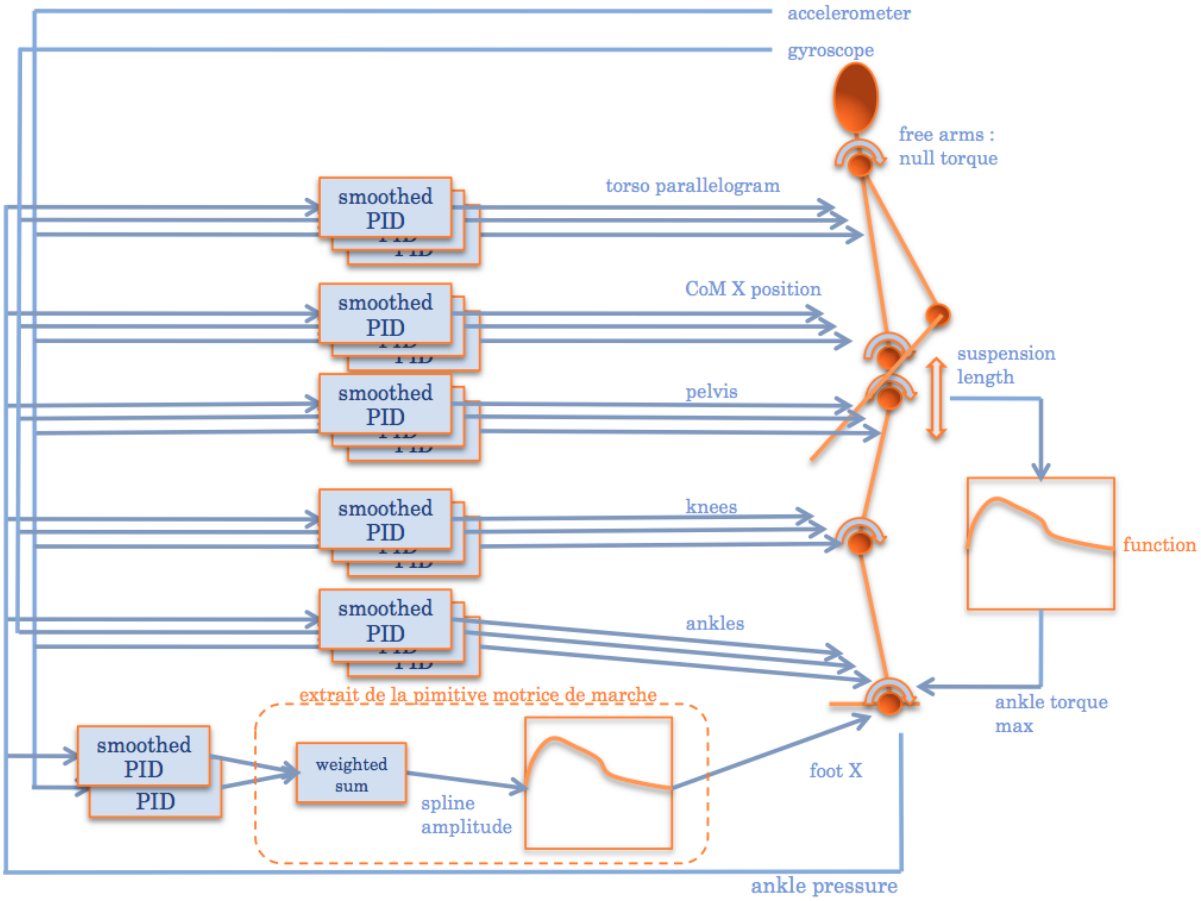


FIGURE 3.6 – Détail de l'architecture de la primitive motrice d'équilibrage dans le plan sagittal. Chaque bloc "Smoothed PID" consiste en un bloc contrôleur de type PID, enchaîné à un filtre lissant le signal. Il comporte plusieurs paramètres. Notons que le couple maximum des chevilles est également contrôlé dynamiquement par l'élongation des amortisseurs. Le but est de rendre la cheville compliante à l'impact du pied au sol pour améliorer l'adhérence et limiter le choc.

du délai impliqué dans la boucle de rétro-action. En effet, l'intégration d'un point de visée absolu nécessite une phase d'analyse d'image non-négligeable [16], introduisant un délai d'environ 150ms. Nous sommes actuellement en phase de test d'une nouvelle solution.

## 3.4 Marche dynamique semi-passive

### 3.4.1 Marche semi-passive

En considérant la primitive motrice d'équilibrage précédente, mais sans les pas, l'action de perturbations externes latérales et périodiques (voir video http ://www.youtube.com/watch ?v=gKEjkckxzBU engendre une marche semi-passive.

16. Rappelons que nos robots sont autonomes et embarquent tous leurs calculs

Alors que le robot utilise le même contrôleur pour son équilibrage, cette perturbation, amplifiée par le système de triple pendule, engendre des pas vers l'avant, conséquence de l'élasticité des jambes et des pieds. En effet, pendant la phase où le pied est au sol, il accumule de l'énergie qu'il restitue lorsque le poids est transféré sur l'autre jambe sous la forme d'une translation horizontale du pied.

### 3.4.2 Marche dynamique engendré par auto-perturbation

Forts de cette observation, nous avons conçu une primitive motrice de marche en utilisant un primitive motrice périodique ajouté à la primitive motrice de stabilisation. Il s'agit d'une marche dynamique engendrée par une perturbation provoquée par le robot lui-même, que l'on peut dès lors appeler *marcheur passif motorisé*, ou *marcheur semi-passif*.

La primitive motrice de marche est divisée en 2 modules distincts : une partie active conduite par les jambes et le bassin, mixant le transfert latéral de masse et les pas. Et une partie réactive consistant en la primitive motrice d'équilibrage décrite précédemment.

Le robot utilise le système de triple pendule décrit précédemment comme accumulateur local d'énergie. Le mouvement pendulaire est entretenu par le mouvement des jambes et du bassin. Et dans le même temps, le pendule contribue de façon significative au transfert latéral de poids : d'une part il modifie la distribution globale de la masse, et d'autre part il induit un moment de réaction de la partie inférieure du corps via le bassin.

Notons que l'utilisation du bassin, indépendamment du torse pour les pas est largement inspirée de la marche humaine (voir e.g. [116, 97]).

Par ailleurs, le cycle de marche peut être contrôlé de sorte à faire pivoter le robot ou le faire avancer à vitesse variable, en modifiant l'amplitude des splines gauche ou droite. Une variante lui fait faire des pas chassés. Cela réduit le contrôle de la marche à deux paramètres déterminant la somme des amplitudes d'une part et d'autre part leur différence relative.

**Application à la RoboCup.** Dans le cadre de la RoboCup, une tâche importante est la poursuite du ballon. Notre système marche comme suit : le cou est asservi pour maintenir le ballon au centre de l'image, de sorte à ne pas perdre la vision du ballon. Ensuite, en combinant la position du cou et la position du ballon dans l'image, on évalue la position du ballon relativement au robot. Puis on asservit la marche (pilotée par les deux paramètres vitesse/direction mentionner plus haut), de sorte à approcher le robot du ballon. Un contrôleur (de type PID) asservit la direction, maintenant le ballon en face du robot, et un autre contrôleur le fait avancer. L'action de ce dernier est atténuée par l'erreur en orientation, le faisant ralentir s'il est dans la mauvaise direction. Notons sur ce point que le comportement intègre plusieurs modes suivant la distance au ballon, correspondant à plusieurs modes de la marche (tourner par une rotation, se déplacer sur le côté par des pas chassés, etc.). Ainsi, l'approche du ballon n'est pas basée sur une planification des pas, mais sur un asservissement (à base de PID) de la marche pour converger vers le ballon.

Remarquons que l'action de tir est une primitive motrice à base de splines qui enregistre les mouvements des 4 membres. Cette primitive motrice est développée par itérations essais/erreurs en expérimentant directement sur le robot. Elle est associée à la primitive motrice d'équilibrage postural, nécessaire pour stabiliser le robot à la fin de l'action en particulier.

## 3.5 Ajustement expérimental des paramètres

La question est de converger vers un contrôleur acceptable. Se pose alors le problème de l'évaluation objective du système.

### 3.5.1 Comprendre les synergies du système.

Un robot humanoïde est un objet physique très complexe, intégrant un grand nombre de degrés de liberté, des segments flexibles, des éléments souples, du jeu, etc. Le processus de marche, pas après pas, est un système dynamique difficile à appréhender.

Notons qu'une modélisation/simplification théorique du système engendre déjà un système dynamique difficile à comprendre. La figure 3.1 montre qu'un exemple réduit au plus simple engendre déjà un comportement très irrégulier (voir [71] pour une étude approfondie). Pour autant, rechercher un cycle de marche stable se traduit par rechercher une trajectoire quasi-périodique convergeant vers un attracteur. Dans le cas du marcheur passif simplifié, une technique naturelle (illustré i.e. dans [48]) consiste à rechercher de façon expérimentale (néanmoins sur un support de simulation) des paramètres du système ainsi que des conditions initiales assurant un jacobien contractant du système dynamique (i.e., dont les valeurs propres sont expérimentalement de module strictement inférieur à 1), ce qui assure une marche stable, robuste dans une certaine mesure [17] aux perturbations.

En collaboration avec H. Gimbert et G. Passault, nous avons mis au point une suite logicielle permettant de visualiser et d'analyser les différentes sorties du contrôleur du robot. Cela nous permet d'avoir des retours immédiat du comportement du robot. Cet environnement est crucial dans notre méthode car il permet d'investiguer de façon très fine tous les paramètres du robot.

### 3.5.2 Indicateurs du cycle de marche

#### Elements quantitatifs

A partir de là, une phase essentielle du processus de développement est la synthèse d'indicateurs. En effet, l'objectif global est d'obtenir une marche robuste. La première mesure évidente pour cela est d'observer si le robot tombe ou pas, et dans quelles conditions. Cependant, cette mesure, de nature discrète, et même binaire, n'est pas adaptée à un processus de convergence, i.e., l'ajustement de paramètre est un processus de convergence. En effet, un tel processus nécessite une mesure dans un espace continu.

La figure 3.7 montre l'extraction de certains signaux du robot pendant la marche. On peut y observer :

- les courbes rouges et marrons dans la partie supérieure indiquent l'angle de vision d'un point fixe. Un repère visuel fournit un élément de positionnement absolu précieux. Le crénelage de la courbe est du à la résolution de la caméra embarqué dans la tête du sigmaban. Les deux courbes suivantes forment le lissage de ces signaux.
- les courbes suivantes, marrons et bleu ciel relèvent l'élongation des liaisons linéaires verticaux amortis situés dans les hanches. On les interprète comme une approximation de la force de réaction déployée par le pied sur le sol. A noter que la nouvelle version de

17. sous reserve de rester dans une zone à Jacobien contractant

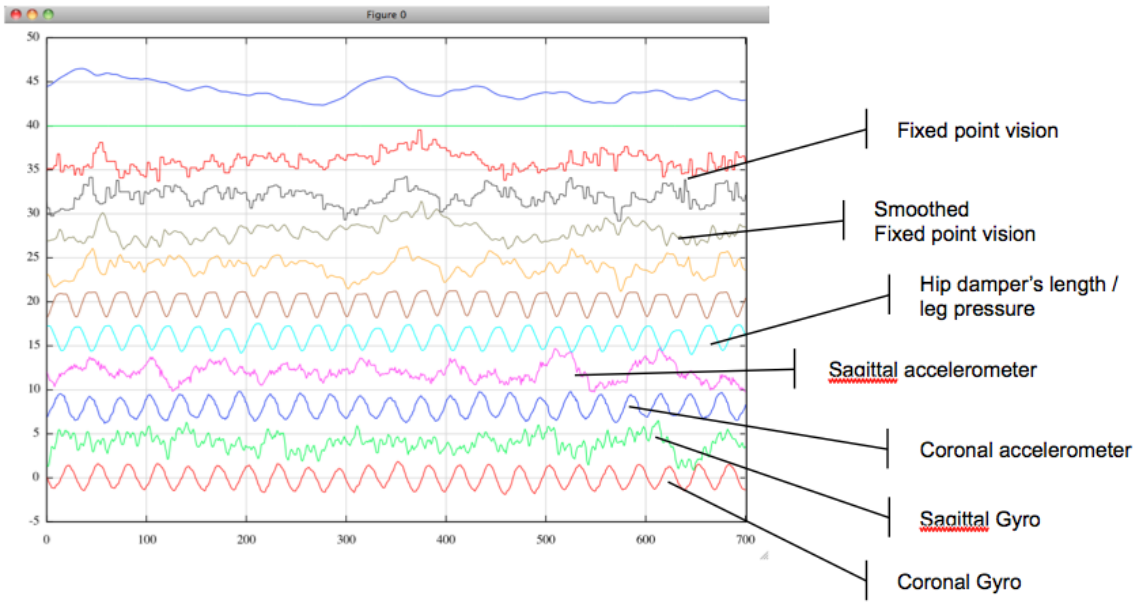


FIGURE 3.7 – Signaux capteurs

Sigmaban intègre un réseau de capteur de pression dans les semelles, destiné à indiquer la position du point d'application de la force.

- les quatres dernières courbes relèvent respectivement les accéléromètres et les gyromètres situés au niveaux du bassin.

Notons que Sigmaban intègre également la vision, ou plus exactement la position d'un point de référence dans le champs de vision. Actuellement, pour des raisons techniques, nous n'avons pas encore d'utilisation active de ce paramètre, i.e., dans le contrôleur. Mais c'est là un trait important que nous comptons développer dans le futur.

Les figures 3.7 et 3.8 montre le calcul d'indicateurs.

Une étape primordiale est le *le recalage* des signaux. Elle est illustrée dans la partie haute de la figure 3.8. Pour ce faire, nous utilisons classiquement la transformée de Fourier rapide (FFT) pour extraire une approximation de la phase du signal quasi-périodique produit par l'élongation des liaisons passives linéaires des hanches. C'est une approximation de la phase du cycle de marche.

Notons bien que le cycle de marche n'est pas nécessairement périodique, comme l'a montré l'exemple du marcheur passif simplifié, figure 3.1. Sur un système comme Sigmaban ou Acroban, la situation n'a aucune raison d'être plus favorable de ce point de vue.

Le recalage des signaux permet d'obtenir un enregistrement des transitions du système.

Nous considérons plusieurs indicateurs :

• L'amplitude de l'accéléromètre et du gyromètre dans le plan sagittal. Cet indicateur quantifie les mouvements de balancement dans le plan sagittal. Cette amplitude est à minimiser eu égard à la difficulté de l'équilibrage postural.

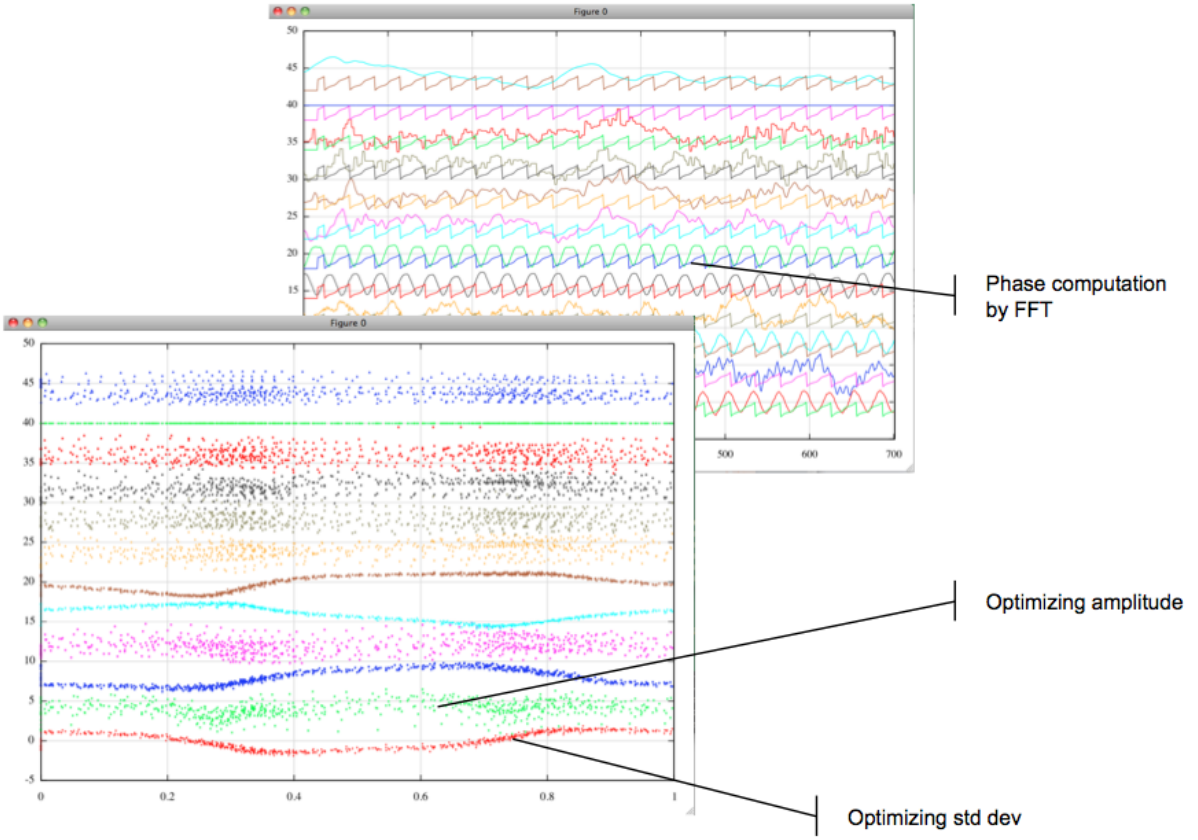


FIGURE 3.8 – Calcul d'indicateurs

• L'énergie, i.e. la norme $L^2$, de l'accéléromètre et du gyromètre dans le plan sagittal [18]. Cet indicateur est également à minimiser pour les mêmes raisons.

• Le déterminisme du cycle de marche est également important. Il est un élément essentiel de sa validation. Nous évaluons ce déterminisme de façon globale par l'écart type des signaux recalés de l'accéléromètre et du gyromètre (voir figure 3.8).

• Nous proposons également un indicateur que nous considérons comme *essentiel*, bien que difficile à évaluer. Il s'agit du maximum des modules des valeurs propres du Jacobien du système dynamique sous-jacent au cycle de marche. En simulation (comme pour le marcheur simplifié), outre les raisons pratiques de mise en oeuvre, la possibilité du choix des conditions initiales facilite cette évaluation. Dans notre contexte, l'évaluation provient d'expériences réelles sur le robot, dont nous ne pouvons pas assurer la mise en conditions initiales précises. Néanmoins, notre méthode consiste à se baser sur un enregistrement suffisamment long pour obtenir un ensemble de transitions suffisamment riche pour garantir une évaluation probante.

Cet indicateur est fondamental, car il constitue une *validation de la stabilité du système*. En effet, il est une preuve de l'existence d'un attracteur pour le cycle de marche. De plus, il permet

18. L'usage de la norme $L^2$ a un caractère arbitraire parmi les normes $L^n$.

de *quantifier* les perturbations admissibles sous lesquelles le système continue sa marche.

Notons que la symétrie de la marche est un élément relativement secondaire. En effet, la facture des robots tels Acroban ou Sigmaban reste relativement approximative et la symétrie de leur structure n'est assurée qu'approximativement. Dans ces conditions, il est difficile de viser un cycle de marche absolument symétrique.

L'ajonction de nouveaux capteurs est un processus constant dans notre démarche. Nous souhaitons approcher l'état de la plateforme, au sens de l'état du système dynamique sous-jacent, de la façon la plus précise et la plus complète possible. Et pour transformer le système d'indicateur qui nous permet actuellement d'évaluer le cycle de marche en un système prédictif.

**Elements qualitatifs**

Les indicateurs quantitatifs décrit précédemment sont complétés par des éléments qualitatifs qui nous aident à évaluer la robustesse et la fiabilité de la marche.

Notons que le même système en éliminant la souplesse des articulations de la colonne vertébrale et des bras montre un accroissement significatif de la consommation, et une dégradation de la robustesse de la locomotion. On observe dans ce cas une propagation du choc à l'impact du pied au sol visible dans les bras.

Les expériences conduites en collaboration avec M. Lapeyre, étudiant en thèse au sein de l'équipe Inria Flowers, montrent une robustesse significative du cycle de marche, bien que restant à améliorer (voir Figure 3.9). Ces robots sont capables de marche malgré des perturbations diverses non négligeables. Notons que la marche reste également robuste en considérant un sol dont la pente est variable.

Par ailleurs, nous évaluons également des primitives motrices par des tests de robustesse constants nous conduisant à perturber le système de diverses façons (en le poussant, en lui soumettant des chocs horizontaux, en le faisant marcher sur un skateboard...) Ces tests, par leur facilité de mise en oeuvre sont des éléments utiles, car ils permettent une évaluation rapide de modifications de paramètres. Cela est possible grâce à la robustesse mécanique et au gabarit du robot qui lui permet de tomber sans risque.

### 3.5.3 Apprentissage

Le matériel présenté dans la Section 3.5.2 ouvre la voie à l'expérimentation de procédures d'apprentissage.

Au fil des développements, nous avons testé des procédures semi-automatiques ad-hoc d'optimisation de paramètre, destinées à accélérer le processus de réglage de paramètres.

Cependant, notre but dans le futur immédiat est l'expérimentation de procédure d'apprentissage. Pour cela plusieurs directions sont privilégiées.

Pour limiter l'espace de recherche, il faut limiter les paramètres de la primitive motrice, en ne considérant que les essentiels. Cette opération a un caractère ad-hoc dont il est difficile de se départir. Elle est pourtant essentielle pour assurer la faisabilité du processus d'apprentissage. Les éléments de mesure qualitatifs et quantitatifs décrits précédemment peuvent être d'une aide précieuse.

Une première grande classe de méthode consiste à apprendre la marche en agissant directement sur les paramètres et en se dotant de critères quantitatifs permettant de mettre au point des

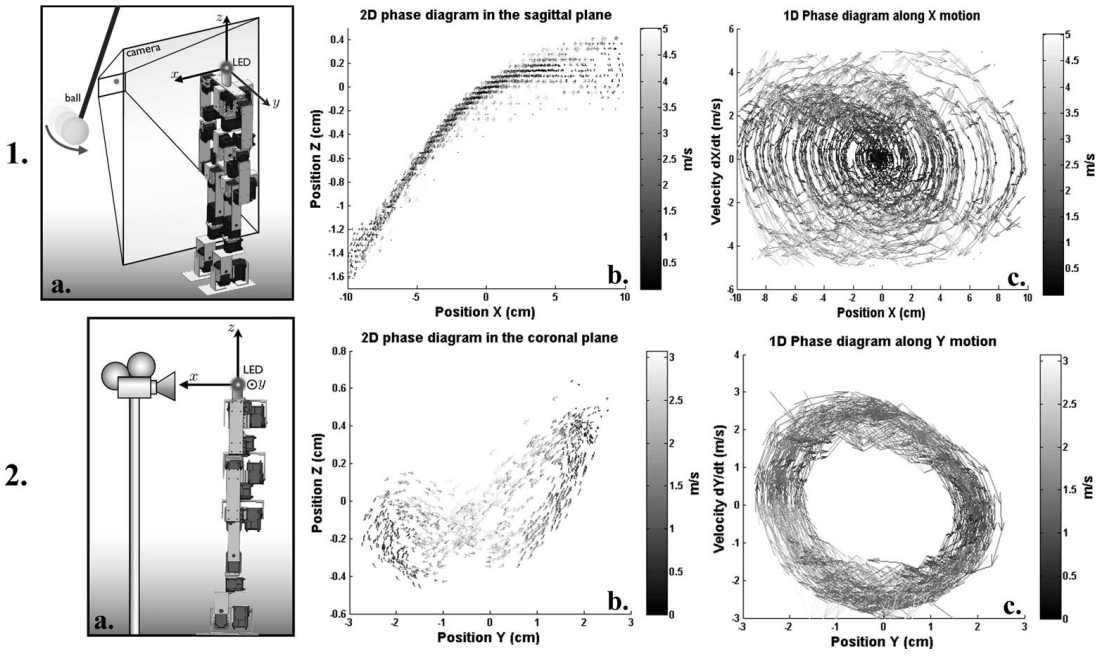


FIGURE 3.9 – **1.a.** *Protocole Experimental 1 :* Nous utilisons une balle (154g) suspendue au bout d'une corde (87cm) produisant en se balançant une impulsion horizontale dans le plan sagittal sur le torse du robot avec une énergie de 1.3J, que nous appelons perturbation externe. **1.b.** Ce diagramme montre la dynamique de l'équilibrage postural dans le plan sagittal sous l'action de perturbations externes. Il represente un système clos, convergeant vers une position neutre, rendant le robot stable. **1.c.** Il s'agit du diagramme de phase pour le mouvement $X(t)$ de la dynamique de l'équilibrage postural sous l'action de perturbations externes. Le diagramme montre une spirale convergente nécessitant 7 tours avant d'atteindre la position neutre. Cela montre un comportement stable bien que lent pour la convergence. **2.a.** *Protocole expérimental 2 :* Le robot est placé en face d'une caméra mesurant sa position dans le plan de l'image $(y, z)$. Le robot est actionné par une primitive motrice périodique produisant un transfert latéral de poids d'un pied sur l'autre. **2.b.** Ce diagramme montre la dynamique du transfert de masse dans le plan frontal. Le point décrit une courbe fermée quasi-périodique en forme de 8, montrant la stabilité du système. **2.c.** Il s'agit du diagramme de phase de $Y(t)$ lors du transfert de poids. Le diagramme forme un cercle montrant la stabilité et le déterminisme du système.

fonctions de récompense, dans le cadre d'apprentissage par renforcement. Plusieurs travaux ont été conduits dans ce sens, en se servant souvent du nombre de pas (voir e.g. [82, 83]). D'autres méthodes se basent également sur des techniques d'optimisation (voir [53], utilisée notamment par les Darmstadt dribblers)

Nous souhaitons mettre en oeuvre ce type de méthode en se basant sur les éléments quantitatifs décris dans la Section 3.5.2.

Une autre direction qui nous intéresse fortement consiste à apprentre le comportement du système dynamique sous-jacent au cycle de marche. Pour ce faire, nous souhaitons explorer l'existence d'une fonction de potentiel régissant la primitive motrice du système décrit au moyen de ces entrées comme par exemple les signaux de la Figure 3.7. Et ce pour doter le système d'une fonction de prédiction d'état, de sorte par exemple à anticiper les déséquilibres.

**Apprentissage développemental**

Dans notre démarche, d'une importance forte sont les travaux de M. Lapeyre auxquels nous avons collaboré avec P.-Y. Oudeyer. Ces travaux ont consisté à mettre en oeuvre une expérience d'apprentissage développemental à partir d'un modèle d'Acroban plongé dans un simulateur physique. Il s'agit de permettre à Acroban d'apprendre à marcher selon un plan développemental précis inspiré de l'homme. Dans ce modèle le robot se sert d'un charriot de marche, en s'inspirant de l'apprentissage de la marche des petits enfants. En effet, ce charriot permet d'introduire le problème de l'équilibrage postural de façon graduelle, en libérant les roues petit à petit, tout en restant dans un environnement réaliste.

Cette voie est très prometteuse, eu égard au fait que l'homme et son processus d'apprentissage demeurent le principal exemple réussi d'apprentissage de la locomotion bipède.

### 3.5.4 Interaction Physique

L'interaction physique homme/robot, c'est-à-dire la possibilité pour l'utilisateur d'interagir physiquement avec le robot, est probablement un des modes d'interaction les plus naturels eu égard au fait que les éléments constitutifs d'un robot sont avant tout un calculateur muni d'un appareil sensori-moteur.

En collaboration avec P.-Y. Oudeyer, nous avons expérimenté un mode d'interaction en tirant parti de la compliance du robot. Dans ce mode, les bras et la colonne vertébrale sont rendus souples de sorte à offrir une résistance minime à l'interaction de l'utilisateur. L'utilisateur peut typiquement lui prendre les bras et le manipuler.

Nous avons développé ce mode dans le but de rendre l'expérience utilisateur la plus intéressante possible en donnant une certaine impression de vie au robot. Principalement :

- Le robot "écoute" la position des articulations manipulées par l'utilisateur (les épaules et les coudes). Il en tire certains comportements : Il chercher à s'orienter vers l'utilisateur, en utilisant la torsion de la colonne vertébrale ainsi que la torsion des hanches. Il cherche à opposer une résistance progressive (proportionnelle à l'ouverture des articulations des épaules) à l'utilisateur, en réglant le taux de compliance dans toutes les articulations opérant dans le plan sagittal.
- Il continue à s'équilibrer activement, selon la primitive d'équilibrage évoquée précédemment.

Il y a aussi un comportement interessant remarqué par P.-Y. Oudeyer, apparu fortuitement sans l'avoir prévu, qui consiste à guider Acroban par les mains durant la marche. Soulignons que cette interaction n'est absolument pas programmée de façon volontaire dans le robot d'une quelconque façon. Physiquement, on tire parti de la flexibilité d'Acroban, accompagnée par les réactions d'équilibrage qui le font suivre son guide.

Ces modes d'interaction ont été testés de nombreuses fois durant des démonstrations auprès d'un public varié, incluant des enfants. Qualitativement, on observe un intérêt certain des personnes ayant accès au robot ; intérêt qu'ils expriment souvent comme une sorte d'illusion de vie, et ce, malgré l'apparence physique du robot, fort éloignée de l'homme. Point intéressant qui rappelle que cette illusion de vie classiquement provoquée par les robots auprès des gens provient en premier lieu de l'appareil sensori-moteur. L'interaction physique accroît ce phénomène.

# Chapitre 4

# Robotique Agricole

## 4.1 Un robot autonome pour l'entretien de la vigne

Il s'agit de concevoir un robot autonome, le *Vitirover* pour l'entretien de la vigne. Le Vitirover est un robot lent, mais entièrement autonome, tirant son énergie de panneaux solaires embarqués et optimisés pour le robot. En particulier, la technologie permet de maintenir une production d'énergie maximale y compris lorsque le panneau est partiellement ombragé.

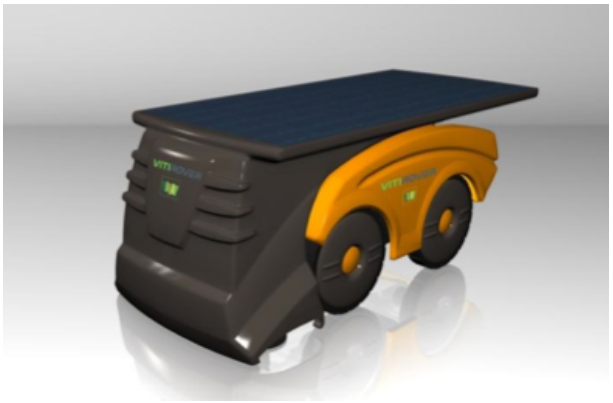

FIGURE 4.1 – Le vitirover (image ©Vitirover)

Le but du robot est l'entretien de la végétation adjacente à la vigne ; l'objectif est de conserver un éco-système équilibré propice à la culture du vin. Puis à terme, de fournir un outil de monitoring pour le processus de croissance du vignoble.

Nous collaborons à l'élaboration de stratégie motrice de déplacement du robot. Au niveau local pour mettre au point une locomotion robuste écartant tout blocage ; à niveau plus élevé pour assurer un traitement uniforme du vignoble.

## 4.2 Plantations robotisées

Nous collaborons avec la société ATH spécialisée dans les technologies hydrauliques pour la mise au point d'une machine de plantation viticole automatique, guidée par un système de

positionnement GPS centimétrique. Le premier prototype est depuis peu opérationnel. De façon plus détaillée, il s'agit d'une mini pelle robotisée destinée à planter des piquets de palissage dans la vigne. Le bras est équipé d'un outil d'enfoncement permettant de planter des piquets. Nous avons enrichi le système hydraulique de distributeurs à bobine contrôlés par un bus série (de type CAN). Chaque élément mobile est équipé d'inclinomètres pour obtenir la position (en tout 5 inclinomètres, auquel s'ajoute un capteur de distance à ultrasons pour la position de la rotation horizontale). Le prototype suivant (en cours de développement) équipera chaque vérin d'un capteur de position interne pour gagner en précision et éliminer les problèmes d'interdépendance des inclinomètres. Sur cette base, nous avons développé l'*ensemble de la chaîne de contrôle robotique du système*, incluant le modèle dynamique de la machine et la planification des mouvements.

Par ailleurs, la tête du bras est équipée d'un GPS centimétrique. Pour des raisons de contraintes budgétaires, *nous avons développé notre propre système de positionnement centimétrique* sur la base de techniques de double différence en utilisant des puces GPS standard à bas coût. De façon classique, le calcul du positionnement fait appel à une analyse de signal reposant sur la mise en oeuvre d'un filtre à particule notamment. Par ailleurs, nous avons mis au point une série d'heuristiques pour résoudre les problèmes de perte de signal. Soulignons que ce développent utilise dans puces GPS standards.

Le premier prototype est aujourd'hui opérationnel et a été présenté au salon Vinitech-Sifel 2012.

# Conclusion et Perspectives

Nous formulons ici quelques remarques d'ordre général en guise de conclusion, et nous traçons les grandes lignes de nos recherches pour l'avenir.

## Analyse de code exécutable

Pour ce qui concerne les méthodes formelles et la sécurité, nous souhaitons orienter nos travaux à venir vers l'analyse de code exécutable. L'enjeu est très important. En particulier pour ce qui concerne la sécurité logicielle. En effet, elle est centrale dans la protection contre les malwares, omniprésents dans la vie quotidienne. Elle est également au coeur des problématiques de protection du logiciel contre le piratage.

Dans le même temps, l'étude du code exécutable révèle un paradoxe surprenant. Elle incarne un décrochage relativement profond entre la théorie et la pratique. En effet, d'un point de vue théorique, la situation est relativement fermée eu égard à des résultats négatifs profonds. On peut songer à l'impossibilité de l'obfuscation de Barak. On peut également penser aux nombreuses questions indécidables que recèle l'analyse de code bas-niveau.

D'un autre côté, eu égard à l'enjeu, la protection de licence, ou encore l'analyse de virus suscite une énergie considérable dans l'industrie, pour un résultat parfois très relatif. Ainsi, de nombreuses techniques existent, qui parfois manquent d'un support théorique solide.

C'est là une situation très excitante où le contexte classique de l'analyse de programme y est remanié de façon profonde. On peut penser à l'abandon de la donnée a priori du graphe de flot de contrôle. Le projet de développement d'un environnement logiciel dédié est un challenge réel pour nous.

A moyen terme, plusieurs questions nous intéressent :

#### Reconstruction du graphe de flot de contrôle.

La reconstruction du code assembleur est une tâche qui peut s'avérer extrêmement complexe dans le cas d'un code obfusqué, comme un virus par exemple. Les nombreuses techniques d'obfuscation, allant jusqu'au code auto-modifiant, montrent que la question est indécidable en général. Pour autant, plusieurs techniques restent à mettre en oeuvre dans l'environnement Insight. Nous souhaitons contribuer à cela dans l'avenir.

#### Reconstruction de la structure fonctionnelle.

Dans le code exécutable, le découpage fonctionnel n'est pas explicite. Il est produit à la compilation. En l'absence de toute obfuscation, la reconstruction de ce découpage est possible,

bien que complexe. C'est un pas crucial dans la compréhension du code. Ensuite, si l'on considère du code obfusquée, la tâche peut devenir très difficile. Néanmoins, l'étude de cette question reste nécessaire.

**Decompilation.**

La décompilation consiste à interpréter un code exécutable sous une forme structurée, comparable à du code source dans un langage de plus haut niveau (typiquement le langage C). Il s'agit d'une part de reconstruire les structures de donnée utilisées dans le programme dans toute leur complexité, le point de départ étant les opérations et leurs types de base. D'autre part, il s'agit d'analyser le flot de contrôle du programme et d'en produire une forme également structurée (en blocs, boucles, etc). L'utilité, mais la difficulté aussi sont évidentes. La structuration fonctionnelle évoquée précédemment est une brique pour aboutir à cela.

## Robotique

La robotique a toujours suscité une fascination importante auprès du grand public. Le cinéma, la littérature, depuis les années soixante confirme très régulièrement cet intérêt intrinsèque. Mais il aura fallu des dizaines d'années pour mettre à la portée du grand public de véritables robots. C'est depuis peu le cas, et on observe aujourd'hui une réelle effervescence autour de la robotique, en particulier de la robotique personnelle, et de la robotique humanoïde également.

Ce déploiement progressif bénéficie très directement du développement massifs des technologies mobiles, en particulier des technologies embarquées. Il devra s'en distinguer au fil des années futures. Au plan technique, la route est encore longue. Elle est jalonnée de nombreux défis scientifiques et techniques (locomotion, préhension, segmentation, représentation du monde, etc). Et l'équation qui mettra en relation de véritables robots avec les attentes parfois fantasmatiques des consommateurs est une des inconnues de ce marché naissant.

Si les robots faisant l'objet d'un déploiement à grande échelle restent relativement limités aujourd'hui, ils sont pour autant de véritables succès commerciaux. Et cette nouvelle synergie entre ce marché qui trouve petit à petit son positionnement, et le monde de la recherche promet de belles avancées pour les années à venir. Cette problématique nous passionne.

**Apprentissage et optimisation.**

Dans le contexte de la locomotion, et en particulier de la locomotion bipède, les techniques basées sur l'apprentissage et d'optimisation constituent pour notre méthode une suite naturelle. De nombreux travaux ont construit aujourd'hui un corpus conséquent en matière d'apprentissage. Nous souhaitons explorer leur application dans le cadre de la marche bipède, question difficile encore largement ouverte. Nous souhaitons en particulier approfondir la question des fonctions de récompense (du point de vue de l'apprentissage par renforcement), question très liée aux indicateurs quantitatifs de la marche que nous avons introduit en Section 3.5. Nous souhaitons approfondir et développer ces indicateurs, comprendre leurs structures (sont-ils par exemple liés à des potentiels ?) et comprendre leurs interdépendances, et notamment les inscrire dans une stratégie d'apprentissage graduel de la locomotion bipede, une stratégie typiquement développementale, inspirée de l'humain.

L'évaluation du rayon spectral du jacobien du système dynamique lié au cycle de marche par l'expérience, voire par une stratégie d'apprentissage, est une question très importante pour nous.

Une autre question essentielle est la réduction des paramètres des primitives motrices de locomotion. Le nombre de paramètre de ces primitives est très important, et doit être réduit pour envisager des techniques exploratoires. A l'heure actuelle, cette réduction est arbitraire. Mais nous souhaitons étudier ce problème de façon plus uniforme, en suivant par exemple les pistes suggérées par les méthodes de type NMF (voir [72]).

L'optimisation et l'apprentissage forment une voie importante pour nous d'un autre point de vue : en effet, notre intérêt pour la robotique personnelle nous porte à nous intéresser à l'usage de matériel bas coût (moteurs, structures, etc). Dès lors, on ne peut s'appuyer sur l'hypothèse d'une adéquation parfaite entre le modèle et la plateforme concrète, et l'optimisation et l'apprentissage, en particulier sur la plateforme elle-même et non en simulation, peuvent permettre de combler ce manque.

**Morphologie.**

Illustrée par l'apprentissage développemental évoqué plus haut, l'inspiration biologique montre en effet une voie très excitante ; elle l'est également pour la morphologie qui distingue encore profondément le vivant des robots. Nous souhaitons comprendre quelles morphologies globales (l'architecture des articulations de la colonne vertégrale par exemple) ou locales (e.g. la forme des pieds) peuvent améliorer la stabilité ou la consommation énergétique. Les questions liées à la dynamique passive et semi-passive relève de cela. Nous souhaitons explorer le continuum entre la dynamique passive et l'usage de matériaux souples ou d'articulations libres ou amorties.

D'une part, nous souhaitons continuer ce type d'étude dans un cadre théorique, appuyé sur la simulation. La modélisation des différents matériaux que nous utilisons comme par exemple le silicone constitue un travail important pour doter d'outils de simulation probants.

Nous souhaitons continuer en parallèle l'expérimentation méthodique et systèmatique de structures utilisant des matériaux ou mécanismes souples (comme les amortisseurs).

La conception d'un quadripède peut être aussi une piste intéressante dans cette optique, en ceci qu'elle simplifie la problématique de l'équilibrage.

**Interaction Homme/Robot.**

Nous nous intéressons également aux problématiques d'interaction. Dans ce domaine, nous souhaitons nous concentrer sur l'interaction physique.

Plusieurs types d'application sont envisageables dans cette optique, parmi lesquelles la mise en oeuvre de tâches collaboratives entre les hommes et les robots. Cependant, en dehors de toute application, nous avons constaté lors de multiples démonstrations d'Acroban un intérêt certain du public pour l'interaction physique avec le robot, durant la marche, ou même immobile.

Par ailleurs, ces dernières années ont vu apparaître, en particulier en Asie, plusieurs prototypes de robots résolument orientés vers des applications de divertissement (”entertainment”). Mentionnons par exemple la célèbre HRP-4C dont les démonstrations de danse (voire de chant !), sont aujourd'hui célèbres dans le monde entier. Nous souhaitons explorer l'interaction physique dans ce contexte.

La possibilité d'interagir avec l'utilisateur, en même temps que l'exécution de tâches complexes comme la marche par exemple pose des problèmes très importants du point de vue du contrôle. Les modes d'interaction eux-même méritent la mise en oeuvre d'expériences utilisateurs systématiques pour en évaluer l'intérêt.

**RoboCup.**

Pour les années futures, la participation à des événements comme la RoboCup est pour nous porteuse de challenges pour améliorer la motricité bipède, et nous souhaitons développer cette activité. Elle pose également les questions fondamentale de la robotique autonome, liées à l'énergie, aux systèmes embarqués, à la représentation de l'environnement, à l'intégration. Il faut enfin souligner l'apport pédagogique très substantiel de cette activité, génératrice de projets extrêmement motivants pour les étudiants.

## Robotique Agricole

Dans un tout autre domaine, la robotique agricole nous intéresse en ceci qu'elle constitue un terrain de déploiement immédiat pour les nouvelles technologies robotiques, où l'adaptabilité est nécessaire, sans pour autant requérir une sûreté de fonctionnement réddibitoire. Les problèmatiques de vision, de locomotion, de préhension, de navigation y sont immédiatement présentes.

C'est de façon plus globale une activité de transfert, importante dans notre mission, en ceci qu'elle contribue à la synergie excitante entre les contraintes liées à l'application concrète et la recherche.

**Positionnement.**

Nous projetons de nous intéresser plus précisemment au positionnement et à la navigation. Sur la base de notre système de positionnement centimétrique par satellite, nous souhaitons explorer les possibilités de couplage entre les ordres de commande et la résolution statistique de la solution GPS. Cela pour améliorer la qualité de notre positionnement et par voie de conséquence du contrôle.

**Vitirover : supervision.**

Nous collaborons également à la conception du robot Vitirover, dans sa seconde version. Il s'agit là de se doter d'un outil robotique pour la supervision de la culture de la vigne.

Le prototype actuel assure la locomotion, y compris au plan énergétique, et propose des fonctionnalités d'entretien de la végétation. L'évolution à venir vise à le doter d'outil de monitoring, en particulier une caméra portée par un bras. Il s'agira de développer cette partie en tenant compte des contraintes très fortes liées à l'énergie (le robot est complètement autonome, et fonctionne à l'énergie solaire), mais également aux coûts, qui doivent rester raisonnables pour envisager un déploiement à grande échelle.

Ainsi, en collaboration avec l'IMS en particulier, il s'agira d'identifier et de repérer certaines maladies notamment, de fournir une traçabilité pied à pied, etc.

# Bibliographie


[1] M. Abadi and L. Lamport. Composing Specifications. *ACM Transactions on Prog. Lang. and Systems (TOPLAS)*, 15(1) :73–132, 1993.

[2] A. Albu-Schaffer, O. Eiberger, M. Fuchs, M. Grebenstein, S. Haddadin, C. Ott, A. Stemmer, T. Wimbock, S. Wolf, C. Borst, et al. Anthropomorphic Soft Robotics-from Torque Control to Variable Intrinsic Compliance. In *14th International Symposium on Robotics Research*, 2009.

[3] H. R. Andersen, C. Stirling, and G. Winskel. A compositional proof system for the modal mu-calculus. In *9th Symp. on Logic in Comp. Sci. (LICS'94)*, pages 144–153. IEEE Comp. Soc. Press, 1994.

[4] S. O. Anderson, M. Wisse, C. Atkeson, J. Hodgins, and G. Zeglin. Powered Bipeds Based on Passive Dynamic Principles. In *Proc. of IEEE International conference on Humanoid Robots*, 2005.

[5] T. Aura and D. Gollman. Software license management with smartcards. In *Usenix Worksh opon Smartcard Technology (Smartcard'99)*, 1999.

[6] B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. Vadhan, and K. Yang. On the (im)possibility of obfuscating programs. In *Lecture Notes in Computer Science*, pages 1–18. Springer-Verlag, 2001.

[7] S. Bardin and P. Herrmann. OSMOSE : automatic structural testing of executables. *Software Testing, Verification and Reliability*, 21(1) :29–54, 2011.

[8] S. Bardin, P. Herrmann, and F. Vedrine. Refinement-based CFG reconstruction from unstructured programs. In *Proc. of 12th International Conference on Verification, Model Checking, and Abstract Interpretation (VMCAI'2011)*, volume 6538 of *LNCS*, pages 54–69, Austin, 2011. Springer.

[9] B. Barras et al. The Coq Proof Assistant Reference Manual – Version V6.1. Technical Report 0203, INRIA, 1997.

[10] G. Barthe, P. Courtieu, G. Dufay, M. Huisman, S. Mello de Sousa, G. Chugunov, L.-A. Fredlund, and D. Gurov. Temporal Logic and Toolset for Applet Verification : Compositional Reasoning, Model Checking, Abstract Interpretation. Technical report, VERIFICARD Project, `http://www.verificard.org/`, Sept 2002. Deliverable 4.1.

[11] G. Barthe, D. Gurov, and M. Huisman. Compositional Verification of Secure Applet Interactions. In *Fundamental Approaches to Soft. Eng. (FASE'02)*, volume LNCS 2306, pages 15–32, 2002.

[12] M. Bauderon and H. Jacquet. Node Rewriting in Graphs and Hypergraphs : A Categorical Framework. *Theoretical Computer Science*, 266(1-2) :463–487, 2001.

[13] G. Betarte, B. Chetali, E. Gimenez, and C. Loiseaux. Formavie : Formal modelling and verification of the javacard 2.1.1 security architecture. In *e-smart 2002*, 2002.

[14] B. Bigge and I. Harvey. Humanoid Robots in Waseda University Hadaly-2 and WABIAN. *Journal of Autonomous Robots*, 12(1), 2002.

[15] C. Borst, T. Wimböck, F. Schmidt, M. Fuchs, B. Brunner, F. Zacharias, P. R. Giordano, R. Konietschke, W. Sepp, S. Fuchs, C. Rink, A. Albu-Schäffer, and G. Hirzinger. Rollin' justin - mobile platform with variable base. In *ICRA*, pages 1597–1598, 2009.

[16] J. Buchli, M. Kalakrishnan, M. Mistry, P. Pastor, and P. Schaal. Compliant quadruped locomotion over rough terrain. Intelligent Robots and Systems. In *IROS 2009. IEEE/RSJ International Conference on*, pages 814–820, 2009.

[17] J. W. Cannon. The combinatorial structure of cocompact discrete hyperbolic groups. *Geom. Dedicata*, 16(2) :123–148, 1984.

[18] J. Ceccato. *Le tronc, de la locomotion à la commande*. PhD thesis, Université de Montpellier2, 2009.

[19] Z. Chen. *Java Card technology for Smart Cards : Architecture and Programmer's guide*. The Java Series. Addison-Wesley, 2000.

[20] B. Chevallier-Mames, D. Naccache, P. Paillier, and D. Pointcheval. How to Disembed a Program ? In M. Joye and J.-J. Quisquater, editors, *Cryptographic Hardware and Embedded Systems (CHES '04)*, volume 3156 of *Lecture Notes in Computer Science*, pages 441–454, Boston, Massachusetts, 2004. Springer.

[21] B.-K. Cho, S.-S. Park, and J.-H. Oh. Controllers for running in the humanoid robot, hubo. In *Humanoids*, pages 385–390, 2009.

[22] C. Choffrut, J. Karhumaki, and I. Petre. The Commutation of Finite Sets : A Chalenging Problem. *Theoretical Computer Science*, 273 :69–79, 2002.

[23] D. Christensen, U. Schultz, and K. Stoy. A distributed strategy for gait adaptation in modular robots. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 2765–2770. IEEE, 2010.

[24] S. Collins, A. Ruina, R. Tedrake, and M. Wisse. Efficient bipedal robots based on passive-dynamic walkers. *Science*, 307(5712) :1082–5, 2005.

[25] S. Collins, A. Ruina, R. Tedrake, and M. Wisse. Efficient bipedal robots based on passive-dynamic walkers. *Science*, 307(5712) :1082, 2005.

[26] J. Conway. *Regular Algebra and Finite Machines*. Chapman and Hall, 1971.

[27] B. Courcelle. The Monadic Second-order Logic of Graphs II : Infinite Graphs of Bounded Width. *Math. Syst. Theory*, 21 :187–221, 1989.

[28] P. Cousot and R. Cousot. Abstract interpretation : a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Proceedings of the 4th Annual ACM SIGPLAN-SIGACT Symp. on Principles of Programming Languages*, pages 238–252, Los Angeles, California, 1977.

[29] S. Dalibard, A. Nakhaei, F. Lamiraux, and J.-P. Laumond. Manipulation of documented objects by a walking humanoid robot. In *Humanoids*, pages 518–523. IEEE, 2010.

[30] K. Daltorio, A. Horchler, S. Gorb, R. Ritzmann, and R. Quinn. A small wall-walking robot with compliant, adhesive feet. In *Adhesive Feet, Intelligent Robots and Systems (IROS 2005), IEEE/RSJ International Conference on*, 2005.

[31] M. Dam and D. Gurov. Compositional Verification of CCS processes. In *Proceedings of PSI'99*, volume LNCS 1755, pages 247–256, 1999.

[32] J. de Vaucanson. *Le Mecanisme du fluteur automate*. Academie royale des sciences, 1738.

[33] V. Feipel, T. De Mesmaeker, P. Klein, and M. Rooze. Three-dimensional kinematics of the lumbar spine during treadmill walking at different speeds. *European Spine Journal*, 10(1) :16–22, 2001.

[34] J.-C. Filliâtre. Formal Verification of MIX Programs. In *Journées en l'honneur de Donald E. Knuth*, October 2007.

[35] Y. Fukuoka, H. Kimura, and A. Cohen. Adaptive dynamic walking of a quadruped robot on irregular terrain based on biological concepts. *The International Journal of Robotics Research*, 22(3-4) :187, 2003.

[36] J. A. Goguen and J. Meseguer. Unwinding and interference control. *IEEE Symp. on Security and Privacy*, 1984.

[37] D. Gouaillier, V. Hugel, P. Blazevic, C. Kilner, J. Monceaux, P. Lafourcade, B. Marnier, J. Serre, and B. Maisonnier. Mechatronic design of nao humanoid. In *Robotics and Automation (ICRA), 2009 IEEE International Conference on*, pages 769–774, 2009.

[38] L. Goubin and J. Patarin. DES and differential power analysis (the "duplication" method). In *CHES*, pages 158–172, 1999.

[39] A. Griggio. A Practical Approach to Satisfiability Modulo Linear Integer Arithmetic. *Journal on Satisfiability, Boolean Modeling and Computation (JSAT)*, 8 :1–27, January 2012.

[40] O. Grumberg and D. Long. Model Checking and Modular Verification. *ACM Trans. on Prog. Lang. & Syst.*, 16(3) :843–871, 1994.

[41] Y. Gurevich. Monadic Second-Order Theories. In J. Barwise and S. Feferman, editors, *Model Theoretic Logic*, pages 479–506. Springer, 1985.

[42] J. D. Hartog, J. Verschuren, E. de Vink, J. de Vos, and W. Wiersma. Pinpas : A tool for power analysis of smartcards. In *Security and privacy in the age of uncertainty : IFIP TC11 18th International Conference on Information Security (SEC2003)*, page 453. Springer, 2003.

[43] J.-C. Heudin. *Les creatures artificielles : des automates aux mondes virtuels*. Odile Jacob, 2008.

[44] K. Hirai. Current and future perspective of honda humanoid robot. In *IEEE/RSJ Int. Conference on Intelligent Robots and Systems*, pages 500–508, 1997.

[45] K. Hirai, M. Hirose, Y. Haikawa, and T. Takenaka. The development of honda humanoid robot. In *ICRA*, pages 1321–1326, 1998.

[46] F. Iida, J. Rummel, and A. Seyfarth. Bipedal walking and running with compliant legs. In *2007 IEEE International Conference on Robotics and Automation*, pages 3970–3975, 2007.

[47] A. Ijspeert, A. Crespi, D. Ryczko, and J. Cabelguen. From swimming to walking with a salamander robot driven by a spinal cord model. *Science*, 315(5817) :1416, 2007.

[48] Y. Ikemata, A. Sano, and H. Fujimoto. Generation and local stabilization of fixed point based on a stability mechanism of passive walking. In *ICRA*, pages 3218–3223. IEEE, 2007.

[49] Y. Ikemata, K. Yasuhara, A. Sano, and H. Fujimoto. A Study of the Leg-swing Motion of Passive Walking. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2009.

[50] H. Ishiguro. Studies on humanlike robots - humanoid, android and geminoid. In S. Carpin, I. Noda, E. Pagello, M. Reggiani, and O. von Stryk, editors, *Simulation, Modeling, and Programming for Autonomous Robots, First International Conference, SIMPAR 2008, Venice, Italy, November 3-6, 2008. Proceedings*, volume 5325 of *Lecture Notes in Computer Science*. Springer, 2008.

[51] S. Ito, H. Kawasaki, K. Moriki, and M. Sasaki. Robot Experiment of Torque Learning for Biped Balance with respect to Periodic External Force. In *Advanced Robotics, 2005. ICAR '05. Proceedings., 12th International Conference on*, pages 418–423, 2005.

[52] T. Jensen, D. Le Métayer, and T. Thorn. Verifying Security Properties of Control-Flow Graphs. In *Proc. of the 20th Symposium on Security and Privacy, Berkeley*, pages 89–103. IEEE Computer Society Press, 1999.

[53] D. R. Jones, M. Schonlau, and W. J. Welch. Efficient global optimization of expensive black-box functions. *J. of Global Optimization*, 13(4) :455–492, Dec. 1998.

[54] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa. Biped walking pattern generation by using preview control of zero-moment point. In *IEEE International Conference on Robotics and Automation*, volume 2, pages 1620–1626. Citeseer, 2003.

[55] S. Kajita, K. Kaneko, F. Kanehiro, K. Harada, M. Morisawa, S. Nakaoka, K. Miura, K. Fujiwara, E. S. Neo, and I. Hara. Cybernetic human HRP-4C : A humanoid robot with human-like proportions. In *ISRR*, pages 301–314, 2009.

[56] K. Kaneko, F. Kanehiro, M. Morisawa, T. Tsuji, K. Miura, S. Nakaoka, S. Kajita, and K. Yokoi. Hardware improvement of cybernetic human HRP-4C for entertainment use. In *IROS*, pages 4392–4399. IEEE, 2011.

[57] J. Karhumaki, M. Kunc, and A. Okhotin. Computing by Commuting. *Theoretical Computer Science*, 356(1-2) :200–211, 2006.

[58] J. Karhumaki, M. Latteux, and I. Petre. Commutation with Codes. *Theoretical Computer Science*, 340(1), 2005.

[59] J. Karhumaki, M. Latteux, and I. Petre. Commutation with Ternary Sets of Words. *Th. Comput. Syst.*, 38(2), 2005.

[60] J. Karhumaki and I. Petre. Conway's Problem for Three-Word Sets. *Theoretical Computer Science*, 289 :705–725, 2002.

[61] J. O. Kephart and W. C. Arnold. Automatic extraction of computer virus signatures. *4th Virus Bulletin International Conference. Virus Bulletin Ltd.*, pages 178–184, 1994.

[62] J. Kinder and H. Veith. Jakstab : A static analysis platform for binaries. In *Proceedings of 20th International Conference on Computer Aided Verification (CAV'2008)*, volume 5123 of *Lecture Notes in Computer Science*, pages 423–427, Princeton, NJ, USA, 2008. Springer.

[63] J. Kruskal. Well-Quasi-Ordering, the Tree Theorem and Vazsonyi's Conjecture. *Trans. Amer. Math. Soc.*, 95 :210–225, 1960.

[64] M. Kunc. On Language Inequalities $XK \subseteq LX$. In *Developments in Language Theory (DLT)*. Springer, 2005.

[65] M. Kunc. Regular Solutions of Language Inequalities and Well Quasi-Orders. *Theoretical Computer Science*, 348(2-3) :277–293, 2005.

[66] M. Kunc. Simple Language Equations. *Bulletin of the European Association for Theoretical Computer Science*, 85 :81–102, 2005.

[67] M. Kunc. The Power of Commuting with Finite Sets of Words. In *LNCS*, volume 2404, 2005.

[68] D. Lie, J. Mitchell, C. A. Thekkath, and M. Horowitz. Specifying and verifying hardware for tamper-resistant software. *IEEE Symp. on Security and Privacy*, pages 166 – 178, 2003.

[69] C. E. M., D. E. Long, and K. L. McMillan. Compositional Model Checking. In *Proc. of the 4th Symp. on Logic in Comp. Sci. (LICS'89*, pages 353–362, 1989.

[70] A. Mana, J. Lopez, J. J. Ortega, E. Pimentel, and J. M. Troya. A framework for secure execution of software. *Int. J. Inf. Secur.*, 3(2) :99–112, 2004.

[71] T. Mandersloot, M. Wisse, and C. G. Atkeson. Controlling Velocity in Bipedal Walking : A Dynamic Programming Approach. In *Proc. of IEEE International conference on Humanoid Robots*, 2006.

[72] O. Mangin and P.-Y. Oudeyer. Learning to recognize parallel combinations of human motion primitives with linguistic descriptions using non-negative matrix factorization. In *Conference on Intelligent Robots and Systems (IROS)*, 2012.

[73] H. Marques, M. Jäntsch, S. Wittmeier, C. Alessandro, A. Diamond, M. Lungarella, R. Knight, and O. Holland. ECCE1 : the first of a series of anthropomimetic musculoskeletal upper torsos., 2010.

[74] H. Marques, M. Jäntsch, S. Wittmeier, C. Alessandro, O. Holland, C. Alessandro, A. Diamond, M. Lungarella, and R. Knight. Semi-passive dynamic walking for humanoid robot using controllable spring stiffness on the ankle joint. In *Proceedings of Humanoids*, 2010.

[75] S. Matyas and J. Oseas. Code protection using cryptography. In *United States Patent, no. 4.757.534, 1988*, 1988.

[76] T. McGeer. Passive dynamic walking. *The Int. Journal of Robotics Research*, 9(2) :62, 1990.

[77] L. Mendona de Moura and N. Bjrner. Z3 : An efficient SMT solver. In *14th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'08)*, volume 4963 of *Lecture Notes in Computer Science*, pages 337–340, Budapest, Hungary, 2008. Springer.

[78] G. Mennitto and M. Buehler. CARL : A compliant articulated robot leg for dynamic locomotion. *Robotics and Autonomous Systems*, 18(3) :337–344, 1996.

[79] G. Metta, G. Sandini, D. Vernon, L. Natale, and F. Nori. The iCub humanoid robot : an open platform for research in embodied cognition. In *Proceedings of the 8th Workshop on Performance Metrics for Intelligent Systems*, pages 50–56. ACM, 2008.

[80] I. Mizuchi, T. Yoshikai, D. Sato, S. Yoshida, M. Inaba, and H. Inoue. Swing Motion by a Spined Whole-Body Tendon-Driven Humanoid" Kenta". *Nippon Robotto Gakkai Gakujutsu Koenkai Yokoshu (CD-ROM)*, 20 :1C38, 2002.

[81] I. Mizuuchi, M. Inaba, K. Nagashima, R. Tajima, T. Yoshikai, Y. Kuniyoshi, and H. Inoue. Design and control of a flexible spine for the whole-body humanoid'Kenta'. *Nippon Robotto Gakkai Gakujutsu Koenkai Yokoshu*, 19 :777–778, 2001.

[82] T. Mori, Y. Nakamura, M.-A. Sato, and S. Ishii. Reinforcement learning for a CPG-driven biped robot. In *Proceedings of the 19th national conference on Artifical intelligence*, AAAI'04, pages 623–630. AAAI Press, 2004.

[83] J. Morimoto, G. Cheng, C. G. Atkeson, and G. Zeglin. A simple reinforcement learning algorithm for biped walking. In *ICRA*, pages 3030–3035, 2004.

[84] K. Muecke and D. Hong. DARwIn evolution : development of a humanoid robot. In *IEEE International Conference on Intelligent Robotics and Systems*, 2007.

[85] K. S. Namjoshi and R. J. Trefler. On the completeness of compositional reasoning. In *Proc. of the 12th Int. Conference on Computer Aided Verification (CAV'00)*, pages 139–153. Springer-Verlag, 2000.

[86] K. Nishiwaki and S. Kagami. High frequency walking pattern generation based on preview control of ZMP. In *ICRA*, pages 2667–2672, 2006.

[87] K. Nishiwaki and S. Kagami. Simultaneous planning of com and zmp based on the preview control method for online walking control. In *Humanoids*, pages 745–751, 2011.

[88] A. Okhotin. Decision Problems for Language Equations. In *Internat. Colloq. on Automata, Languages and Programming (ICALP)*, pages 239–251. LNCS 2719, 2003.

[89] A. Okhotin. A Characterization of the Arithmetical Hierarchy by Language Equations. In *Descriptional Complexity of Formal Systems (DCFS)*, pages 225–237, 2004.

[90] A. Omer, R. Ghorbani, H. ok Lim, and A. Takanishi. Semi-passive dynamic walking for humanoid robot using controllable spring stiffness on the ankle joint. In *Int. Conf. on Autonomous Robots and Agents (ICARA 2009)*, pages 681–685, 2009.

[91] J. Or and A. Takanishi. From lamprey to humanoid : the design and control of a flexible spine belly dancing humanoid robot with inspiration from biology. *International Journal of Humanoid Robotics*, 2(1) :81, 2005.

[92] C. Paul, M. Lungarella, and F. Iida. Morphology, control and passive dynamics. *Robotics and Autonomous Systems*, 54(8) :617–618, 2006.

[93] S. Peter, S. Grimmer, S. Lipfert, and A. Seyfarth. Variable joint elasticities in running. *Autonome Mobile Systeme 2009*, pages 129–136, 2009.

[94] R. Pfeifer, M. Lungarella, and F. Iida. Self-organization, embodiment, and biologically inspired robotics. *Science*, 318(5853) :1088, 2007.

[95] M. Raibert, K. Blankespoor, G. Nelson, and R. Playter. Bigdog, the rough-terrain quadruped robot. In *17th World Congress, The Int. Federation of Automatic Control*, 2008.

[96] E. Rips and Z. Sela. Canonical representatives and equations in hyperbolic groups. *Invent. Math.*, 120 :489–512, 1995.

[97] P. Rodman and H. McHenry. Bioenergetics and the origin of hominid bipedalism. *American Journal of Physical Anthropology*, 52(1) :103–106, 1980.

[98] J. Rose and J. Gamble. *Human walking*. Lippincott Williams & Wilkins, 2006.

[99] G. Rozenberg. *Handbook of Graph Grammars and Computing by Graph Transformation*, volume 1. World Scientific, 1997.

[100] G. Sandini, G. Metta, and D. Vernon. *The iCub Cognitive Humanoid Robot : An Open-System Research Platform for Enactive Cognition*, pages 358–369. Springer-Verlag, Berlin, Heidelberg, 2007.

[101] U. Saranli, M. Buehler, and D. Koditschek. Design, modeling and preliminary control of a compliant hexapod robot. In *IEEE International Conference on Robotics and Automation*, volume 3, pages 2589–2596, 2000.

[102] U. Saranli, M. Buehler, and D. E. Koditschek. Rhex : A simple and highly mobile hexapod robot. *International Journal of Robotics Research*, 20 :616–631, 2001.

[103] U. Scarfogliero, C. Stefanini, and P. Dario. The use of compliant joints and elastic energy storage in bio-inspired legged robots. *Mechanism and Machine Theory*, 44(3) :580–590, 2009.

[104] I. Schaumueller-Bichl and E. Piller. A method of software protection based on the use of smart cards and cryptographic techniques. *Lecture Notes in Computer Science*, 209 :446 – 454, 1985.

[105] D. Scholz, M. Friedmann, and O. von Stryk. Fast, robust and versatile humanoid robot locomotion with minimal sensor input. In *Proc. 4th Workshop on Humanoid Soccer Robots at the 2009 IEEE-RAS Int. Conf. on Humanoid Robots*, Paris, 2009.

[106] M. Souissi, V. Hugel, and P. Blazevic. Influence of the number of humanoid vertebral column pitch joints in flexion movements. In *5th IEEE Int. Conf. on Automation, Robotics and Applications (ICARA'2011)*, pages 277–282, 2011.

[107] C. Sprenger, D. Gurov, and M. Huisman. Simulation Logic, Applets and Compositional Verification. Technical Report 4890, INRIA, 2003.

[108] B. J. Stephens and C. G. Atkeson. Dynamic balance force control for compliant humanoid robots. In *IROS*, pages 1248–1255, 2010.

[109] C. Stirling. A complete compositional modal proof system for a subset of CCS. *LNCS*, 194 :475–486, 1985.

[110] R. Tedrake, T. Zhang, M. Fong, and H. Seung. Actuating a simple 3D passive dynamic walker. In *IEEE International Conference on Robotics and Automation*, volume 5, pages 4656–4661, 2004.

[111] A. Thakur, J. Lim, A. Lal, A. Burton, E. Driscoll, M. Elder, T. Andersen, and T. Reps. Directed proof generation of machine code. In *Proceedings of the 22nd International Conference on Computer Aided Verification (CAV'10)*, volume 6174 of *LNCS*, pages 288–305. Springer, 2010.

[112] W. Thomas. Languages, automata, and logic. In *Rozenberg and Salomaa ed., Handbook of Formal Languages. Springer Verlag*, volume 3, pages 389–455, 1997.

[113] A. Thorstensson, H. Carlson, M. Zomlefer, and J. Nilsson. Lumbar back muscle activity in relation to trunk movements during locomotion in man. *Acta Physiologica Scandinavica*, 116(1) :13–20, 1982.

[114] M. Thorup. All structured programs have small tree-width and good register allocation. *Inf. and Comp.*, 142(2) :159–181, 1998.

[115] H. Ulbrich, T. Buschmann, and S. Lohmeier. Development of the humanoid robot lola, 2006.

[116] C. Vaughan. Theories of Bipedal Walking : An Odyssey. *Journal of biomechanics*, 36(4) :513–523, 2003.

[117] C. Walsh, K. Endo, and H. Herr. A quasi-passive leg exoskeleton for load-carrying augmentation. *International Journal of Humanoid Robotics*, 4(3) :487–506, 2007.

[118] P.-B. Wieber. Trajectory free linear model predictive control for stable walking in the presence of strong perturbations. In *Proceedings of Humanoids*, pages 137–142, 2006.