

# Contribution à l'étude des interfaces haptiques

## Le DigiHaptic : un périphérique haptique de bureau à degrés de liberté séparés

### THÈSE

présentée et soutenue publiquement le 1<sup>er</sup> octobre 2004

pour l'obtention du

Doctorat de l'Université des Sciences et Technologies de Lille

(spécialité instrumentation et analyses avancées)

par

Géry CASIEZ

#### Composition du jury

<i>Président :</i>	Pr. Philippe Mathieu
<i>Directeurs de thèse :</i>	Pr. Christophe Chaillou (LIFL – USTL) Pr. Betty Semail (L2EP – USTL)
<i>Rapporteurs :</i>	Pr. Sabine Coquillart Dr. Vincent Hayward Pr. Dominique Scapin
<i>Examineurs :</i>	Dr. Claude Andriot Dr. Patricia Plénacoste

Mis en page avec la classe thloria.

# Remerciements

Ce mémoire est l'aboutissement de trois années de travail menées au sein de l'équipe Graphix du LIFL et de l'équipe commande du L2EP. Je tiens à remercier l'ensemble des personnes qui ont contribué à la réalisation de ce travail.

Je tiens d'abord à remercier Christophe Chaillou de m'avoir donné la possibilité de réaliser cette thèse. Sa confiance, son dynamisme et son optimisme ont été des moteurs dans la bonne conduite de mes travaux.

Je remercie Betty Semail pour avoir co-encadré cette thèse ainsi que pour sa disponibilité et son écoute.

J'adresse également mes remerciements à Sabine Coquillart, Vincent Hayward et Dominique Scapin d'avoir gentiment accepté d'être rapporteurs de ce mémoire, Philippe Mathieu d'avoir accepté de présider le jury, Claude Andriot et Patricia Plénacoste pour avoir accepté de participer au jury.

Je remercie par ailleurs chaleureusement Thierry Flamen et Daniel Montignies pour leurs précieux conseils et leur grande disponibilité pour toute la réalisation de l'électronique, Thomas Dienne et Claude Fustin pour leur aide précieuse dans la conception et la réalisation des pièces mécaniques.

Je n'oublie pas tous les membres de la salle E101, Fred, Poulpe, François, Nikko et Gérald qui m'ont apporté à un moment ou un autre leurs compétences et qui ont partagé au jour le jour la progression de mes travaux. Merci aux membres de l'équipe Graphix, Philippe, Julien, Lol, Sam, Stéphane et Jérémy pour leur aide ponctuelle. Je remercie également l'ensemble des stagiaires que j'ai été amené à encadrer au cours de ma thèse, en particulier Sylvain.

Je tiens enfin à remercier tout particulièrement mes parents qui m'ont toujours encouragé tout au long de mes études et Sonia pour son soutien et ses encouragements quotidiens.

*A mon frère Maxence.*



# Table des matières

<b>Introduction</b>	<b>1</b>
---------------------	----------

<b>Chapitre 1</b>
<b>Etat de l'art</b>

1.1	Introduction . . . . .	3
1.2	Périphériques d'entrée . . . . .	4
1.2.1	Taxonomie des périphériques . . . . .	4
	Mouvements primitifs . . . . .	5
	Opérateurs de composition . . . . .	5
1.2.2	Degré de résistance . . . . .	6
	Interfaces isotoniques . . . . .	6
	Interfaces isométriques/élastiques . . . . .	8
1.2.3	Fonction de transfert . . . . .	9
1.2.4	Interfaces intégrales et séparables . . . . .	10
1.2.5	Périphériques directs et indirects . . . . .	11
1.2.6	Isomorphisme . . . . .	12
1.2.7	Control display . . . . .	13
1.3	Le retour haptique . . . . .	14
1.3.1	Définition du retour haptique . . . . .	14
1.3.2	Caractéristiques physiologiques . . . . .	15
	Physiologie du toucher . . . . .	15
	Physiologie du retour kinesthésique . . . . .	16
	Boucle sensori-motrice . . . . .	17
1.3.3	Types de retours haptiques . . . . .	18
1.3.4	Technologies d'actionneurs pour le retour d'effort . . . . .	18
1.3.5	Interfaces à retour d'effort . . . . .	19

Les gants . . . . .	19
Bras maîtres et stylos à retour de force . . . . .	20
Souris et manches à retour de force . . . . .	20
Systèmes à câbles . . . . .	22
1.3.6 Interfaces à retour tactile . . . . .	22
1.4 Intégration du retour haptique . . . . .	23
1.4.1 Détection de collision . . . . .	24
1.4.2 Calcul du retour de force . . . . .	25
Simulation de contacts rigides . . . . .	26
1.4.3 Architecture . . . . .	28
1.5 Applications du retour haptique . . . . .	29
1.6 Conclusion . . . . .	29

<b>Chapitre 2</b>
-------------------

<b>Conception, réalisation et commande du DigiHaptic</b>
--

2.1 Introduction . . . . .	31
2.2 Conception . . . . .	32
2.2.1 Limites des périphériques existants . . . . .	32
2.2.2 Choix de la séparation des ddl . . . . .	33
2.2.3 Périphériques à ddl séparés existants . . . . .	33
2.2.4 Affordances . . . . .	35
2.2.5 Fatigue . . . . .	36
2.3 Réalisation de la maquette . . . . .	37
2.3.1 Choix des doigts . . . . .	37
2.3.2 Ergonomie . . . . .	37
2.4 Réalisation du prototype fonctionnel . . . . .	38
2.4.1 Conception mécanique . . . . .	39
Etude du Paddle . . . . .	39
Rayon des manettes . . . . .	39
Force maximale . . . . .	40
Transmission du couple . . . . .	40
Choix des moteurs . . . . .	41
Choix définitif du rayon des manettes . . . . .	42
2.4.2 Conception électronique . . . . .	42
Boucle de contrôle sous QNX . . . . .	43

---

	API Windows/Linux . . . . .	43
	Solution DSP . . . . .	45
	La carte de puissance des moteurs . . . . .	45
	Les conditionneurs . . . . .	46
2.5	Identification des paramètres du système . . . . .	47
2.5.1	Détermination de l'inertie de la manette J . . . . .	48
2.5.2	Détermination des frottements secs et visqueux . . . . .	48
2.6	Commande du système . . . . .	50
2.6.1	Filtrage de la vitesse . . . . .	51
2.6.2	Commande en position . . . . .	52
2.6.3	Calcul des paramètres optimaux d'un mur virtuel . . . . .	53
	Calcul de la raideur et de l'amortissement virtuels . . . . .	53
	Calcul de la raideur physique . . . . .	55
2.6.4	Commande en impédance du système . . . . .	57
	Retour de force en mode isotonique . . . . .	57
	Retour de force en mode élastique . . . . .	57
	Simulation de murs en mode élastique . . . . .	58
	Simulation de vibrations . . . . .	60
2.7	Conclusion . . . . .	60

## Chapitre 3

### Interaction avec le DigiHaptic

3.1	Introduction . . . . .	63
3.2	Tâches . . . . .	64
3.2.1	Manipulation . . . . .	64
3.2.2	Navigation . . . . .	64
3.3	Interaction avec le DigiHaptic dans les tâches de manipulation . . . . .	66
3.3.1	Mode isotonique . . . . .	67
3.3.2	Mode élastique . . . . .	69
	Problème du frottement sec . . . . .	71
	Question du retour d'effort . . . . .	72
3.3.3	Passage entre mode isotonique et mode élastique . . . . .	72
3.3.4	Résumé des modes d'utilisation . . . . .	73
3.3.5	Combinaison mode isotonique et mode élastique . . . . .	73
	Métaphore du cube simple avec contrôle hybride isotonique élastique . . . . .	74

	Métaphore du cube avec mode hybride isotonique élastique . . . . .	76
	Métaphore de la sphère avec mode hybride isotonique élastique . . . . .	76
3.4	Interaction avec le DigiHaptic dans les tâches de navigation . . . . .	78
3.4.1	Métaphore de contrôle de la caméra . . . . .	78
3.4.2	Calcul du retour de force sans glissement sur la surface . . . . .	80
3.4.3	Glissement sur les surfaces . . . . .	80
3.5	Le retour d’effort en mode élastique . . . . .	80
3.5.1	Introduction . . . . .	80
3.5.2	Analyse . . . . .	81
3.5.3	Mode maître . . . . .	82
3.5.4	Mode esclave . . . . .	83
3.5.5	Comparaison des modes maître et esclave . . . . .	83
3.6	Conclusion . . . . .	84

<b>Chapitre 4</b>
-------------------

<b>Evaluation du DigiHaptic</b>
---------------------------------

4.1	Introduction . . . . .	87
4.2	Outils d’évaluation . . . . .	88
4.2.1	Loi de Fitts . . . . .	88
4.2.2	Bandes passantes . . . . .	90
4.2.3	La loi de suivi de trajectoires . . . . .	91
4.2.4	Coordination . . . . .	92
4.2.5	Apprentissage . . . . .	93
4.3	Présentation des évaluations . . . . .	93
4.4	Evaluation en suivi de trajectoires 3D . . . . .	94
4.4.1	Outils . . . . .	94
	Mesure de la coordination . . . . .	94
	Loi de suivi de trajectoire . . . . .	95
4.4.2	Méthode . . . . .	95
	Présentation de la tâche . . . . .	95
	Sujets . . . . .	96
	Procédure . . . . .	97
	Matériel . . . . .	97
4.4.3	Résultats et discussion . . . . .	97
	Temps de parcours du circuit . . . . .	97



---

Coordination . . . . .	99
Sorties . . . . .	100
Contacts . . . . .	100
4.4.4 Questionnaires . . . . .	101
4.4.5 Conclusion . . . . .	101
4.5 Evaluation en navigation avec retour d'effort . . . . .	102
4.5.1 Méthode . . . . .	102
Présentation de la tâche . . . . .	102
Sujets . . . . .	103
Procédure . . . . .	104
Matériel . . . . .	104
4.5.2 Résultats . . . . .	104
Temps de parcours . . . . .	104
Nombre de contacts . . . . .	106
Temps de contact . . . . .	106
Distance parcourue . . . . .	107
Résultats par zones . . . . .	107
Nombre de degrés de liberté utilisés . . . . .	107
Questionnaires . . . . .	109
4.5.3 Discussion et conclusion . . . . .	111
4.6 DigiHaptic version 2 . . . . .	112
4.6.1 Présentation de la V2 . . . . .	112
4.6.2 Autre design . . . . .	112
4.7 Conclusion . . . . .	113
<b>Conclusion</b>	<b>115</b>
<b>Annexes</b>	<b>119</b>
<b>Annexe A Quaternions</b>	<b>119</b>
A.1 Définition . . . . .	119
A.2 Conversion d'une matrice homogène de rotation en quaternion . . . . .	119
A.3 Conversion d'un quaternion en axe et angle de rotation . . . . .	120
A.4 Conversion d'un axe et angle de rotation en quaternion . . . . .	121
A.5 Multiplication de quaternions . . . . .	121

<b>Annexe B Les arbres BSP</b>	<b>123</b>
B.1 Principe des arbres BSP . . . . .	123
B.2 Création des arbres BSP . . . . .	123
B.3 Rendu de la scène . . . . .	125
B.4 Gestion des collisions avec les arbres BSP . . . . .	127
<b>Index</b>	<b>129</b>
<b>Bibliographie</b>	<b>131</b>

# Table des figures

1.1	Classification des périphériques suivant la nature des degrés de liberté (linéaire ou rotatif) et le type de grandeur mesurée : position, variation de position, force ou variation de force ; respectivement angle, variation d'angle, couple, variation de couple. La position du cercle dans la colonne indique la résolution du degré de liberté. . . . .	7
1.2	Flying Mouse de Logitech [Log] . . . . .	7
1.3	La Spaceball (à gauche) et la SpaceMouse (à droite) de 3dConnexion [3dC] . . . . .	8
1.4	Relation idéale entre la variation de la grandeur d'entrée du périphérique (colonne de gauche) et le résultat sur la variation de position dans l'environnement informatique suivant la fonction de transfert utilisée (d'après Zhai [Zha95]). $K$ est une constante. . .	9
1.5	Taxonomie des périphériques 6 ddls selon l'intégration de leurs ddls, leur degré de résistance et la fonction de transfert utilisée (adapté d'après [ZM93]). . . . .	12
1.6	Homunculus - Représentation des différentes parties du corps déformées selon la place qu'ils occupent dans les cortex sensitif (gauche) et moteur (droite) (d'après [Sag77]). .	15
1.7	Structure de la peau (adapté d'après [Gol99]) . . . . .	16
1.8	Schéma simplifié des boucles de contrôle d'un membre (adapté d'après [FCvdL04]) . .	17
1.9	Gants Rutgers et le CyberGrasp . . . . .	19
1.10	Le CyberForce . . . . .	20
1.11	Le Virtuouse6D, le DELTA Haptic Device et PHANToM6D. . . . .	21
1.12	Le PHANToM Desktop et le PHANToM Omni. . . . .	21
1.13	Le PenCat/Pro et la Wingman Force Feedback. . . . .	22
1.14	Le SPIDAR-8 et le SPIDAR-G. . . . .	22
1.15	Le STRESS de l'université de McGill. . . . .	23
1.16	Schéma d'architecture de rendu haptique - interaction entre l'homme et la machine (inspiré par [SB97]) . . . . .	24
1.17	Le proxy bouge localement pour minimiser la distance entre la position du doigt de l'utilisateur et les contraintes de l'environnement (adapté de [MHS01]). . . . .	26
1.18	Modélisation d'un mur par un ressort et un amortisseur en parallèle. . . . .	26
1.19	Illustration de la création d'énergie d'un ressort virtuel (d'après [CGSS93]). . . . .	27
2.1	Le Spidar et le Claw . . . . .	34
2.2	FPS game - Ici Counter Strike . . . . .	35
2.3	Classification de périphériques suivant la relation entre le degré de séparation de leurs degrés de liberté matériels et les degrés de liberté manipulés dans l'environnement virtuel (voir §1.2.1). . . . .	36

---

2.4	Correspondance entre le déplacement des doigts et celui des objets pour les translations et rotations en 3D . . . . .	38
2.5	Schéma général de commande du dispositif . . . . .	39
2.6	Le paddle et son électronique. . . . .	39
2.7	Modèle Catia d'un ensemble élémentaire du DigiHaptic . . . . .	41
2.8	Le DigiHaptic avec les trois manettes et sa coque. A gauche le modèle Catia et à droite le prototype fonctionnel . . . . .	41
2.9	Représentation des connexions entre les différentes parties matérielles. . . . .	43
2.10	Algorithme de la boucle de contrôle . . . . .	44
2.11	Classes de gestion du DigiHaptic . . . . .	44
2.12	Schema des amplificateurs de courant pour les moteurs . . . . .	45
2.13	Schéma du conditionneur pour un potentiomètre . . . . .	46
2.14	L'ensemble de l'électronique. . . . .	47
2.15	Modèle de base du système . . . . .	48
2.16	Position angulaire au cours du temps avec $K_\theta = 10.0$ pour un échelon indiciel quand $\theta_{ref}$ passe de $0,5\text{rad}$ à $0,17\text{rad}$ ( $10^\circ$ ). Le trait continu représente l'essai expérimental et le trait discontinu, la simulation sous Simulink. . . . .	49
2.17	Modèle du système avec prise en compte des frottements secs et visqueux. . . . .	49
2.18	Position angulaire au cours du temps avec $K_\theta = 10.0$ pour un échelon indiciel quand $\theta_{ref}$ passe de $0,5\text{rad}$ à $0,17\text{rad}$ ( $10^\circ$ ). Le trait continu représente l'essai expérimental et le trait discontinu, la simulation sous Simulink. . . . .	51
2.19	Vitesses angulaires filtrée et non filtrée . . . . .	52
2.20	Schéma blocs de la commande en position avec retour de vitesse. . . . .	52
2.21	Commande en position de la manette avec retour de vitesse avec $K_\theta = 25,7$ , $K_\Omega = 0,311$ pour un échelon indiciel quand $\theta_{ref}$ passe de $0,6\text{rad}$ à $0,17\text{rad}$ ( $10^\circ$ ). Le trait continu représente l'essai expérimental et le trait discontinu, la simulation sous Simulink. . . . .	54
2.22	Représentation d'un mur virtuel. . . . .	54
2.23	Schéma blocs de la simulation d'un mur virtuel. . . . .	55
2.24	Interface de paramétrage des manettes. . . . .	58
2.25	Schéma blocs de la simulation d'un mur virtuel avec simulation de raideurs sur les manettes. . . . .	59
3.1	Projection du vecteur force dans le repère cartésien. . . . .	68
3.2	Contrôle en position du cube avec retour de force sur les murs. . . . .	68
3.3	Contrôle en position du pointeur avec retour de force sous e-Touch. . . . .	69
3.4	Contrôle en position du pointeur avec retour de force sous Spore. . . . .	70
3.5	Rapport entre position angulaire et vitesse de déplacement pour deux fonctions. . . . .	70
3.6	Application de démonstration de manipulation d'objets en mode élastique (exemple d'objet manipulé). . . . .	71
3.7	Vitesse de déplacement sans prise en compte du frottement sec (gauche) et avec prise en compte (droite). . . . .	72
3.8	Représentation des possibilités de passage entre les modes isotoniques et élastique, contrôle des translations et rotations d'objets. . . . .	73
3.9	Résumé des modes d'utilisation du DigiHaptic (voir §1.2.1). . . . .	74
3.10	Courbe du couple sur la manette en fonction de sa position. . . . .	74

---

3.11	Métaphore du cube simple avec mode hybride isotonique élastique. . . . .	75
3.12	Métaphore du cube glissant avec mode hybride isotonique élastique. . . . .	76
3.13	Métaphore du cube collant avec mode hybride isotonique élastique. . . . .	77
3.14	A gauche, le pointeur manipulé par l'utilisateur est à l'intérieur de la sphère (normalement la sphère n'est pas représentée dans cette situation). Les manettes sont en mode isotonique avec contrôle en position du pointeur. A droite lorsque l'utilisateur touche les parois de la sphère isotonique, celle-ci apparaît progressivement et les manettes passent en mode élastique avec contrôle en vitesse de la sphère transparente. Le pointeur transparent à l'extérieur de la sphère, non représenté en pratique, indique ici la direction de déplacement. . . . .	78
3.15	Mouvements des doigts et mouvements associés de la caméra. L'annulaire sert soit à la translation de la caméra suivant la direction haut/bas, soit au contrôle du tangage. . .	79
3.16	Collision de la caméra avec un mur. . . . .	80
3.17	Capture écran de l'application de navigation. . . . .	81
4.1	Choix de la largeur de l'objet pour l'application de la loi de Fitts dans les tâches 2D (d'après [MB92]). . . . .	89
4.2	Exemple de suivi de trajectoire [AZ99]. . . . .	91
4.3	Illustration de la coordination de deux degrés de liberté (d'après [ZM98]). . . . .	92
4.4	Les trois chemins droits (haut) et les trois orientations des chemins (bas) . . . . .	95
4.5	Le pointeur en rouge à l'extérieur du chemin et son fantôme en gris à l'intérieur du chemin pendant un essai après que l'utilisateur soit sorti. . . . .	96
4.6	La couleur du pointeur dépend de sa position par rapport au centre du chemin . . . .	96
4.7	Temps moyen de parcours (ms) pour chaque chemin et périphérique . . . . .	99
4.8	Coefficient de coordination pour la SpaceMouse et le DigiHaptic suivant le chemin . .	99
4.9	Nombre moyen de sorties pour chaque chemin et périphérique . . . . .	100
4.10	Nombre moyen de contacts pour chaque chemin et périphérique . . . . .	100
4.11	Capture écran de l'intérieur du tunnel et de l'ensemble du circuit. . . . .	102
4.12	Degrés de liberté contrôlés par le DigiHaptic et la SpaceMouse. . . . .	103
4.13	Position des sujets en fonction de leur temps total de parcours et du nombre total de contacts. . . . .	105
4.14	Temps moyen de parcours du circuit (ms) pour l'ensemble des sujets, par tour de circuit, pour le DigiHaptic et la SpaceMouse. . . . .	105
4.15	Nombre moyen de contacts pour l'ensemble des sujets, par tour de circuit, pour le DigiHaptic et la SpaceMouse. . . . .	106
4.16	Pourcentage de temps suivant le nombre de degrés de liberté modifiés. . . . .	108
4.17	La version 2 du DigiHaptic avec et sans la coque. . . . .	112
4.18	Placement de la main sur le DigiHaptic. . . . .	113
B.1	Représentation simplifiée d'un arbre BSP. . . . .	124
B.2	Première étape de création de l'arbre BSP. . . . .	125
B.3	Deuxième étape de création de l'arbre BSP. . . . .	126
B.4	Algorithme de rendu arrière vers avant . . . . .	126
B.5	Affichage de l'arbre BSP selon la position et l'orientation du point de vue. . . . .	127
B.6	Algorithme de gestion des collisions . . . . .	128



# Liste des tableaux

1.1	Inventaire des grandeurs mesurables sur un périphérique d'entrée continu en fonction de la nature du degré de liberté. . . . .	5
2.1	Résumé des caractéristiques du moteur . . . . .	42
4.1	Bandes passantes des périphériques . . . . .	90
4.2	Bandes passantes des membres selon Balakrishnan et al. [BM97] et la norme ISO 9241-9. . . . .	91
4.3	Coefficients des courbes d'apprentissage de différents périphériques (d'après [CEB78]). . . . .	93
4.4	Tableau de correspondance entre l'ID des chemins et leurs représentation . . . . .	97
4.5	Temps moyen par tour . . . . .	105
4.6	Nombre de contacts moyen par tour . . . . .	106
4.7	Temps moyen de contact . . . . .	106
4.8	Distance moyenne parcourue. . . . .	107
4.9	Temps de parcours total (ms) par zone, par groupe et par périphérique. . . . .	107
4.10	Nombre de contacts totaux par zone, par groupe et par périphérique. . . . .	108
4.11	Résultats des questionnaires pour la SpaceMouse en pourcentages. . . . .	110
4.12	Résultats des questionnaires pour le DigiHaptic en pourcentages. . . . .	110





# Introduction

Le travail que nous présentons dans ce mémoire s'inscrit dans le cadre du projet Alcove (action et collaboration sur des objets virtuels complexes) de l'INRIA et du LIFL (USTL-CNRS). Cette thèse porte sur l'aspect matériel du projet avec le développement de nouveaux périphériques pour l'interaction en environnements 3D. Plus précisément, nous avons travaillé sur le développement d'un nouveau périphérique haptique pour l'interaction en environnements 3D.

L'ordinateur est une machine de calculs au service de son utilisateur : celui-ci entre ou modifie des données et reçoit le résultat des calculs. Ces échanges homme-machine se font par le biais des périphériques d'entrée et des périphériques de sortie (jusqu'au jour où un périphérique "ultime" agira directement sur les neurones sans passer par les intermédiaires physiologiques de l'utilisateur!).

Les périphériques d'entrée permettent de modifier l'état de la machine en introduisant de nouvelles données. Chaque périphérique d'entrée est dédié à une tâche bien précise : là où le clavier est utilisé pour saisir des caractères, la souris permet de manipuler des objets graphiques. Les périphériques d'entrée prennent ainsi une multitude de formes selon la nature des données à modifier.

Les périphériques de sortie transforment, quant à eux, les résultats du calcul de l'ordinateur en une information sensible et intelligible pour l'utilisateur. Ils sont la médiation entre les données contenues dans la machine et les sens de l'utilisateur : l'écran produit une sortie visuelle des données et les enceintes une sortie auditive.

Ces deux exemples concernent les sens de la vue et de l'ouïe qui toutes deux reçoivent des informations de nature physique. Le toucher, qui reçoit également des informations de nature physique, est à rapprocher de la vue et de l'ouïe (à la différence du goût et de l'odorat qui reçoivent des informations chimiques). Il se distingue néanmoins de la vue et de l'ouïe par son caractère bidirectionnel : par le toucher, nous recevons des informations de notre environnement mais nous agissons également sur lui. L'interaction qui passe par ce sens est donc difficile à mettre en oeuvre et nécessite un nouveau type de périphériques : les périphériques d'entrée-sortie.

Les études sur l'utilisation du toucher pour les périphériques ont été fortement encouragées par le développement des applications 3D. L'interaction avec un environnement 3D virtuel se trouve grandement enrichi par l'utilisation du sens du toucher puisqu'il joue un rôle important dans nos activités quotidiennes.

La recherche dans ce domaine a connu un essor rapide ces dix dernières années mais beaucoup de chemin reste à parcourir pour exploiter pleinement la modalité du toucher.

Dans ce contexte, nous avons commencé notre travail par l'étude des périphériques existants avec pour objectif de les classer de la manière la plus exhaustive possible. Cette première étape nous a aidé dans la recherche et la conception d'un nouveau périphérique. Nous avons alors eu l'idée de développer un périphérique basé sur la séparation des degrés de liberté et le retour d'effort : le DigiHaptic. Une partie importante du travail a ensuite concerné la réalisation et la commande du périphérique qui exigent des compétences multi-disciplinaires. Nous avons ensuite cherché les usages potentiels du périphérique et nous l'avons enfin évalué dans certains cas pour en mesurer les performances. La structure du mémoire reprend en grande partie la chronologie de notre travail.

Dans un premier chapitre, nous présenterons un état de l'art des périphériques d'entrée et du retour haptique. Nous commencerons par étudier les classifications existantes, du point de vue des propriétés physiques des périphériques. Nous étudierons ensuite les relations entre ces propriétés et les grandeurs informatiques contrôlées.

Nous verrons ensuite comment exploiter le sens du toucher pour utiliser les périphériques d'entrée en périphériques d'entrée-sortie en utilisant le retour haptique. Nous détaillerons alors comment intégrer le retour d'effort dans une application ainsi que les problèmes que cela pose.

Dans un second chapitre, nous présenterons, à partir des limites des périphériques existants, le concept de séparation des degrés de liberté. Nous montrerons les étapes de conception et de réalisation du DigiHaptic pour nous attarder sur les problèmes de commande pour le retour d'effort. Nous présenterons alors l'outil d'optimisation de la commande que nous avons développé.

Dans un troisième chapitre, nous étudierons les possibilités d'utilisation du périphérique dans des applications de manipulation et de navigation. Pour les tâches de manipulation, nous détaillerons tous les modes d'utilisation du périphérique ainsi que les moyens de passer d'un mode à l'autre. Nous présenterons alors des métaphores originales de contrôle hybride du DigiHaptic. Nous montrerons ensuite l'utilisation du DigiHaptic pour la navigation en environnement 3D avec retour d'effort lors des collisions de la caméra avec l'environnement. Nous continuerons sur l'étude de la réalisation du retour d'effort en mode élastique.

Dans un dernier chapitre, nous ferons un bref état de l'art des outils d'évaluation des périphériques puis nous présenterons les évaluations du DigiHaptic dans une tâche de suivi de trajectoire et de navigation en environnement 3D avec retour d'effort. Nous comparerons le DigiHaptic à la SpaceMouse afin d'observer l'influence de la séparation des degrés de liberté sur les performances de l'utilisateur.

# Chapitre 1

## Etat de l'art

### Sommaire

---

<b>1.1</b>	<b>Introduction</b>	<b>3</b>
<b>1.2</b>	<b>Périphériques d'entrée</b>	<b>4</b>
1.2.1	Taxonomie des périphériques	4
1.2.2	Degré de résistance	6
1.2.3	Fonction de transfert	9
1.2.4	Interfaces intégrales et séparables	10
1.2.5	Périphériques directs et indirects	11
1.2.6	Isomorphisme	12
1.2.7	Control display	13
<b>1.3</b>	<b>Le retour haptique</b>	<b>14</b>
1.3.1	Définition du retour haptique	14
1.3.2	Caractéristiques physiologiques	15
1.3.3	Types de retours haptiques	18
1.3.4	Technologies d'actionneurs pour le retour d'effort	18
1.3.5	Interfaces à retour d'effort	19
1.3.6	Interfaces à retour tactile	22
<b>1.4</b>	<b>Intégration du retour haptique</b>	<b>23</b>
1.4.1	Détection de collision	24
1.4.2	Calcul du retour de force	25
1.4.3	Architecture	28
<b>1.5</b>	<b>Applications du retour haptique</b>	<b>29</b>
<b>1.6</b>	<b>Conclusion</b>	<b>29</b>

---

## 1.1 Introduction

Ce chapitre présente dans un premier temps une classification générale des périphériques d'entrée. A partir de celle-ci, nous verrons qu'il est possible de regrouper les périphériques en familles.

Nous verrons ensuite comment créer une relation bi-directionnelle entre l'utilisateur et la machine par la stimulation du sens du toucher. Ce retour d'information, appelé *retour haptique*, est alors défini et les différents moyens de le produire sont présentés et illustrés à travers une présentation d'interfaces<sup>1</sup> existantes.

Enfin, nous étudierons les problèmes d'intégration du retour haptique dans une application.

## 1.2 Périphériques d'entrée

Dans cette section, nous présentons une taxonomie des périphériques basée sur la nature des grandeurs mesurables sur un périphérique d'entrée continu. De cette classification, nous distinguons alors les périphériques suivant le degré de résistance qu'ils offrent à l'utilisateur. Nous montrons ensuite que la fonction de transfert qui lie la grandeur mesurée sur le périphérique à la grandeur contrôlée dans l'environnement graphique est directement dépendante du degré de résistance du périphérique. Nous distinguons ensuite les interfaces intégrales et séparables ainsi que les périphériques directs et indirects pour approfondir la classification et nous terminons sur les différents critères qui influencent le caractère intuitif d'un périphérique.

### 1.2.1 Taxonomie des périphériques

Avant de développer de nouveaux périphériques, il est nécessaire d'en dresser une classification afin de comprendre la place des périphériques existants et les relations qui les unissent. C'est aussi un moyen de découvrir de potentiels périphériques à inventer.

Foley, Wallace et Chan [FWC84] classent les périphériques suivant les tâches graphiques qu'ils sont capables de réaliser (la tablette graphique permet, par exemple, de reconnaître un caractère).

Buxton a, quant à lui, proposé une classification des périphériques manipulés avec la main, basée sur leurs propriétés physiques et leur nombre de degrés de liberté<sup>2</sup> [Bux83]. Cette classification regroupe les périphériques suivant six critères : continu/discret (un périphérique est continu s'il peut prendre une multitude d'états et discret s'il prend un nombre réduit d'états, le plus souvent deux), le moyen d'action (main, pied, voix), la grandeur mesurée (position, angle, pression), le nombre de degrés de liberté (1,2 ou 3) et enfin le type d'interaction (suivant le type de saisie du périphérique). La classification de Buxton a l'inconvénient de ne prendre en compte que les périphériques à degrés de liberté continus. Plus récemment, Card, Mackinlay et Robertson ont amélioré la classification de Buxton [CMR90] [CMR91] en la généralisant à l'ensemble des périphériques continus et discrets<sup>3</sup>.

---

<sup>1</sup>Nous utilisons le terme interface comme synonyme de périphérique, sans nuance de signification.

<sup>2</sup>Un degré de liberté définit la possibilité de déplacement d'un objet dans l'espace en translation ou rotation selon les axes du repère associé à l'espace. Il y a six degrés de liberté possibles : trois en translation et trois en rotation.

<sup>3</sup>Bleser et Sibert ont par ailleurs conçu un logiciel permettant de choisir le périphérique approprié suivant la tâche d'interaction [BS90].

Les deux paragraphes qui suivent décrivent la méthode de classification de Card, Mackinlay et Robertson.

La communication homme-machine peut être modélisée par l'interaction dans un langage artificiel entre trois agents : un homme, une machine de communication et une application. Le langage comprend un vocabulaire de base et un ensemble d'opérateurs de composition de ce vocabulaire de base. Le vocabulaire de base est l'ensemble des mouvements primitifs réalisables avec un périphérique. Un mouvement primitif est un mouvement suivant un degré de liberté, linéaire ou rotatif. La composition de ces mouvements primitifs dans l'espace physique du périphérique, associée à la résolution du périphérique, va permettre de spécifier un ensemble d'éléments dans l'espace de l'application. La résolution correspond à la plus petite variation mesurable de la grandeur considérée.

### Mouvements primitifs

Le tableau 1.1 regroupe l'ensemble des grandeurs mesurables sur un périphérique d'entrée continu. Les degrés de liberté peuvent être linéaires ou rotatifs. Il est possible de mesurer soit la position ou la variation de position de ces degrés de liberté, soit la force ou la variation de force appliquée sur ceux-ci. Bien que d'autres moyens d'interaction soient envisageables (tel que l'utilisation de la voix), tous les périphériques d'entrée mécaniques utilisent une composition des grandeurs du tableau.

TAB. 1.1 – Inventaire des grandeurs mesurables sur un périphérique d'entrée continu en fonction de la nature du degré de liberté.

		degré de liberté	
		linéaire	rotatif
position	absolue	position P	angle R
	relative	mouvement dP	delta angle dR
force	absolue	force F	couple T
	relative	delta force dF	delta couple dT

Notons que les degrés de liberté sont ceux de l'effecteur et non ceux des capteurs utilisés pour mesurer ces derniers. Ainsi la souris a deux degrés de liberté linéaires même si l'acquisition des variations de position se fait par l'intermédiaire de la rotation d'une boule qui fait tourner des axes.

### Opérateurs de composition

Les opérateurs de composition décrivent les relations entre les différents degrés de liberté d'un périphérique.

Il y a trois opérateurs de composition : composition de fusion, composition d'agencement et composition de connexion<sup>4</sup>. La composition de fusion est la combinaison de deux périphériques tel que le domaine d'entrée est le produit vectoriel des domaines d'entrée

<sup>4</sup>Respectivement *merge composition*, *layout composition* et *connect composition* dans l'article [CMR91].

de chacun des périphériques. Ainsi la souris peut être considérée comme la composition orthogonale de deux curseurs linéaires ayant chacun un degré de liberté. La composition d'agencement correspond à la présence de plusieurs périphériques sur un même dispositif. Pour reprendre l'exemple de la souris, les trois boutons et le capteur de position XY sont quatre périphériques qui sont agencés en un seul. Enfin, la composition de connexion apparaît quand le domaine de sortie d'un périphérique correspond au domaine d'entrée d'un autre périphérique. Pour la souris, le domaine de sortie est connecté au domaine d'entrée du pointeur à l'écran. Ainsi, les périphériques n'ont pas besoin d'être physiques pour entrer dans la classification.

La figure 1.1 représente la classification des périphériques, basée sur les mouvements primitifs représentés au tableau 1.1. Un périphérique est représenté par un ensemble de cercles connectés entre eux. Chaque cercle représente la grandeur physique mesurée par un capteur. Chaque ligne représente l'opérateur de composition : ligne continue pour l'opérateur de fusion, ligne discontinue pour l'opérateur de d'agencement et double ligne pour l'opérateur de connexion. La souris représentée sur la figure 1.1 est l'agencement de quatre périphériques : le premier résulte de la fusion de deux périphériques qui mesurent les variations de position en X et Y (d'où le trait continu) et les trois autres périphériques sont de simples boutons (d'où le trait discontinu). La place des cercles dans chaque colonne indique la résolution du degré de liberté (voir la dernière ligne du tableau). Si le cercle est placé à gauche, le degré de liberté ne peut prendre que quelques états. Plus le cercle est placé à droite de la colonne plus le nombre d'états que peut prendre le périphérique est important. Les cercles pour les degrés de liberté en X et Y sont placés à droite de la colonne X (respectivement Y) pour indiquer que la résolution est quasiment continue. Le cercle pour les boutons est placé à gauche pour indiquer qu'ils sont discrets : ils ne peuvent prendre que deux états. Lors des tâches de pointage, la souris sert au contrôle de la position absolue du pointeur à l'écran, dans un plan orthogonal à celui de la souris (XZ)<sup>5</sup>.

### 1.2.2 Degré de résistance

Nous pouvons affiner la classification précédente (fig. 1.1) en prenant en compte les interfaces suivant le degré de résistance qu'elles offrent à l'utilisateur : isotonique (les deux premières lignes de la classification), isométrique ou élastique (les deux dernières lignes de la classification).

#### Interfaces isotoniques

D'après la définition du dictionnaire Larousse, l'adjectif isotonique désigne « une contraction musculaire telle que la force développée reste constante alors que la longueur du muscle diminue. » Ainsi, un périphérique isotonique doit avoir une résistance constante (le plus souvent nulle). Dans la classification précédente, cela correspond aux interfaces dont les positions ou variations de positions sont mesurées (voir [WF02] pour le détail des technologies). On peut citer en exemple les souris informatiques comme périphériques

---

<sup>5</sup>Card et al. contrôlent le pointeur dans un plan (XY) mais nous pensons qu'il est plus juste de tenir compte de l'orientation de l'écran.

	linéaire			rotatif			
	X	Y	Z	rX	rY	rZ	
P	○ pointeur écran		○				R
dP	○ souris	○	○ ③				dR
F							T
dF							dT
	1 10 100 Inf	1 10 100 Inf	1 10 100 Inf	1 10 100 Inf	1 10 100 Inf	1 10 100 Inf	

FIG. 1.1 – Classification des périphériques suivant la nature des degrés de liberté (linéaire ou rotatif) et le type de grandeur mesurée : position, variation de position, force ou variation de force ; respectivement angle, variation d'angle, couple, variation de couple. La position du cercle dans la colonne indique la résolution du degré de liberté.

2D ou les *souris volantes* comme périphérique 3D tel que la Flying mouse (fig. 1.2) de Logitech [Log] ou le Bat de Ware [WJ88].



FIG. 1.2 – Flying Mouse de Logitech [Log]

Il est possible de distinguer les périphériques isotoniques relatifs et les périphériques isotoniques absolus. La nature absolue ou relative d'un périphérique isotonique dépend de son architecture mécanique (avec ou sans chaîne cinématique) et du nombre de degrés de liberté qu'il possède (deux ou trois).

Dans le cas d'un périphérique isotonique à deux degrés de liberté, il est absolu s'il possède une chaîne cinématique et relatif sinon. Pour les périphériques isotoniques relatifs à deux degrés de liberté, les variations de position du périphérique dans l'espace sont mesurées. Il est possible d'utiliser la troisième dimension pour débrayer le périphérique en le soulevant et en le déplaçant ailleurs. Le débrayage a pour effet de translater l'origine du périphérique. Nous pouvons citer par exemple la souris informatique. Pour les périphériques isotoniques absolus à deux degrés de liberté, la position absolue du périphérique dans l'espace est mesurée et il n'est pas possible d'utiliser la troisième dimension pour débrayer. Le débrayage est ici possible explicitement par l'intermédiaire d'un bouton. Nous pouvons citer par exemple la Wingman Force Feedback de Logitech [Log].

Pour les périphériques à trois degrés de liberté, il n'existe pas de quatrième dimension permettant de débrayer. Ils sont par conséquent toujours absolus quelque soit leur architecture mécanique. C'est le cas de la Flying Mouse (sans chaîne cinématique) et du Phantom de Sensable [Sen] (avec une chaîne cinématique). Pour ces périphériques, il est nécessaire d'utiliser un bouton pour débrayer.

### Interfaces isométriques/élastiques

D'après la définition du dictionnaire Larousse, l'adjectif isométrique désigne « une contraction musculaire telle que la longueur du muscle ne change pas alors que la force développée par le muscle augmente. » Ainsi, un périphérique isométrique mesure des forces ou des couples mais ne bouge pas. Un exemple est la Spaceball [Spa] qui permet de mesurer les forces et les couples appliqués par l'utilisateur suivant six degrés de liberté (fig. 1.3) ou encore le Trackpoint d'IBM [IBM] pour contrôler le pointeur de la souris à l'écran.

Entre les périphériques isométriques qui ont une résistance infinie et les périphériques isotoniques qui n'ont pas de résistance ou une résistance constante, il existe les périphériques élastiques qui ont une résistance variable. Sur ces interfaces, la force de résistance augmente avec le déplacement de l'effecteur. Le périphérique contient en général des ressorts qui replacent l'effecteur dans une position neutre. Nous pouvons citer comme exemple les joysticks pour les applications 2D et la SpaceMouse pour les environnements 3D (fig. 1.3).



FIG. 1.3 – La Spaceball (à gauche) et la SpaceMouse (à droite) de 3dConnexion [3dC]

La différence la plus importante entre interfaces élastiques et isométriques est le retour kinesthésique. Selon la définition du dictionnaire Larousse, kinesthésique vient du grec *kinein*, se mouvoir et *aisthêsis*, sensation et correspond à la « sensibilité nerveuse consciente concernant les muscles, leur position, leur tension et leur mouvement ». Ainsi le déplacement qui existe sur les périphériques élastiques donne à l'utilisateur un retour proprioceptif<sup>6</sup> plus important que celui des interfaces isométriques. Zhai a montré [Zha95]

---

<sup>6</sup>Larousse : « Se dit de la sensibilité du système nerveux aux informations sur les postures et les mouvements, venant des muscles et des articulations. » Les trois domaines liés à la proprioception sont les sensibilités à la position dans l'espace, au mouvement du corps et aux forces exercées sur les muscles. Les deux premiers domaines correspondent au sens kinesthésique [FM03a].



qu'il n'y a pas de grandes différences du point de vue des performances<sup>7</sup> entre les deux catégories de périphériques mais que la facilité d'apprentissage est plus importante pour les périphériques élastiques.

Nous avons pour l'instant abordé l'aspect matériel des dispositifs et nous venons de voir qu'ils peuvent avoir différents degrés de résistance. Nous allons désormais nous intéresser aux liens entre les grandeurs mesurées physiquement et les grandeurs contrôlées dans l'environnement informatique.

### 1.2.3 Fonction de transfert

La fonction de transfert<sup>8</sup> décrit la relation mathématique qui existe entre la position dans l'environnement graphique et la position ou la force mesurée sur le périphérique. Par exemple, une fonction de transfert d'ordre 0 fait correspondre directement, à un facteur près, la position absolue dans l'environnement informatique à la position ou force mesurée sur le périphérique. On parle dans ce cas de *contrôle de position*. Dans le cas d'une fonction de transfert d'ordre 1, la vitesse dans l'environnement informatique est proportionnelle à la position ou force mesurée sur le périphérique. Dans ce cas, on parle de *contrôle en vitesse*. La figure 1.4 schématise les contrôles en position et en vitesse.

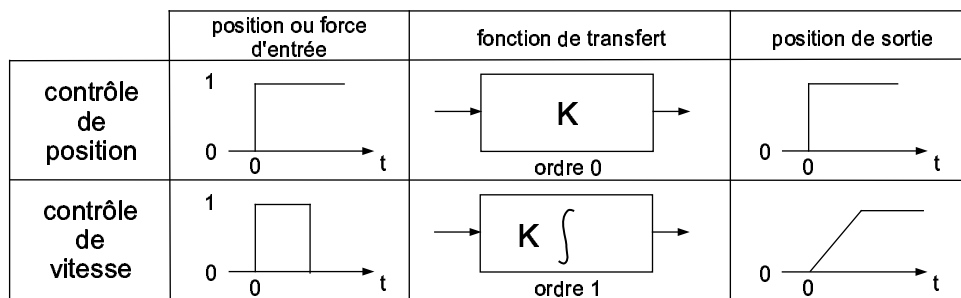


FIG. 1.4 – Relation idéale entre la variation de la grandeur d'entrée du périphérique (colonne de gauche) et le résultat sur la variation de position dans l'environnement informatique suivant la fonction de transfert utilisée (d'après Zhai [Zha95]).  $K$  est une constante.

Les périphériques isotoniques, isométriques et élastiques peuvent aussi bien être utilisés pour le contrôle en position que pour le contrôle en vitesse. Par exemple la souris est une interface isotonique qui sert habituellement à faire du contrôle de position. Ce périphérique peut être aussi utilisé pour réaliser un contrôle en vitesse de la position du pointeur. Dans ce cas, on définit une position de référence de la souris pour laquelle la vitesse de déplacement du pointeur est nulle. Ensuite, plus la souris est écartée de sa position neutre, plus la vitesse de déplacement du pointeur est importante dans la direction correspondante à celle de la souris.

<sup>7</sup>Pour les deux tâches expérimentées : une tâche de superposition d'objets et une autre de poursuite d'objet.

<sup>8</sup>Il faut comprendre la fonction de transfert au sens automatique : c'est la relation qui lie la sortie à l'entrée du système. L'ordre de la fonction de transfert est défini par le degré du polynôme au dénominateur.

Wickens et Poulton ont montré que les contrôles en position et vitesse sont tous deux préférables aux fonctions de transfert d'ordre supérieur [Wic92][Pou74]. Les expériences de Massimino et al. confirment ces résultats [MSR89]. L'accélération, par exemple, qui correspond à une fonction de transfert d'ordre deux, est en effet une grandeur difficilement contrôlable.

Le contrôle en position reproduit dans l'environnement informatique tous les mouvements de la main de l'utilisateur, volontaires ou involontaires. Le contrôle en vitesse a, quant à lui, l'avantage d'introduire un filtre passe-bas caractéristique de la fonction d'intégration, qui a pour effet de supprimer tous les bruits hautes fréquences associés aux mouvements involontaires. Cela permet à l'utilisateur de réaliser des trajectoires plus régulières lors du contrôle de la vitesse de l'objet manipulé. Enfin, avec le contrôle en position, l'espace de travail est limité (à moins de débrayer le périphérique), alors que l'espace de travail est illimité avec les périphériques isométriques.

Zhai et al. ont montré qu'il existe un lien entre les degrés de résistance des périphériques, les fonctions de transfert et les performances des utilisateurs [Zha95][ZM93]. Dans leur expérience, qui consiste à venir superposer deux tétraèdres avec une souris volante ou une Spaceball, ils ont en effet obtenu de meilleurs résultats pour le contrôle en position avec le périphérique isotonique et de meilleurs résultats pour le contrôle en vitesse avec le périphérique isométrique. Ils ont aussi montré que le contrôle en position avec les périphériques isotoniques est plus intuitif que le contrôle en vitesse avec les périphériques isométriques, mais que l'écart diminue avec l'entraînement. Par ailleurs, le contrôle en position avec les périphériques isotoniques est plus fatigant que le contrôle en vitesse avec les périphériques isométriques. Enfin, les trajectoires réalisées avec les périphériques isométriques avec contrôle en vitesse sont plus régulières.

Ces chercheurs ont en outre montré, dans une seconde expérimentation sur une tâche de manipulation, que les interfaces isotoniques donnent lieu à des temps de réalisation plus courts qu'une interface isométrique mais que cette dernière permettait des mouvements plus coordonnés [ZM98] (voir chapitre suivant).

Nous avons vu dans le §1.2.1 qu'un périphérique peut résulter de l'agencement de plusieurs périphériques. Chaque degré de liberté peut alors être contrôlé indépendamment des autres. Parallèlement, il est possible de considérer le degré de séparation d'une tâche. Nous allons maintenant analyser le lien entre le degré de séparation de la tâche à effectuer et le niveau d'intégration des degrés de liberté du périphérique.

#### 1.2.4 Interfaces intégrales et séparables

Un objet est défini entre autre par ses attributs. Par exemple, un cercle rouge a une taille, une couleur, une forme et une position. Ces attributs définissent un espace de perception. Garner a observé que les relations entre les attributs d'un objet peuvent être perçues de deux façons différentes suivant la facilité d'identification de chaque attribut [Gar74]. Certains attributs, dont les valeurs se combinent pour former une perception unique dans l'esprit de l'observateur, ont une relation intégrale entre eux. D'autres attributs ont une relation séparable, c'est-à-dire que l'observateur ne les agrège pas et perçoit l'objet comme un ensemble d'attributs. Par exemple, les attributs de position horizontale

et verticale d'un point sont perçus comme intégraux alors que la couleur et la forme sont perçus comme séparables. Les attributs intégraux sont en général de natures homogènes alors que les attributs séparables sont en général de natures hétérogènes. Nous avons utilisé le terme séparable, et non séparé, car les attributs sont physiquement intégrés et qu'ils peuvent être séparés dans l'esprit de l'observateur. Une tâche intégrale est alors une tâche dans laquelle l'utilisateur va devoir modifier des attributs intégraux d'un objet et une tâche séparable consiste à modifier des attributs séparables d'un objet.

Jacob et al. définissent les interfaces intégrales et séparables de manière similaire [JS92][JSMM94]. Selon leur définition, un périphérique ayant plus d'un degré de liberté peut être considéré comme intégral ou séparable suivant qu'il est naturel (ou possible) de se déplacer « diagonalement » suivant les dimensions de l'espace : avec un périphérique intégral, le mouvement s'effectue dans l'espace euclidien et modifie toutes les dimensions de contrôle à la fois. Par exemple, le chemin réalisé pour aller d'un point à un autre de l'espace est la ligne droite. A l'opposé, un périphérique séparable produit un mouvement en marches d'escalier : le mouvement s'effectue suivant une seule dimension à chaque instant. Ainsi un périphérique séparable résulte de la composition de plusieurs périphériques et un périphérique intégral résulte de la fusion de plusieurs périphériques (voir §1.2.1).

Ces chercheurs ont montré dans une expérimentation [JSMM94] que les interfaces intégrales sont mieux adaptées aux tâches intégrales et les interfaces séparables aux tâches séparables. Ils ont pour cela utilisé une souris volante comme périphérique intégral et une souris classique comme périphérique séparable. Cette dernière possède deux degrés de liberté intégraux. Le troisième est obtenu en appuyant sur un bouton de la souris tout en la déplaçant suivant une direction. Il est donc possible d'utiliser soit les deux degrés de liberté intégraux à la fois, soit le troisième degré de liberté.

La figure 1.5 représente un schéma synthétique des différentes notions dont nous avons parlé précédemment [ZM93]. Le degré de résistance est représenté sur l'axe vertical. Entre les périphériques isotoniques et les périphériques isométriques se situent les périphériques élastiques. En abscisse est représentée la fonction de transfert. Nous avons vu que les périphériques isotoniques sont mieux adaptés pour le contrôle en position et les périphériques isométriques pour le contrôle en vitesse. Cela a été vérifié pour les interfaces intégrales mais n'a pas encore été vérifiée pour les interfaces séparables, bien qu'il n'y ait pas de raison que le résultat soit différent. En profondeur est représenté le degré d'intégration des périphériques. Près de l'origine sont positionnés les périphériques complètement intégrés, où tous les degrés de liberté sont sur le même effecteur. De l'autre côté, se situent les périphériques séparés où tous les degrés de liberté sont séparés. Entre les deux, nous pouvons placer par exemple les périphériques qui sont séparés en deux périphériques qui ont chacun trois degrés de liberté (l'un sert par exemple aux translations et l'autre aux rotations).

### 1.2.5 Périphériques directs et indirects

Cette notion permet de décrire la relation qui existe entre la propriété mesurée et la propriété contrôlée. Dans une tâche de pointage par exemple, la propriété contrôlée est la position du pointeur à l'écran. Dans cette situation, un périphérique est direct si la position

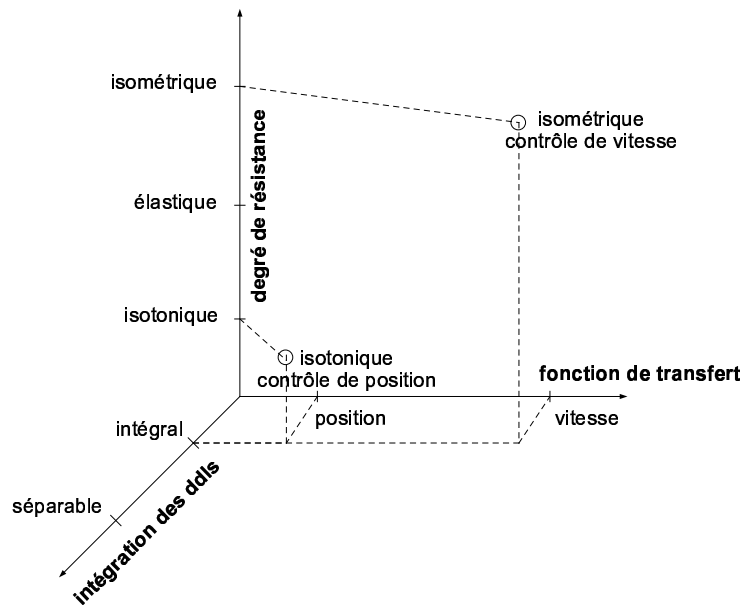


FIG. 1.5 – Taxonomie des périphériques 6 ddl selon l'intégration de leurs ddl, leur degré de résistance et la fonction de transfert utilisée (adapté d'après [ZM93]).

à l'écran peut être indiquée directement, sans intermédiaire. Les crayons optiques ou les écrans tactiles sont des exemples de périphériques directs. Les périphériques indirects sont physiquement situés à l'écart de l'écran et nécessitent un intermédiaire pour indiquer la position. Les souris ou les trackballs sont des exemples de périphériques indirects car ils utilisent un pointeur pour désigner une position à l'écran [Gri99].

### 1.2.6 Isomorphisme

Zhai s'est également intéressé aux facteurs qui influent sur le caractère intuitif d'une interface [ZM98]. Plus la relation mathématique qui lie l'espace physique et l'espace virtuel est compliquée, moins l'interaction est isomorphique. Il recense les trois facteurs suivants qui influent l'isomorphisme : la fonction de transfert, la position et l'orientation.

En étendant la classification de Zhai, nous définissons au total cinq paramètres qui vont influencer sur le degré d'isomorphismes de l'interface :

- fonction de transfert
- position
- orientation
- direction
- déplacement

Les fonctions de transfert d'ordre supérieur à zéro, qui nécessitent une ou plusieurs intégrations, sont moins isomorphiques et nécessitent un apprentissage par rapport aux fonctions de transfert d'ordre zéro pour lesquelles une simple multiplication fait correspondre les positions des espaces physiques et virtuels. Par ailleurs, un coefficient multiplicatif (ou sensibilité) de valeur égale à un correspondra à une reproduction exacte de

la position de l'espace physique dans l'espace virtuel. Les coefficients inférieurs ou supérieurs à un sont par conséquent moins isomorphiques. Pour le contrôle en position, les périphériques isotoniques absolus sont plus isomorphiques que les périphériques relatifs qui nécessitent un débrayage de la part de l'utilisateur.

Un autre facteur influençant le caractère intuitif d'une interface est le décalage ou *offset* de position entre les espaces physiques et virtuels, qui rejoint la notion de périphériques directs et indirects. Plus cet écart est faible, plus le périphérique est isomorphique. Les écrans tactiles et les tablettes graphiques sont deux périphériques isotoniques absolus mais ces derniers sont situés à une quarantaine de centimètres de l'écran alors que les premiers agissent directement sur l'écran. L'importance de ce facteur a été démontrée par Paljic pour un Plan de Travail Virtuel [Pal04].

L'orientation entre les espaces physiques et virtuels intervient également. Pour reprendre les exemples précédents, la souris a son espace de travail orienté à  $90^\circ$  par rapport à l'écran alors qu'il n'y a pas de décalage d'orientation pour l'écran tactile.

La direction de déplacement influence également l'isomorphisme. Si le pointeur de la souris se déplace vers la gauche quand la souris est déplacée vers la droite, la transformation sera moins isomorphique.

Enfin, nous pouvons définir l'isomorphisme de déplacement qui fait correspondre les mouvements des degrés de liberté physiques et virtuels. L'utilisation d'un degré de liberté rotatif pour effectuer un déplacement linéaire est moins isomorphique que l'utilisation d'un degré de liberté linéaire pour effectuer un déplacement linéaire.

Nous avons vu que la fonction de transfert joue un rôle important sur l'isomorphisme. Si, dans le cas d'un contrôle en position, une sensibilité égale à un permet d'obtenir un isomorphisme parfait, une modification de cette valeur permet généralement d'améliorer les performances de l'utilisateur suivant le type de tâche. Pour cette raison, nous allons étudier les influences de la sensibilité sur les performances de l'utilisateur.

### 1.2.7 Control display

Nous venons d'utiliser le terme de sensibilité que nous avons traduit de l'anglais *control display*. Nous pensons cependant que le terme anglais est mieux adapté pour traduire la relation qui lie le déplacement du périphérique (control) au déplacement à l'écran (display). Pour cette raison, nous préférons garder le terme anglais. Le control display correspond au rapport entre le déplacement dans l'environnement informatique et le déplacement ou la force appliquée au périphérique. Dans le cas de la souris, cela correspond au rapport entre la distance parcourue avec le pointeur à l'écran et la distance parcourue avec la souris. Dans le cas d'un joystick, cela correspond au rapport entre la vitesse de déplacement du pointeur en mètres par seconde et la force appliquée au joystick en newtons. Il est également possible de définir des relations non-linéaires de manière à tenir compte de la vitesse de déplacement du périphérique [JC90]. Des contrôles non-linéaires ont également été proposés pour les souris isométriques [RS90][BSRO95].

Le control display influence les performances de l'utilisateur. S'il est trop faible, l'utilisateur va perdre en rapidité et s'il est trop important, il va perdre en précision. Pour évaluer cette influence, il est possible de tracer la courbe du temps de réalisation d'une

tâche en fonction du control display. Certains chercheurs ont obtenu, pour des tâches de pointage, une courbe en forme de U [AG90][Zha95], d'autres une relation linéaire [Gib62] et d'autres ont montré qu'il n'y a pas de relation [JC90]. Plus précisément, Gibbs a étudié l'influence du control-display et des retards d'affichage pour des tâches de sélection avec contrôle en position et contrôle en vitesse [Gib62][AZ01]. L'équation qu'il a obtenu (eq. 1.1) montre que si le retard ( $L$ ) est supérieur à zéro, alors la courbe obtenue a une forme en U, sinon c'est une ligne droite.  $\frac{1}{G}$  représente le control display.

$$T = 0.91 + 1.212 L - 0.4 L^2 - 0.02 \frac{1}{G} - 0.106 \frac{L}{G} + 0.032 \frac{L}{G^2} - 0.003 \frac{L^2}{G} \quad (1.1)$$

Nous pouvons tout de même remarquer que la forme en U est au moins valable pour les valeurs extrêmes de control display, quelque soit le retard. Si le contrôle display est trop important, il sera en dehors des limites des membres humains et s'il est trop faible, il sera en dehors de la précision des membres. Accot et al. ont montré que la forme en U est également valable pour des control-display modérés pour les tâches de suivi de trajectoires [AZ01].

## 1.3 Le retour haptique

Nous venons de définir différents critères de classification pour les périphériques d'entrée. Comme leur nom l'indique, l'information est unidirectionnelle : elle va de l'utilisateur à l'ordinateur. Le retour d'information, de la machine à l'utilisateur, s'effectue généralement par le biais de l'écran pour la vue et des enceintes pour l'ouïe. Nous allons voir qu'il est possible de créer un autre échange d'information bi-directionnel pour les périphériques d'entrée afin d'exploiter le sens du toucher. Dans ce cas, on ne parle plus de périphériques d'entrée mais d'interfaces haptiques.

Après une définition du retour haptique, nous verrons quelles sont les caractéristiques physiologiques du toucher. Nous verrons ensuite les différents types de retour haptiques qui permettent de stimuler le sens du toucher, pour nous attarder sur le retour haptique actif qui est notre sujet d'étude. Nous verrons alors rapidement les technologies d'actionneurs permettant ce retour et leur mise en œuvre dans différentes interfaces à retour d'effort. Enfin, nous parlerons rapidement des interfaces à retour tactile.

### 1.3.1 Définition du retour haptique

Le terme haptique vient du mot grec *haptein* qui signifie toucher. Il est défini dans le Larousse par ce « qui concerne la sensibilité cutanée » et l'« étude scientifique du toucher ». Dans l'usage, le terme haptique est utilisé pour parler de deux types de retour sensoriels : le *retour d'effort* ou *retour kinesthésique* et le *retour tactile*. Le premier est relatif à la perception des forces de contact, de dureté, de poids et d'inertie d'un objet. Ce type de retour qui contraint les mouvements, sollicite les muscles, tendons et articulations. Le *retour tactile*, quant à lui, concerne la perception des états de surface (rugosité, texture), de température, des glissements, de détection des arêtes.

### 1.3.2 Caractéristiques physiologiques

Contrairement aux autres sens qui sont très localisés, le toucher concerne toute la surface extérieure du corps humain avec plus ou moins de sensibilité. D'après la figure 1.6, nous pouvons remarquer que des parties du corps comme l'épaule sont très peu représentées dans les cortex sensitifs et moteurs alors que la main et en particulier les doigts sont largement présents en raison du nombre important de capteurs sensitifs qu'ils possèdent. Pour ces raisons et pour des raisons pragmatiques d'utilisation, les interfaces haptiques vont généralement être contrôlées par les mains et les doigts. Nous allons nous intéresser plus en détails aux récepteurs présents sous la peau des mains et des doigts.

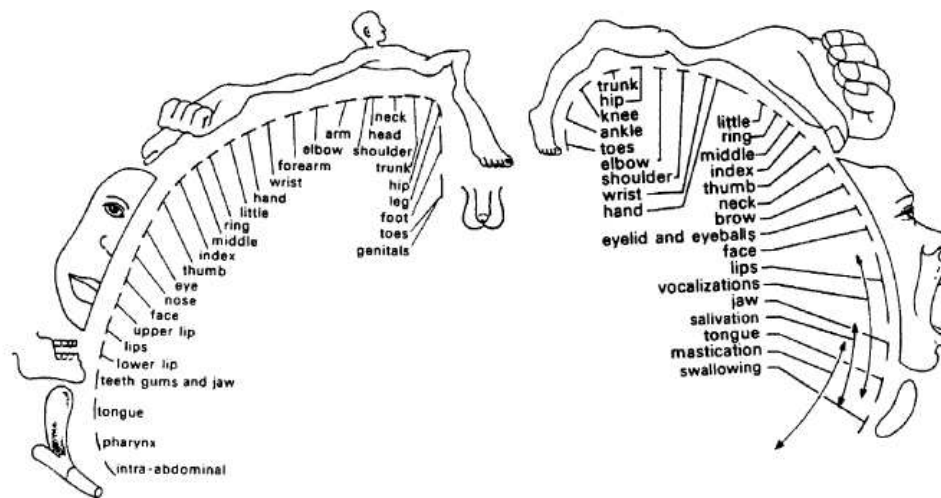


FIG. 1.6 – Homunculus - Représentation des différentes parties du corps déformées selon la place qu'ils occupent dans les cortex sensitif (gauche) et moteur (droite) (d'après [Sag77]).

#### Physiologie du toucher

Sous la peau se trouvent divers capteurs en concentration plus ou moins grande. Il existe cinq types de capteurs spécialisés suivant le type de stimulus : terminaisons nerveuses, corpuscules de Meissner, disques de Merkel, corpuscules de Pacini et corpuscules de Ruffini (fig. 1.7). Les terminaisons nerveuses, proches de la surface de la peau, sont sensibles à la douleur (nocicepteurs). Les autres récepteurs réagissent aux excitations mécaniques (mécanorécepteurs). Parmi ceux-ci, les corpuscules de Meissner, situés juste en dessous de l'épiderme, représentent environ 40% des récepteurs tactiles de la main. Comme ils bougent avec le derme de la peau, ceux-ci sont bien adaptés pour détecter les mouvements à travers la peau (discontinuités de surface, géométrie de surface) et fonctionnent comme capteurs de vitesse. Les disques de Merkel représentent 25% des récepteurs de la main et sont principalement utilisés comme capteurs de pression mais peuvent également être sensibles aux vibrations. Les corpuscules de Pacini, représentant 13% des récepteurs de la main, sont situés plus profondément dans la peau. Ils fonctionnent comme détecteurs d'accélération et sont également sensibles aux vibrations. Finalement les corpuscules de

Ruffini, sensibles à l'intensité et à la direction de forces statiques, tout comme aux déformations de la peau et la chaleur (thermorécepteurs) représentent 19% des récepteurs de la main [Bur96].

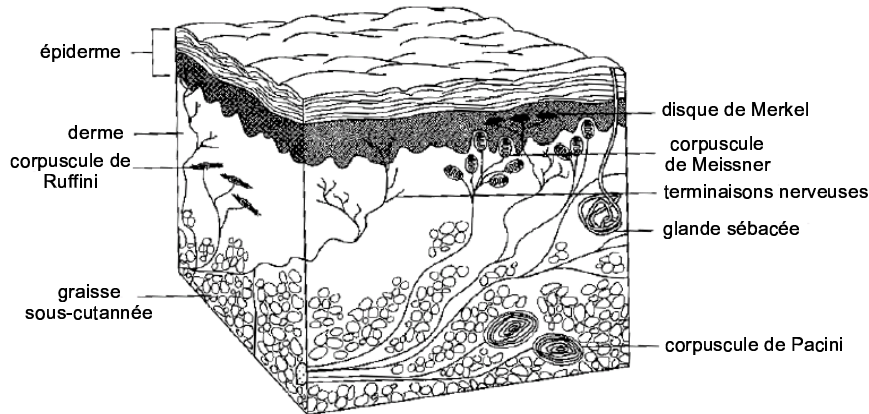


FIG. 1.7 – Structure de la peau (adapté d'après [Gol99])

Les corpuscules de Meissner et de Pacini sont à adaptation rapide avec une bande passante située entre 20 et 50 Hz pour les premiers, 100 et 300 Hz pour les seconds. Les disques de Merkel et les corpuscules de Ruffini sont à adaptation lente avec une bande passante située entre 0 et 10 Hz.

La résolution spatiale des récepteurs joue également sur la sensibilité du toucher. Des expériences ont ainsi montré que le seuil minimal de discrimination (JND<sup>9</sup>) entre deux points au niveau du doigt est de 0,7 à 1,0 mm.

### Physiologie du retour kinesthésique

Les récepteurs physiologiques sensibles au retour kinesthésique sont situés au niveau des tendons d'attache des muscles aux os, au niveau des articulations et dans les muscles. Les organes de Golgi sont ainsi situés entre les muscles et les tendons correspondants. Ils mesurent les forces et règlent la contraction des muscles. Des récepteurs sont aussi présents entre les fibres musculaires dans tout le muscle. Ceux-ci mesurent l'étirement entre les fibres musculaires et mesurent de cette façon le taux d'élongation du muscle. La fatigue du muscle intervient également dans la perception de la force. Les récepteurs de la peau participent par ailleurs à la perception du retour kinesthésique [Bur96]. La bande passante de ces capteurs est située entre 20 et 30 Hz.

Nous pouvons par ailleurs distinguer le *retour extéroceptif* du *retour proprioceptif*. Le premier concerne la mesure et la perception des phénomènes extérieurs au corps humain (chocs, poids et raideur d'un objet) alors que le second concerne les phénomènes internes à son propre corps (position des membres, verticalité du corps) [Léc01].

Ainsi, en ce qui concerne les différentes parties du corps, le JND en position du poignet est de 2,0° et celui du doigt 2,5°. Le JND sur la raideur d'un objet est de 22%, celui sur

<sup>9</sup>Just Noticeable Difference : variation minimale d'intensité que l'on perçoit pour un stimulus donné [Léc01].



le poids de 10% et celui en force de contact entre 5% et 15% [Bur96]. La force maximale contrôlable par les doigts varie entre 50 et 100N. Cette force dépend des muscles utilisés (muscles des doigts uniquement ou muscles du bras également) [TSEC94] mais la gamme de force typique lors d'opérations d'exploration et de manipulation se situe entre 5 et 15 N. Enfin, la résolution de contrôle d'une force est d'environ 0,04 N [SB97].

### Boucle sensori-motrice

Le principe de contrôle d'un membre humain est basé sur deux boucles : une boucle rapide dépendant des propriétés mécaniques du membre et une boucle lente de contrôle nerveux dont la bande passante est comprise entre 1 et 10 Hz (fig. 1.8). Celle-ci dépend de la tâche en train d'être réalisée : 1 à 2 Hz pour des signaux déstabilisants, 2 à 5 Hz pour des signaux périodiques, jusqu'à 5 Hz pour des trajectoires apprises et environ 10 Hz pour des actions réflexes [SB97].

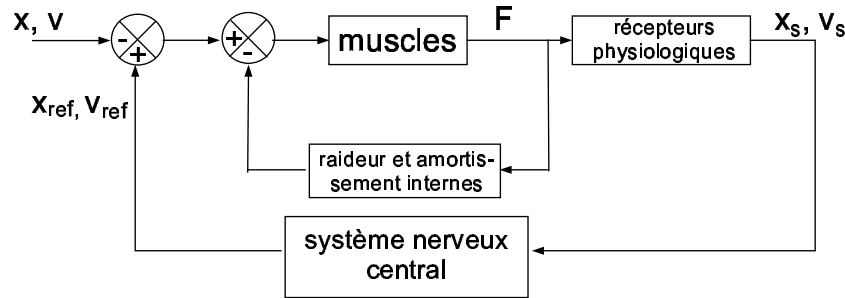


FIG. 1.8 – Schéma simplifié des boucles de contrôle d'un membre (adapté d'après [FCvdL04])

La différence entre les positions et vitesses ( $x_{ref}, v_{ref}$ ) que le système nerveux central cherche à imposer au membre et les positions et vitesses appliquées par l'environnement extérieur ( $x, y$ ) vont servir à contracter le muscle. Les capteurs physiologiques vont alors mesurer les positions et vitesses obtenues ( $x_s, y_s$ ) qui vont être interprétées par le système nerveux central.

Hajian et Howe ont cherché à caractériser l'impédance des doigts [HH97]. Ils considèrent que la relation entre une force transitoire  $f(t)$  appliquée au bout du doigt et le déplacement correspondant  $x(t)$  est donné par l'équation (1.2) avec  $m$  la masse équivalente du doigt,  $b$  le frottement visqueux de la dernière articulation et  $k$  la raideur de celle-ci. Ils ont trouvé une masse comprise entre 3,5 et 8,7 g, un amortissement entre 4,02 et 7,40  $N.s.m^{-1}$  et une raideur entre 255 et 1255  $N.m^{-1}$ . Le coefficient d'amortissement  $\xi = (b/2\sqrt{mk})$  est environ égal à 1,4.

$$m\ddot{x}(t) + b\dot{x}(t) + kx(t) = f(t) \quad (1.2)$$

Milner et Franklin obtiennent quant à eux des coefficients plus importants et ont montré que la raideur du doigt est anisotropique suivant la position du doigt et la direction de la force reçue [MF98].

Toutes ces études confirment que l'humain est difficile à modéliser. Le plus souvent, l'utilisateur est considéré comme un élément perturbateur du périphérique haptique. Des

modèles humains plus performants permettraient cependant d'améliorer la commande des interfaces haptiques.

### 1.3.3 Types de retours haptiques

Un moyen d'appréhender une scène virtuelle par le sens du toucher est de disposer d'une réplique identique de cette scène sous forme matérielle. On parle alors de *retour haptique passif*. L'objet manipulé dans le monde virtuel est le même que celui manipulé en réalité et les forces ressenties par l'utilisateur sont celles générées entre les objets (appelés *props*) du monde réel. Cette approche a été utilisée par Hinckley dans un outil de visualisation de données médicales 3D obtenues avec un scanner [HPGK94].

Une seconde approche est le *retour pseudo-haptique* qui consiste à exploiter les illusions sensorielles entre la vue et le toucher par exemple. Ce type de retour a été mis en évidence par Lécuyer et al. [Léc01][LCK<sup>+</sup>00]. Dans une expérimentation avec une Spaceball dont le coefficient de raideur reste constant, ces chercheurs ont ainsi montré qu'un utilisateur est capable de discriminer les raideurs de ressorts virtuels en jouant simplement sur le retour visuel.

Une troisième approche est de créer un retour en utilisant des dispositifs qui freinent l'interface. Ce *retour haptique dissipatif* a été évalué par Paljic [Pal04] en utilisant un SPIDAR (voir l'interface en section 1.3.5). Cette technique se contente de s'opposer à un mouvement sans en créer elle-même. Ce retour a également été mis en œuvre dans une souris équipée d'un électro-aimant [SMO04].

Une dernière approche est le *retour haptique actif*. Dans ce cas, une interface haptique est utilisée pour renvoyer à l'utilisateur des informations de type retour tactile ou retour kinesthésique, calculées par la simulation informatique. C'est le retour haptique le plus employé.

### 1.3.4 Technologies d'actionneurs pour le retour d'effort

Différentes technologies d'actionneurs existent aujourd'hui pour réaliser le retour haptique actif, appelé retour de force. Les plus utilisés sont les moteurs à courant continu car ils sont bon marché et faciles à commander (le couple de sortie du moteur est proportionnel au courant). Les moteurs piézo-électriques semblent prometteurs pour les applications à retour d'effort car leur couple massique est important. Cependant leur commande en est encore au stade de la recherche [GLSH01] [GLS04][IM95].

D'autres technologies existent mais ne sont employées qu'à titre anecdotique dans les applications à retour d'effort. On peut citer les technologies hydrauliques, pneumatiques, électromagnétiques (Magic Wrist d'IBM [BH97]), les fluides électro-rhéologiques, les fluides magnéto-rhéologiques [BSSR02], les polymères électroactifs et les alliages à mémoire de forme.

### 1.3.5 Interfaces à retour d'effort

Nous pouvons distinguer deux grandes familles de périphériques à retour d'effort : les périphériques à base fixe<sup>10</sup> et les périphériques à base non-fixe<sup>11</sup>. La première catégorie regroupe les interfaces à retour de force de type bras, stylos, manches ou souris à retour d'effort et systèmes à câbles. La seconde catégorie regroupe tous les périphériques attachés à l'utilisateur de type gants et exosquelettes.

#### Les gants

Les gants permettent la manipulation d'objets avec une grande liberté de mouvement. Dans ce cas, le retour d'effort permet de ressentir la rigidité de l'objet mais ne permet pas de ressentir son poids. Nous pouvons citer comme exemple les gants Rutgers Master II (fig. 1.9) qui permettent de renvoyer des forces à quatre doigts de la main grâce à des vérins pneumatiques[BBPB02]. L'architecture mécanique de ces gants a l'inconvénient d'empêcher la fermeture totale du poing mais l'avantage de les rendre particulièrement légers : moins de 100g. Le CyberGrasp commercialisé par la société Immersion [CC] est un autre exemple de réalisation. Ces gants se distinguent des précédents par la structure exosquelette disposée à l'extérieur de la main de manière à laisser la possibilité à l'utilisateur de fermer complètement la main (fig. 1.9).



FIG. 1.9 – Gants Rutgers et le CyberGrasp

<sup>10</sup> *ground-based* ou *desk-based* en anglais

<sup>11</sup> *man-based* en anglais

Les gants peuvent être couplés à un bras à retour d'effort trois degrés de liberté actifs pour compenser le poids du gant tout en laissant la possibilité à l'utilisateur de ressentir le poids des objets manipulés et les forces de contacts avec l'environnement. C'est le cas du CyberForce commercialisé par la société Immersion [CC] qui est utilisé en combinaison avec le CyberGrasp (fig. 1.10).



FIG. 1.10 – Le CyberForce

### Bras maîtres et stylos à retour de force

Les bras maîtres sont principalement utilisés dans les applications de télé-opérations. Ces systèmes, placés soit sur une table ou sur le sol, sont capables de fournir des forces puissantes à l'utilisateur. Ils sont également utilisés dans les applications de réalité virtuelle.

Ces périphériques ont six degrés de liberté en entrée avec, le plus souvent, trois ddls motorisés pour les translations. Nous pouvons citer comme exemple de périphériques à six degrés de liberté actifs, le Virtuouse6D commercialisé par la société Haption [Hapb], le 6-dof DELTA Haptic Device [GCR<sup>+</sup>01] commercialisé par Force Dimension [FD] et le PHANToM6D commercialisé par Sensable [Sen] (fig. 1.11).

Les versions avec trois ddls actifs sont les plus utilisées avec en particulier le PHANToM Desktop [MS94] et le PHANToM Omni de Sensable (fig. 1.12). Nous pouvons également citer le HapticMASTER [dLLFR02], périphérique à trois degrés de liberté actifs commercialisé par FCS Control Systems [Hapa].

### Souris et manches à retour de force

Comme dispositifs à deux degrés de liberté, nous pouvons citer le PenCat/Pro d'Immersion [Imm] qui permet à l'utilisateur de ressentir un effort à l'extrémité d'un stylo suivant deux degrés de liberté ou encore la souris à retour d'effort Wingman Force Feedback de Logitech [Log] (fig. 1.13). Il existe par ailleurs les joysticks à retour de force,



FIG. 1.11 – Le Virtuouse6D, le DELTA Haptic Device et PHANToM6D.

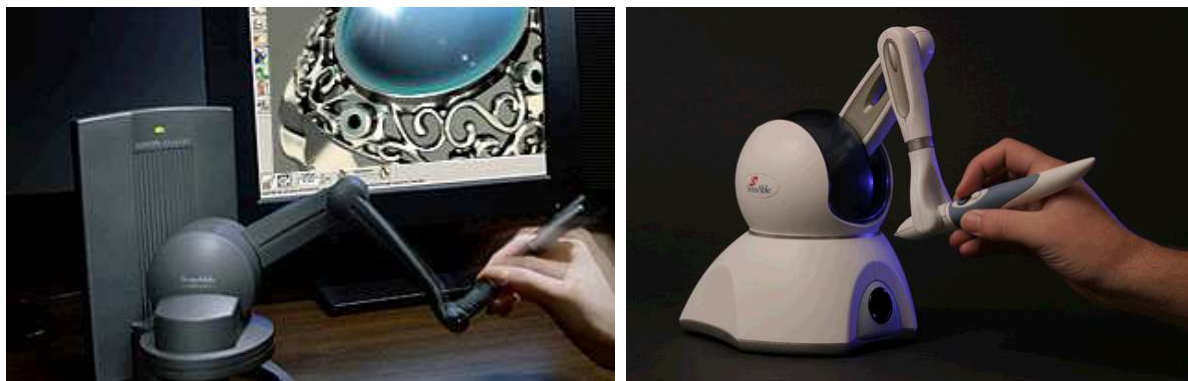


FIG. 1.12 – Le PHANToM Desktop et le PHANToM Omni.

principalement utilisés dans les jeux de simulateurs de vol, qui renvoient différents effets à l'utilisateur suivant la situation. On peut citer par exemple le Sidewinder force feedback 2 de Microsoft [Micb].

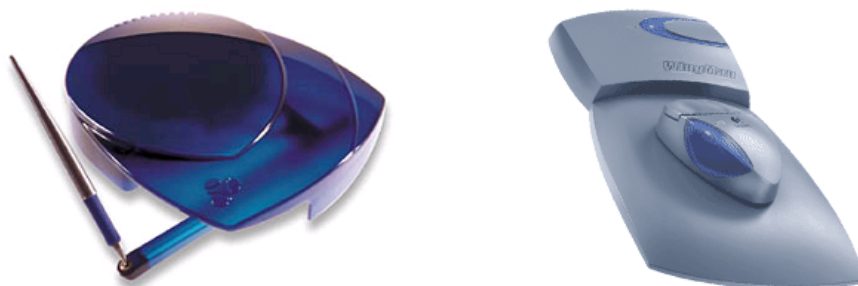


FIG. 1.13 – Le PenCat/Pro et la Wingman Force Feedback.

### Systemes à câbles

Les dispositifs à câbles ont l'avantage de simplifier considérablement la mécanique des dispositifs précédents et permettent de multiples configurations. Le SPIDAR (Space Interface Device for Artificial Reality) de Sato et al. peut être utilisé pour saisir des objets virtuels dans le cas du SPIDAR-8 [WKS99]. Dans cette configuration, trois câbles sont attachés à chaque doigt. Une autre configuration permet de disposer d'un périphérique 6 dds actifs dans le cas du SPIDAR-G [KHKS02] (fig. 1.14). D'autres configurations sont également envisageables. L'inconvénient de ces dispositifs est qu'il faut veiller à éviter d'entremêler les câbles.

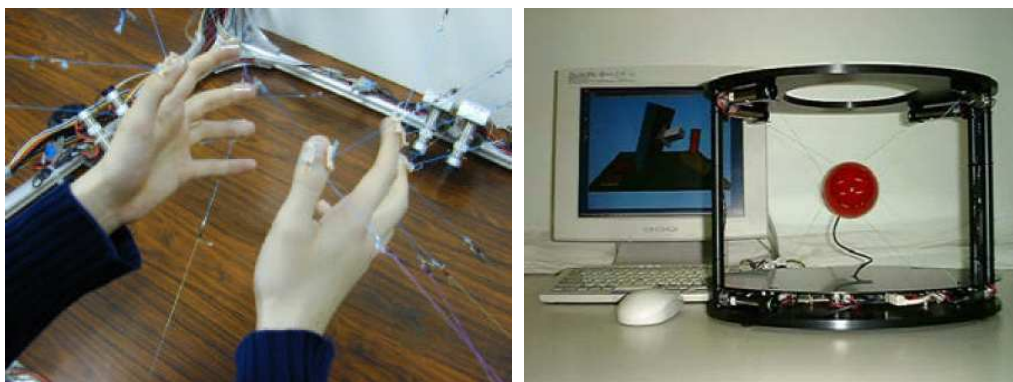


FIG. 1.14 – Le SPIDAR-8 et le SPIDAR-G.

### 1.3.6 Interfaces à retour tactile

La recherche sur les interfaces tactiles n'en est qu'à ses débuts. Les périphériques présentés ici sont donnés à titre d'exemple et ne constituent pas un inventaire exhaustif

de l'état de l'art sur les interfaces à retour tactile.

Les interfaces existantes sont loin de pouvoir simuler correctement les phénomènes réels. La technique habituellement utilisée est de créer une matrice d'aiguilles que l'on va faire vibrer à différentes fréquences pour chercher à recréer une texture. Les actionneurs utilisés sont le plus souvent piézo-électriques, comme pour le STRESS de l'université de McGill [PH03] (fig. 1.15). Si cette technique donne de bons résultats pour créer des motifs Braille, il n'existe pas à l'heure actuelle de modèle pour reproduire une texture physique. D'autres dispositifs à retour tactile utilisent de simples vibreurs pour transmettre une information à l'utilisateur. Par exemple, la souris iFeel de Logitech [Log] est une souris optique classique équipée d'un vibreur qui produit différents types de vibrations suivant la nature des éléments pointés par la souris (icône ou lien hypertexte par exemple).

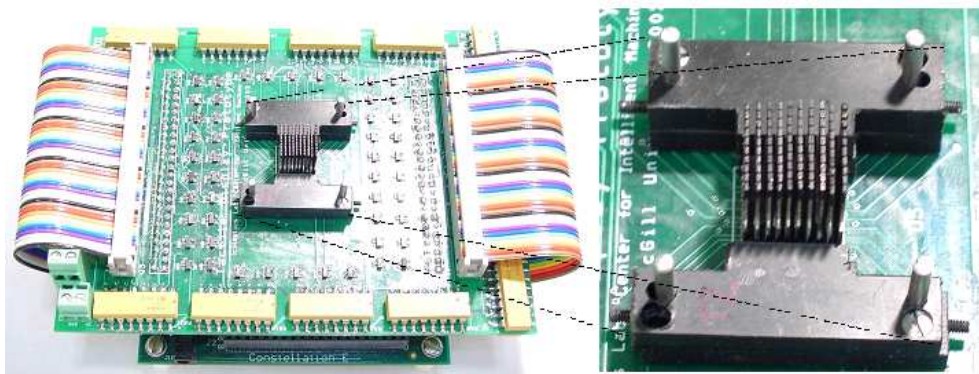


FIG. 1.15 – Le STRESS de l'université de McGill.

## 1.4 Intégration du retour haptique

L'intégration du retour haptique comprend plusieurs étapes : d'une part, la création d'un environnement virtuel capable de simuler des phénomènes physiques plus ou moins réalistes et de calculer des forces ; et d'autre part, la commande de l'interface haptique pour rendre ces efforts à l'utilisateur. Nous allons voir les méthodes existantes et les contraintes pour réaliser ces deux étapes.

Plus précisément, les différentes étapes schématisées sur la figure 1.16 se décomposent de la manière suivante :

1. Acquisition de la position de l'effecteur par les capteurs de position de l'interface et mise à jour de la position de l'objet manipulé dans l'environnement virtuel.
2. Détection de collisions entre l'objet manipulé (ou l'avatar du doigt ou de la main) et les autres éléments de la scène virtuelle.
3. Calcul du retour d'effort. Dans le cas d'une collision, le moteur physique de l'environnement virtuel calcule une force de réaction qui peut être modifiée pour rajouter des effets de texture par exemple.
4. Envoi des consignes de forces à l'interface haptique. L'interface haptique applique les courants nécessaires aux moteurs.

- L'utilisateur ressent alors un effort sur sa main grâce à ses capteurs cutanés et kinesthésiques. Ces informations vont être acheminées au cerveau qui va envoyer des commandes aux muscles. Les muscles font alors bouger l'effecteur et nous recommençons à la première étape.

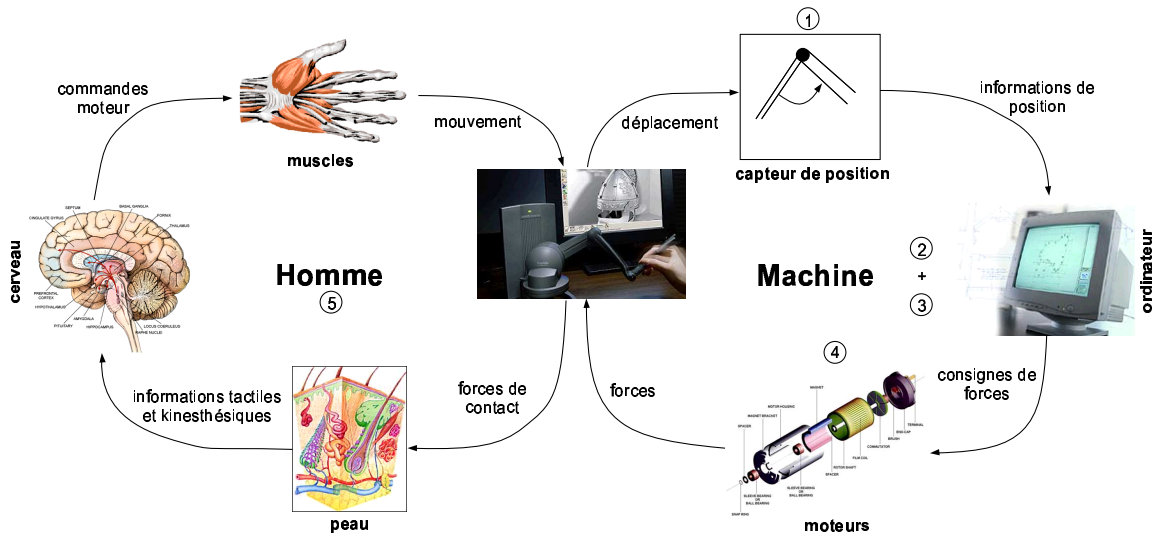


FIG. 1.16 – Schéma d'architecture de rendu haptique - interaction entre l'homme et la machine (inspiré par [SB97])

Dans l'exemple présenté sur la figure 1.16, les moteurs contrôlent la force appliquée en fonction des positions du périphérique. On parle dans ce cas de **contrôle en impédance**. Il est également possible de mesurer une force ou un couple et de contrôler le déplacement de l'interface par le biais des moteurs. Dans ce cas, on parle de **contrôle en admittance**. Le contrôle en impédance, plus simple à mettre en œuvre, est utilisé dans la majorité des périphériques à retour de force. C'est le cas du Phantom. Ce type de contrôle est mieux adapté à la simulation de raideurs faibles à modérées (pour des raisons de stabilité). Le contrôle en admittance est, quant à lui, mieux adapté à la simulation de raideurs importantes (toujours pour des raisons de stabilité) et a également l'avantage de masquer des défauts de la mécanique comme les frottements. Il est par exemple utilisé pour le HapticMaster [dLLFR02].

Nous allons étudier les deux étapes les plus importantes que sont la détection de collision et de calcul de force dans l'environnement virtuel.

### 1.4.1 Détection de collision

L'algorithme de détection de collisions dépend des modèles mathématiques de description des objets de l'environnement : description volumique, utilisation de surfaces paramétriques, de surfaces polygonales ou de fonctions implicites. L'algorithme doit également prendre en compte la dynamique des objets (statiques ou en mouvement) et leur rigidité (rigides ou déformables).

La détection de collisions s'effectue en plusieurs étapes. Une première étape utilisant des boites englobantes autour des objets, consiste à détecter, en cas de collision, quels



objets entrent en collision. En cas de collision, une seconde étape permet de déterminer quelles sont les parties des objets en collision.

Il existe ensuite deux familles d'algorithmes de détection de collision : les algorithmes qui réalisent une détection de collision discrète et ceux qui réalisent une détection continue. Dans le premier cas, il est possible de savoir s'il y a eu collision au pas courant de la simulation. Dans le second cas, il est possible de déterminer l'instant de collision entre deux pas de la simulation.

Des bibliothèques génériques existent comme H-COLLIDE qui réalise une détection de collision discrète ou CONTACT développée par Redon et al. [RKC00] qui réalise une détection de collision continue.

## 1.4.2 Calcul du retour de force

Cette étape est critique dans le rendu haptique car elle détermine la stabilité de l'interface haptique. Le retour d'effort est réalisé avec des périphériques isotoniques avec contrôle en position.

La technique généralement utilisée est la mesure de la distance de pénétration des objets en collision, qui correspond à la distance entre l'objet manipulé et la projection de celui-ci à la surface de l'objet en collision. La force renvoyée est alors proportionnelle à cette distance (méthode par pénalités [WKA92]). L'objet visible à l'écran (appelé "god object" [ZS95] ou "proxy" [RKK97]) est la projection de l'objet manipulé qui est en pénétration dans l'objet en collision, à la surface de celui-ci.

Ce type de méthode peut poser des problèmes de discontinuités. La distance de pénétration correspond en effet à la minimisation de la distance entre la position de l'objet manipulé par l'utilisateur et les différentes facettes de l'objet en collision. Si l'utilisateur pénètre un objet fin et qu'il dépasse la moitié de son épaisseur, le proxy va se trouver projeté de l'autre côté de l'objet et l'utilisateur qui ressent d'abord une force de résistance dans le mur, ressent brusquement une force qui l'entraîne de l'autre côté du mur.

Zilles et Salisbury ont proposé une méthode pour résoudre ce problème de discontinuités en cherchant à maintenir de manière continue le pointeur à la surface géométrique de l'objet en collision [ZS95]. Ils utilisent pour cela un historique des positions du pointeur et calculent la position du proxy en minimisant la distance entre l'objet manipulé et la surface initiale de collision (fig. 1.17).

Dans le cas de la simulation dynamique de l'environnement par un moteur physique, les forces calculées dans l'environnement virtuel viennent de la résolution d'équations différentielles du mouvement.

Il est possible de distinguer deux types d'environnements physiques : ceux constitués de corps rigides, qui représentent la majorité des environnements simulés, et ceux constitués de corps déformables. La simulation de contacts rigides est particulièrement délicate à restituer car elle tend à déstabiliser l'interface haptique alors que la simulation de contacts déformables ne pose pas de problème particulier pour la stabilité de l'interface haptique. Pour ces raisons, nous allons étudier plus en détail comment sont simulés les contacts rigides.

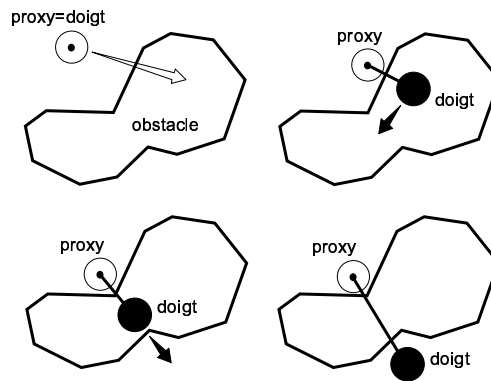


FIG. 1.17 – Le proxy bouge localement pour minimiser la distance entre la position du doigt de l'utilisateur et les contraintes de l'environnement (adapté de [MHS01]).

### Simulation de contacts rigides

Pour être réaliste, un mur doit avoir la plus grande raideur possible tout en restant stable pour éviter les phénomènes de rebonds lorsque l'utilisateur vient en contact. Le réglage de la raideur et de l'amortissement du mur est un problème récurrent en interfaçage haptique, en particulier pour les périphériques commandés en impédance.

Nous considérons le cas de la simulation d'un mur virtuel, modélisé par un ressort qui possède une caractéristique d'effort saturé (fig. 1.18). Deux paramètres sont importants pour que le mur soit perçu comme réaliste. Le premier est la force maximale à rendre ( $F_{sat}$ ) qui doit être supérieure à la force maximale que peut exercer l'utilisateur. Le second est la raideur du mur ( $k$ ) qui doit être supérieure à un certain seuil. Le seuil de perception de rigidité d'un objet a été étudié par Tan et al. [TSEC94]. Ils ont montré que la raideur minimale pour simuler un mur rigide doit être comprise entre  $15300$  et  $41500 N.m^{-1}$ .

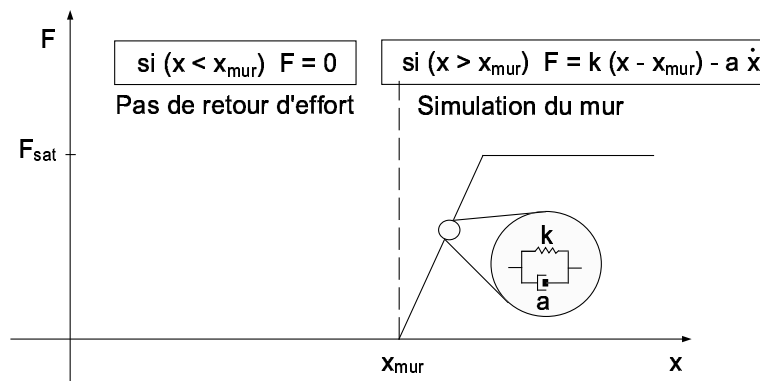


FIG. 1.18 – Modélisation d'un mur par un ressort et un amortisseur en parallèle.

L'augmentation de la raideur du mur virtuel va cependant tendre à déstabiliser le système haptique. En effet, le mur virtuel n'est pas passif<sup>12</sup> ; il crée de l'énergie. Un ressort

<sup>12</sup>Un dispositif est considéré comme passif si l'intégrale de la puissance instantanée est supérieure ou égale à zéro pour l'ensemble des forces générées dans une plage de temps donnée.  $\int_0^t f(\tau) v(\tau) d\tau \geq 0 \forall f(t), t \geq 0$  [CGSS93].

physique idéal est un système sans perte : l'énergie emmagasinée lors de la compression du ressort est intégralement rendue lors du relâchement du ressort. Prenons maintenant le cas d'un ressort virtuel (fig. 1.19). Parce qu'il est programmé en temps discret, la force qu'il va générer ne va pas croître régulièrement avec la compression. Celle-ci va en effet rester constante pendant un certain temps avant d'être mise à jour. A cause de ce phénomène, la force moyenne lors de la compression sera légèrement inférieure à celle d'un ressort réel de raideur identique et la force moyenne lors du relâchement sera légèrement supérieure. Cela montre que la période d'échantillonnage doit être la plus petite possible et être fixée précisément. Pour répondre à ces exigences, un système d'exploitation assurant le temps réel<sup>13</sup> est généralement utilisé avec une fréquence d'échantillonnage supérieure ou égale à 1000 Hz. Pour rendre le système passif, un amortissement est introduit (*a*) afin de dissiper de l'énergie.

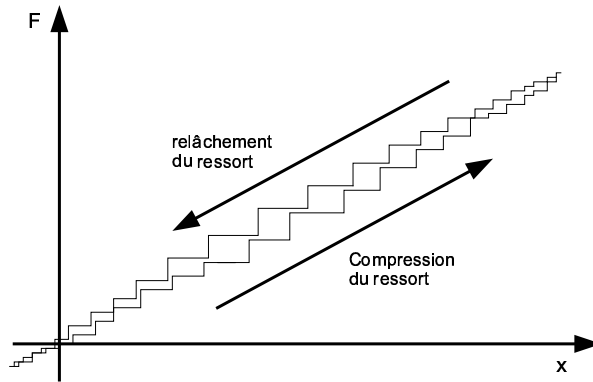


FIG. 1.19 – Illustration de la création d'énergie d'un ressort virtuel (d'après [CGSS93]).

Pour rendre le système stable, Minsky et al. [MyMS<sup>+</sup>90] ont proposé l'expression (1.3) qui donne une relation entre les paramètres du mur (*a* et *k*) et la période d'échantillonnage du système (*T*), *c* étant une constante environ égale à 0,5.

$$\frac{a}{kT} > c \quad (1.3)$$

On en déduit que des périodes d'échantillonnage importantes peuvent être compensées en augmentant le coefficient d'amortissement. Cependant Colgate et al. [CGSS93] ont montré que les performances sont dégradées par l'augmentation de *a* et *T*. Ils définissent par ailleurs un abaque permettant de savoir si un mur sera passif en fonction des coefficients du mur et du coefficient de frottement visqueux du périphérique (*b*). Ce dernier a pour effet d'augmenter les marges de passivité. Ces chercheurs définissent également la relation 1.4 qui donne le frottement visqueux minimal que doit avoir l'interface haptique suivant les coefficients du mur et la période d'échantillonnage. L'influence du frottement visqueux a été vérifiée par Salcudean et al. qui ont simulé un mur pour une interface

<sup>13</sup>Une définition du temps réel pourrait être la suivante : « Un système est dit temps réel lorsque l'information après acquisition et traitement reste encore pertinente ». Dans le cas d'une information arrivant de façon régulière (sous forme d'une interruption périodique du système), un système temps réel est tel que les temps d'acquisition et de traitement doivent rester inférieurs au temps de rafraîchissement de cette information.

à lévitation magnétique [SV97]. L'augmentation du frottement visqueux de l'interface, si elle permet de stabiliser les murs de raideurs importantes, va diminuer cependant la transparence de l'interface en environnement libre. La transparence est la capacité d'une interface à devenir transparente (non perçue) par l'utilisateur au cours de son utilisation [FM03a]. Pour cela, il faut compenser à tout moment le poids de l'interface, son inertie et ses frottements.

$$b > \frac{kT}{2} + a \quad (1.4)$$

Dans les exemples ci-dessus, les coefficients de raideur et d'amortissement du mur virtuel déterminent les correcteurs de la loi de commande de l'interface haptique. L'environnement virtuel doit assurer la réalisation d'un retour de force réaliste tout en assurant la stabilité du périphérique haptique. Cela peut poser problème si plusieurs périphériques haptiques différents sont utilisés dans le même environnement virtuel. Les critères de stabilité de l'un ne sont pas forcément ceux d'un autre.

Pour découpler le périphérique haptique de la scène virtuelle, Colgate et al. ont introduit le couplage virtuel pour les périphériques contrôlés en impédance [CSB95]. Adams et al. [AH99] généralisent cette méthode pour le contrôle en impédance ou admittance. Cette méthode est particulièrement intéressante quand il s'agit de garantir la stabilité lors de la simulation de tout une gamme d'impédances.

En général, seuls la masse et le coefficient de frottement visqueux de l'interface haptique sont pris en compte pour définir les conditions de stabilité.

### 1.4.3 Architecture

La réalisation d'un système haptique nécessite l'affichage de la scène virtuelle avec une mise à jour des éléments à une fréquence supérieure à 25 Hz, une détection de collisions et un calcul de force à une fréquence au moins égale à 1000 Hz ainsi qu'une mise à jour des forces sur l'interface haptique à une fréquence au moins égale à 1000 Hz. La détection de collisions avec le calcul des forces et la mise à jour de l'affichage se font généralement sur la même machine. Une boucle est responsable de l'affichage et une autre de la détection de collisions et du calcul des forces. Le contrôle de l'interface haptique se fait généralement sur une autre machine dédiée avec un système d'exploitation orienté temps réel.

Il existe un certain nombre de bibliothèques permettant de développer l'environnement 3D telles que OpenGL Performer [OP] ou OpenSG [Opeb] qui utilisent la bibliothèque de programmation graphique OpenGL [Opea]. Des bibliothèques haptiques permettent de gérer plus facilement les autres étapes. Elles sont généralement limitées à l'utilisation d'un ou plusieurs périphériques du marché : GHOST de Sensable [Sen] ou Reachin API de Reachin Technologies AB [RTA].

Pour clore le chapitre, nous allons donner quelques exemples d'application du retour haptique.

## 1.5 Applications du retour haptique

Le retour haptique peut être utilisé dans toute application où l'on cherche à renvoyer des informations haptiques à l'utilisateur. Comme principaux domaines d'application, nous pouvons citer :

- *Médecine* : simulateurs chirurgicaux, manipulation de micro et macro robots pour la chirurgie minimale invasive, diagnostics à distance pour la télémédecine, aide pour les handicapés (ex : interfaces haptiques pour les aveugles).
- *Jeux* : jeux vidéos et simulateurs qui permettent à l'utilisateur de sentir différents effets suivant la situation. Par exemple, les volants à retour de force dans les simulateurs de conduite renvoient une force à l'utilisateur qui dépend de la vitesse du véhicule et de l'état de la route.
- *Enseignement* : donner par exemple la possibilité aux étudiants de sentir un phénomène aux échelles nanoscopiques, macroscopiques ou astronomiques.
- *Industrie* : intégration de l'haptique dans les logiciels de CAO pour permettre à l'utilisateur de ressentir les forces de collision lors d'assemblage de pièces. Simulation d'opérations de montage/démontage de pièces. Téléopération (contrôle d'une machine située sur un site distant ou dans un environnement hostile).
- *Art* : musées dans lesquels l'utilisateur peut toucher les objets, sculpture individuelle ou collective sur internet. Par exemple, le logiciel FreeForm de Sensable permet à l'utilisateur de sculpter des virtuellement des objets.

## 1.6 Conclusion

Dans ce chapitre, nous avons présenté la classification de Card et al. qui sépare les périphériques suivant la nature de leurs degrés de liberté et le type de grandeur mesurée.

A partir de cette classification, nous avons distingué les périphériques suivant leur degré de résistance (isotonique, élastique ou isométrique). Nous avons vu que les périphériques isotoniques sont mieux adaptés au contrôle en position et les périphériques isométriques et élastiques au contrôle en vitesse.

La classification, montrant les relations entre les degrés de liberté, permet également de classer les périphériques suivant le degré de séparation de leur degrés de liberté. Nous avons vu qu'il y a une relation entre le degré de séparation des degrés de liberté du périphérique et le degré de séparation des degrés de liberté de la tâche : les interfaces intégrales sont mieux adaptées aux tâches intégrales et les interfaces séparables aux tâches séparables.

Nous avons par ailleurs étudié les paramètres qui influencent le degré d'isomorphisme d'un périphérique. Parmi les cinq critères définis, nous avons étudié plus en détail l'influence de la fonction de transfert à travers l'étude du control display. Si une sensibilité proche de un améliore l'isomorphisme, une modification de la sensibilité permet d'optimiser les performances de l'utilisateur suivant la tâche.

Nous avons ensuite étudié le retour haptique qui permet d'établir une relation bidirectionnelle entre l'utilisateur et la machine, en utilisant le sens du toucher. Les caractéristiques physiologiques de l'homme conduisent à la distinction du retour d'effort

et du retour tactile. Le premier est le retour principalement sollicité, compte tenu des technologies existantes.

Finalement, nous avons étudié les problèmes posés par l'intégration. Nous avons en particulier étudié le cas de la simulation d'un mur virtuel et nous avons vu que la simulation est particulièrement délicate et tend à déstabiliser l'interface haptique.

Fort de notre classification des périphériques et de nos connaissances des interfaces haptiques, nous allons montrer dans le prochain chapitre les limites des périphériques existants. Cela nous permettra de proposer une interface haptique originale.

# Chapitre 2

## Conception, réalisation et commande du DigiHaptic

### Sommaire

---

<b>2.1</b>	<b>Introduction</b>	<b>31</b>
<b>2.2</b>	<b>Conception</b>	<b>32</b>
2.2.1	Limites des périphériques existants	32
2.2.2	Choix de la séparation des ddl	33
2.2.3	Périphériques à ddl séparés existants	33
2.2.4	Affordances	35
2.2.5	Fatigue	36
<b>2.3</b>	<b>Réalisation de la maquette</b>	<b>37</b>
2.3.1	Choix des doigts	37
2.3.2	Ergonomie	37
<b>2.4</b>	<b>Réalisation du prototype fonctionnel</b>	<b>38</b>
2.4.1	Conception mécanique	39
2.4.2	Conception électronique	42
<b>2.5</b>	<b>Identification des paramètres du système</b>	<b>47</b>
2.5.1	Détermination de l'inertie de la manette J	48
2.5.2	Détermination des frottements secs et visqueux	48
<b>2.6</b>	<b>Commande du système</b>	<b>50</b>
2.6.1	Filtrage de la vitesse	51
2.6.2	Commande en position	52
2.6.3	Calcul des paramètres optimaux d'un mur virtuel	53
2.6.4	Commande en impédance du système	57
<b>2.7</b>	<b>Conclusion</b>	<b>60</b>

---

## 2.1 Introduction

Ce chapitre débute avec la description de la démarche qui nous a permis de concevoir le DigiHaptic [CPCS03b][CPCS03a][CCSP03][FM03b][CP03]. A partir des limites des pé-

riphériques existants et dans un souci d'innovation, nous avons proposé la séparation des degrés de liberté.

Nous montrons ensuite la mise en oeuvre de cette idée à travers la résolution des problèmes de dimensionnement mécanique, de conception des cartes de commande et de connexion du périphérique avec un environnement virtuel.

Enfin, nous identifions les paramètres du système pour en optimiser la commande.

## 2.2 Conception

### 2.2.1 Limites des périphériques existants

La caractéristique commune de la plupart des périphériques 3D existants, qu'ils soient isotoniques, élastiques ou isométriques, est de posséder un unique effecteur que l'utilisateur manipule. Nous allons voir les limites de chaque catégorie de périphérique.

Les périphériques isotoniques, utilisés pour le contrôle en position des objets manipulés, ont le principal avantage d'être intuitifs, puisque les mouvements de la main sont directement reproduits dans l'environnement virtuel. Ainsi la souris, pour les environnements 2D, est utilisée depuis l'apparition des fenêtres et boutons dans les tâches de pointage. Ce périphérique isotonique relatif peut, en outre, être débrayé si l'utilisateur la soulève et la dépose ailleurs sur le bureau, ce qui permet de changer la position du périphérique sans changer la position du pointeur. En ce qui concerne les périphériques isotoniques absolus pour les environnements 3D, à chaîne cinématique ouverte comme le Phantom [MS94] ou sans chaîne cinématique comme la flying mouse (voir §1.2.2), le débrayage est plus problématique. Dans le premier cas, l'espace de travail est limité par l'amplitude de débattement de chaque liaison mécanique ; dans le second, il est limité par la position des capteurs de position dans l'espace et la position de l'utilisateur par rapport à la scène virtuelle. Dans les deux cas, il n'est pas possible de débrayer le périphérique aussi *naturellement* que la souris et la plupart du temps un bouton est utilisé à cette fin.

Les boutons sur les périphériques sont en général utilisés pour des tâches de sélection dans l'environnement de travail. Leur utilisation pour le débrayage ne rend pas l'utilisation du périphérique intuitive, elle est fastidieuse et constitue une perte de temps pour l'utilisateur. Pour éviter cela, la sensibilité du périphérique est adaptée en faisant coïncider les espaces de travail matériel et virtuel. Avec les périphériques isotoniques, l'espace de travail dans l'environnement virtuel est donc limité : il est proportionnel à l'espace de travail matériel.

Un autre inconvénient des périphériques isotoniques trois ddl est qu'ils peuvent se révéler fatigants après une courte durée d'utilisation puisque l'utilisateur travaille à main levée et peut faire des mouvements plus ou moins importants. Malgré ces problèmes, le retour d'effort sur ces périphériques est "conceptuellement" simple à mettre en oeuvre puisque l'on cherche à renvoyer sur la main de l'utilisateur une force proportionnelle à celle calculée dans l'environnement virtuel.

Les périphériques isométriques et élastiques sont utilisés pour réaliser un contrôle en vitesse qui n'est pas naturel et qui s'acquiert après apprentissage. Ils ont l'avantage d'être



peu fatigants puisque le débattement de l'effecteur manipulé par l'utilisateur est relativement faible. Cependant, cela a pour conséquence de rendre difficile la réalisation de mouvements précis ; plus particulièrement la réalisation d'une translation suivant une direction donnée dans un plan de l'environnement 3D s'avère délicate, à moins que l'utilisateur ne désactive, par un quelconque moyen, un des degrés de liberté.

Les périphériques isométriques ne bougent pas et ne permettent donc aucun retour d'effort. Quant aux périphériques élastiques, ils ont un débattement qui permet de les utiliser en retour de force actif.

Chaque périphérique est adapté à un type d'application. Réaliser un périphérique qui conjugue les avantages de chaque catégorie de périphériques (non fatigant, intuitif, utilisable dans des environnements virtuels de tailles réduites ou infinies. . .) est donc une chimère.

Par ailleurs, nous considérons que le développement d'un nouveau périphérique de bureau à un effecteur, qu'il soit isotonique ou élastique, aboutira, quelque soit l'architecture mécanique choisie, à un "Phantom like" ou une "SpaceMouse like".

Pour proposer plus qu'une simple innovation sur l'architecture mécanique, proposer de nouveaux modes d'interaction et essayer de conjuguer le maximum d'avantages de chaque catégorie de périphériques, nous avons eu l'idée de séparer les degrés de liberté.

### 2.2.2 Choix de la séparation des ddl

La séparation des degrés de liberté a bien entendu l'avantage d'en permettre un contrôle séparé. Cela permet également la simplification de la conception mécanique pour le retour d'effort et ainsi la diminution des coûts de fabrication, par rapport aux périphériques possédant un effecteur à trois degrés de liberté assemblés.

Nous nous plaçons dans le cas d'un périphérique de bureau et nous faisons le choix d'utiliser une seule main pour manipuler les degrés de liberté séparés. Cela permet ainsi de minimiser l'encombrement du périphérique et de libérer une main qui peut servir à autre chose. Séparer les degrés de liberté suppose donc de les contrôler avec les doigts, ce qui permet alors d'avoir la paume de la main posée. On obtient ainsi un périphérique non fatigant. D'autre part, l'utilisation du bout des doigts pour ressentir des forces est particulièrement intéressante puisque ce sont les parties les plus sensibles de la main au retour d'effort [ZMB96].

Par ailleurs, compte tenu du débattement possible de chaque doigt, l'utilisation du périphérique peut être envisagée en modes isotonique ou élastique.

Cette séparation des ddl pose cependant certaines questions. Les plus importantes sont : l'homme est-il capable de contrôler des degrés de liberté séparés ? Et si oui, dans quelles tâches et après quel temps d'apprentissage ?

### 2.2.3 Périphériques à ddl séparés existants

Il existe des périphériques à degrés de liberté séparés. Par exemple le SPIDAR [WKS99] est un périphérique à câbles qui permet dans certaines configuration à l'utilisateur de jouer au Rubicube (fig. 2.1). Chaque doigt est tenu par trois câbles motorisés qui exercent un

retour de force passif en freinant les câbles. Dans l'exemple de la figure 2.1, les doigts reproduits dans l'environnement virtuel permettent un contrôle en position des objets virtuels manipulés. Chaque doigt de l'utilisateur a deux degrés de liberté et contrôle un doigt dans l'environnement virtuel qui a lui même deux degrés de liberté. On a donc à chaque fois deux degrés de liberté intégrés réels qui contrôlent deux degrés de liberté intégrés virtuels. Pour ce périphérique dans cette application, on ne peut pas parler d'interface séparable au sens de Jacob (voir §1.2.4) puisque l'utilisateur contrôle sur chacun des doigts des degrés de liberté intégrés.



FIG. 2.1 – Le Spidar et le Claw

Le clavier est, quant à lui, un périphérique discret à degrés de liberté séparés et peut être qualifié d'interface séparable au sens de Jacob. Il permet bien sûr de taper du texte mais il est aussi utilisé dans les jeux du type "Quake like" aussi appelés *First Person Shooter (FPS) games* (fig. 2.2). Dans ces jeux 3D, les touches du clavier permettent de déplacer le joueur suivant deux dimensions : avancer/reculer - gauche/droite. Pour un clavier azerty, il s'agit des touches "q" et "d" pour déplacer le personnage latéralement suivant sa gauche ou sa droite, "z" et "s" pour déplacer le personnage en avant ou en arrière, la souris servant à orienter la vue du personnage mais surtout la cible de l'arme utilisée pour abattre les adversaires. Dans cet exemple, l'utilisateur utilise quatre degrés de liberté séparés pour contrôler deux degrés de liberté intégrés à l'écran. Ici chaque degré de liberté matériel est unidirectionnel. Nous remarquons que les touches utilisées sont placées de manière intuitive les unes par rapport aux autres. Si les touches "a", "z" avaient été utilisées pour faire avancer/reculer le personnage et les touches "e" et "r" pour réaliser les déplacements latéraux, le contrôle du personnage aurait été beaucoup moins intuitif et aurait nécessité un temps d'apprentissage beaucoup plus long.

Le Claw [Cla] est quant à lui un périphérique muni de boutons à l'extrémité des doigts de la main gauche. Il se propose de remplacer ergonomiquement le clavier dans les jeux. L'index et l'annulaire sont utilisés pour réaliser les mouvements latéraux du personnage (*strafe*) et le majeur pour avancer/reculer. D'après les critiques du site web, le périphérique demande un court temps d'apprentissage et se révèle aussi performant et beaucoup moins fatiguant que le clavier.

Compte tenu de la facilité avec laquelle les joueurs utilisent le clavier, on peut émettre comme hypothèse que les degrés de liberté séparés peuvent être utilisés s'ils sont placés de manière intuitive les uns par rapport aux autres, au moins en ce qui concerne le contrôle de deux degrés de liberté.



FIG. 2.2 – FPS game - Ici Counter Strike

Dans un autre domaine, les télécommandes de voitures radio-commandées possèdent également des degrés de liberté séparés. Une main contrôle une manette qui a un mouvement gauche/droite pour orienter la voiture et l'autre main contrôle une manette qui a un mouvement d'avancer/reculer pour contrôler le sens d'avancement de la voiture.

Pour résumer, nous pouvons dresser une classification (fig. 2.3) dans laquelle nous associons le degré de séparation des degrés de liberté matériels [JSMM94] aux degrés de liberté manipulés dans l'environnement virtuel. Ainsi le Phantom possède six degrés de liberté isotoniques bi-directionnels intégrés qui permettent le contrôle de la position et de l'orientation d'un objet. Il en va de même pour la SpaceMouse dont les degrés de liberté ne sont pas isotoniques mais élastiques. Pour le clavier, utilisé dans les jeux FPS, les degrés de liberté unidirectionnels pris deux par deux permettent un contrôle séparé des deux degrés de liberté intégrés permettant le déplacement du personnage. La radio-commande a, quant à elle, deux degrés de libertés séparés bi-directionnels qui permettent le contrôle de deux degrés de liberté intégrés, qui sont ceux de la voiture.

Pour le DigiHaptic, nous cherchons à utiliser trois degrés de liberté bi-directionnels isotoniques ou élastiques séparés pour contrôler trois degrés de liberté assemblés dans l'environnement virtuel, tout en les plaçant intuitivement les uns par rapport aux autres : nous cherchons à avoir l'isomorphisme d'orientation et de direction définis au §1.2.6.

## 2.2.4 Affordances

Un paramètre important à prendre en considération lors de la conception d'un périphérique est la notion d'affordance.

Les affordances sont les propriétés relatives à l'interaction entre les animaux et l'environnement qui déterminent le comportement d'un individu [Gib79][SGF99]. Ces interactions sont perçues par le sujet plus selon l'idée qu'il se fait de la fonction de l'objet que de la perception des caractéristiques physiques, géométriques, etc., de l'objet ou de l'environnement. Néanmoins, les affordances ne sont pas de pures constructions psychologiques mais de réelles relations entre les propriétés des animaux et celles de l'environnement. Elles relèvent certes de la subjectivité mais sont aussi raisonnées et liées aux expériences antérieures du sujet. Un jour d'hiver, l'aptitude d'un animal à rester sur une mare gelée sera déterminée en partie par le poids de l'animal (la force qu'il applique par unité de surface sur la glace) relativement à l'épaisseur de la glace. La glace est assez épaisse

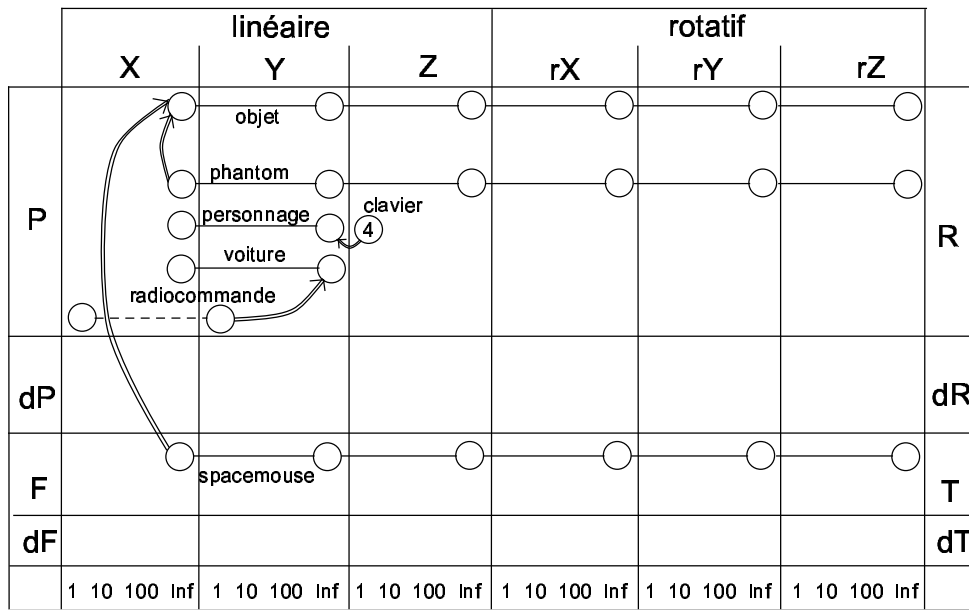


FIG. 2.3 – Classification de périphériques suivant la relation entre le degré de séparation de leurs degrés de liberté matériels et les degrés de liberté manipulés dans l’environnement virtuel (voir §1.2.1).

pour supporter certains animaux et trop fine pour d’autres. Par conséquent, la glace *afforde* la position debout pour certains animaux et pas pour d’autres. De même qu’une chaise *afforde* la position assise suivant la corpulence de la personne et la solidité perçue de la chaise. Les affordances d’un objet ou d’un matériau sont donc relatives à l’espèce et aux individus d’une espèce. Différentes substances de l’environnement ont différentes affordances pour la nutrition ou la fabrication. L’affordance d’une poignée de porte est d’appuyer dessus pour ouvrir la porte. Si on doit soulever la poignée pour ouvrir la porte, l’utilisateur va être perturbé.

Pour un périphérique, l’affordance passe par sa forme et son ergonomie. Elles doivent suggérer à l’individu des propriétés qui correspondent aux fonctions pour lesquelles le périphérique a été conçu. Pour prendre l’exemple du Phantom (§1.3.5), l’effecteur en forme de stylo conduit l’utilisateur à le saisir comme un stylo.

## 2.2.5 Fatigue

La fatigue de l’utilisateur est déterminante dans la conception d’un périphérique. Dans un premier temps, une étude ergonomique doit permettre de tenir compte des contraintes morphologiques des utilisateurs de manière à ne pas contraindre leurs mouvements naturels. Dans un second temps, des expérimentations avec des prototypes doivent permettre d’analyser les défauts ergonomiques et de les corriger. Pour évaluer la fatigue occasionnée par un périphérique, la norme ISO 9241-9 propose une série de questions permettant de mesurer la fatigue de chacun des membres sur une échelle de 1 à 7.

## 2.3 Réalisation de la maquette

Après avoir montré quels doigts utiliser pour contrôler chaque degré de liberté, nous verrons la réalisation de la maquette. Celle-ci consiste en la mise en oeuvre de la séparation des degrés de liberté avec des moyens simples et rapides, ce qui permet une première validation de l'idée et une réalisation plus rapide des étapes suivantes.

### 2.3.1 Choix des doigts

Nous avons associé les trois degrés de liberté au pouce, à l'index et à l'annulaire. Le pouce et l'index sont souvent utilisés dans la vie quotidienne, leur choix est immédiat. Nous avons ensuite choisi l'annulaire, plutôt que le majeur, en raison des chaînes musculaires communes à l'index et au majeur. Le majeur, statique, permet de découpler les mouvements de l'index et l'annulaire et peut être utilisé pour actionner un bouton de sélection.

Dans les applications de manipulation d'objets 3D, six ddl sont nécessaires : trois pour les translations et trois pour les rotations. Comme le périphérique possède trois ddl, l'utilisateur ne peut manipuler que trois des six ddl à la fois. Pour rester simple, nous utiliserons les translations uniquement ou les rotations uniquement et le passage de l'un à l'autre se fera, par exemple, à l'aide d'un bouton au niveau du majeur. Ce changement de mode sera étudié dans le chapitre suivant.

Dans les tâches de manipulation en mode translation, le pouce a un mouvement gauche/droite, aussi l'avons nous associé à la translation d'objets suivant la largeur de l'écran. L'index à un mouvement d'avant/arrière, aussi l'avons nous associé à la translation d'objets suivant la profondeur de l'écran. Quant à l'annulaire qui a un mouvement haut/bas, il est associé à la translation d'objets suivant la hauteur de l'écran. Ainsi, pour les translations, il y a isomorphisme d'orientation et de direction pour rendre l'utilisation de l'interface intuitive (§1.2.6). En mode rotation, le pouce contrôle le roulis<sup>14</sup>, l'index le tangage et l'annulaire le lacet (fig. 2.4). Les rotations associées au pouce et l'index sont isomorphiques en orientation, direction et déplacement car l'axe de rotation des manettes correspond à celui des objets. Par contre la rotation associée à l'annulaire n'est pas isomorphique. Elle ne possède que l'isomorphisme de déplacement et demande donc un apprentissage.

### 2.3.2 Ergonomie

Chaque doigt manipule donc sa propre manette selon les mouvements précédemment décrits. Pour réaliser un périphérique intuitif et non-fatigant, il faut positionner les manettes les unes par rapport aux autres en respectant le mouvement naturel des doigts, quelque soit la taille de la main. Les manettes sont les pièces mécaniques en contact avec les doigts et en rotation autour d'un axe. Pour répondre à ces contraintes ergonomiques,

---

<sup>14</sup>Soit un repère cartésien (XYZ) avec l'axe X orienté vers la droite et l'axe Y orienté vers le haut, le roulis correspond à la rotation autour de Z, le tangage à la rotation autour de X et le lacet à la rotation autour de Y.

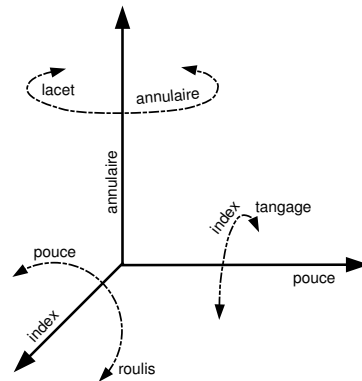


FIG. 2.4 – Correspondance entre le déplacement des doigts et celui des objets pour les translations et rotations en 3D

nous avons travaillé avec une maquette test dont nous avons réglé la position des manettes en considérant différentes morphologies de mains.

La maquette a ensuite servi à modéliser la coque sous Catia<sup>®</sup>. Nous avons pour cela mesuré les positions des manettes que nous avons reportées dans Catia<sup>®</sup>, puis nous avons joué sur leur orientation et celle des moteurs pour respecter l'ergonomie de la main. Enfin, le modèle informatique a permis d'élaborer la coque en stéréolithographie. Ce procédé consiste à polymériser couche à couche une résine photo sensible avec un faisceau laser ultra-violet.

## 2.4 Réalisation du prototype fonctionnel

La réalisation du prototype fonctionnel est la mise en oeuvre de la maquette avec les moyens technologiques appropriés.

Pour la commande du dispositif, nous avons choisi une commande en impédance (voir §1.4) pour la simplicité de mise en oeuvre. Dans cette commande (fig. 2.5), les positions des doigts ( $\alpha_1, \alpha_2, \alpha_3$ ) servent à la mise à jour de la position de l'objet contrôlé dans l'environnement virtuel. Une détection de collisions est ensuite réalisée entre cet objet et les autres objets de l'environnement, ce qui permet le cas échéant de calculer le retour de force à envoyer au périphérique. A cette force de référence, on associe trois forces de référence ( $F_{1ref}, F_{2ref}, F_{3ref}$ ) à exercer à l'extrémité de chacun des doigts. Nous verrons plus loin comment celles-ci sont calculées à partir de la force de référence suivant l'application. Ces forces de référence servent à la détermination des couples de référence ( $C_{1ref}, C_{2ref}, C_{3ref}$ ), calculés à l'aide du modèle mécanique des manettes et du réducteur, à exercer par chacun des moteurs. Enfin le modèle électro-mécanique permet de calculer des courants nécessaires pour obtenir des couples sur les manettes ( $C_1, C_2, C_3$ ) correspondant aux couples de référence. Cette dernière opération s'effectue en boucle ouverte par souci de simplicité.

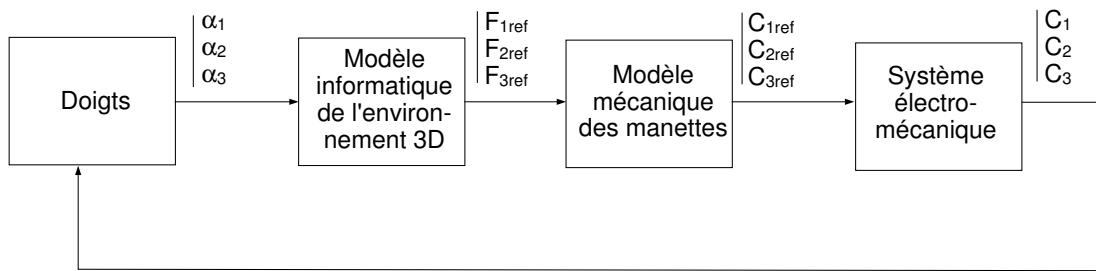


FIG. 2.5 – Schéma général de commande du dispositif

### 2.4.1 Conception mécanique

#### Etude du Paddle

Avant de réaliser le DigiHaptic, nous avons fabriqué un manche à retour de force appelé *Paddle* (fig.2.6) dont on peut trouver la description complète, comprenant le dessin des pièces et l'électronique d'alimentation dans l'article [ROC97].

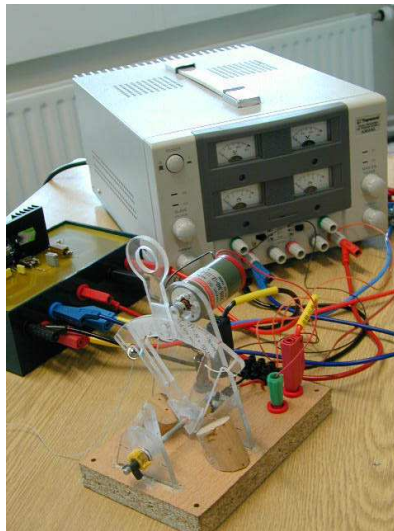


FIG. 2.6 – Le paddle et son électronique.

Le Paddle est capable de fournir une force à son extrémité d'environ 8N. Son étude nous a servi de point de départ dans la conception du DigiHaptic.

#### Rayon des manettes

Le rayon des manettes a été choisi en relation avec l'amplitude maximale de déplacement de chaque doigt lorsque la paume de la main est posée, soit environ 40 mm. Pour cette amplitude, différents rayons sont envisageables. Si nous prenons un rayon important, l'angle relatif au centre de rotation est petit, ce qui permet d'avoir le plus de résolution possible sur la mesure de cet angle. Cependant, plus le rayon de la manette est important, plus la force appliquée à son extrémité sera faible pour un couple donné. Le choix du rayon va donc dépendre de la valeur maximale de cette force.

## Force maximale

Pour déterminer la force maximale à transmettre au doigt de l'utilisateur, nous avons considéré le cas de la simulation d'un mur virtuel, modélisé par un ressort qui possède une caractéristique d'effort saturé comme décrit au §1.4 figure 1.18. Le paramètre important est ici la force maximale à renvoyer au niveau du doigt. Après avoir réalisé plusieurs essais empiriques avec le Paddle, nous avons constaté qu'une force maximale de 2 N pour un doigt était suffisante pour ne pas avoir la sensation de traverser le mur virtuel même si l'utilisateur peut facilement exercer des forces supérieures de l'ordre de 5 à 10 N. Une force maximale de 2 N permet également de rendre l'interface plus sûre pour l'utilisateur (on ne risque pas de blesser l'utilisateur). Nous pouvons noter par ailleurs que l'utilisation des doigts permet également de diminuer la force maximale à renvoyer pour simuler un mur par rapport aux périphériques à ddl assemblés qui renvoient une force sur la main<sup>15</sup>. Connaissant la force maximale à appliquer à l'extrémité de la manette, nous devons à présent étudier la transmission du couple qui va déterminer le rapport de réduction.

## Transmission du couple

Le lien entre le moteur et la manette est assuré par un câble d'acier enroulé autour de la poulie fixée sur l'arbre moteur (fig. 2.7). Le rapport de réduction de vitesse correspond au rapport des rayons de la partie inférieure de la manette et de la poulie. La transmission de couple par câble est la solution la plus souvent adoptée pour les périphériques à retour de force. Elle évite de ressentir le passage d'une dent à l'autre des mécanismes à engrenages. Afin de minimiser les frottements secs et autres comportements non-linéaires, le câble choisi doit être très flexible (faible rayon de courbure), inextensible, non glissant et résistant. Par ailleurs, le diamètre de la poulie doit être au minimum égal à 20 fois le diamètre du câble pour éviter les effets de mémoire de forme. Nous avons choisi un câble inox gainé de polyamide ayant un diamètre interne de 0,75 mm et un diamètre externe de 1,00 mm avec une construction en toron de  $7 \times 19$  de la société CarlStahl [Car]. Ce câble, qui peut résister à des forces de 417N, n'est pas 20 fois plus petit que le diamètre de la poulie de 6 mm mais s'est avéré être le meilleur compromis trouvé pour l'application (en effet, plus le rayon de la poulie est petit, plus le rapport de réduction est important).

Pour lire la position angulaire de la manette, nous avons choisi un potentiomètre ayant un faible couple de frottement, une grande précision et une bonne linéarité. Notre choix s'est porté sur un potentiomètre Spectrol 10 K $\Omega$  avec un très faible couple de frottement sec et une très bonne linéarité. Le potentiomètre permet de lire la position angulaire absolue de la manette contrairement aux codeurs optiques. Il évite aussi la perte d'information en cours de fonctionnement et le calibrage du périphérique en début d'utilisation. Nous avons placé le potentiomètre sur l'axe de la manette plutôt que sur l'axe du moteur, tout d'abord pour la simplicité de réalisation et ensuite pour éviter de multiplier les frottements du potentiomètre par le rapport de réduction. Placer le potentiomètre sur l'axe du moteur aurait cependant eu l'avantage de multiplier la résolution de mesure de l'angle par le rapport de réduction. Nous n'avons pas choisi d'encodeurs pour plusieurs raisons : d'abord ils ne permettent pas de lire une position absolue, ensuite il ont une résolution

---

<sup>15</sup>La force maximale pour le PHANToM Desktop est par exemple de 8,5 N



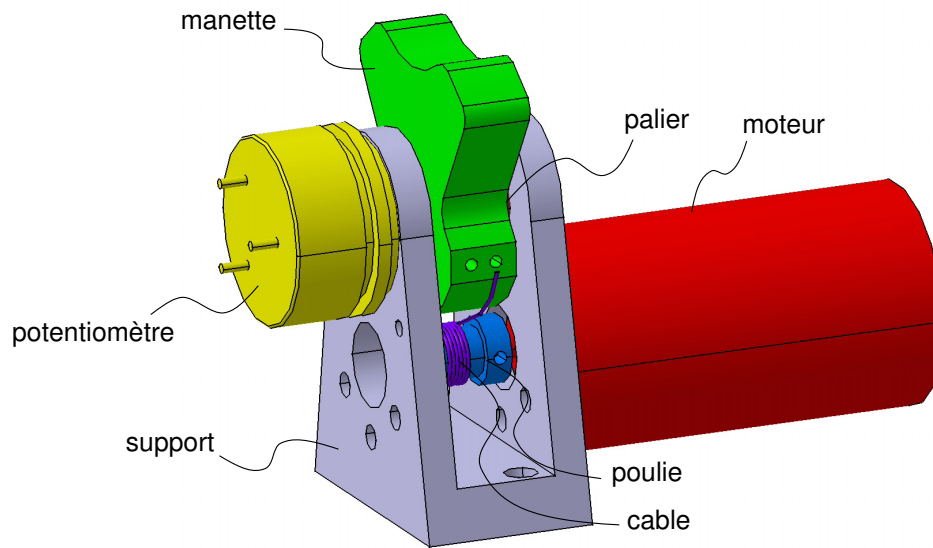


FIG. 2.7 – Modèle Catia d'un ensemble élémentaire du DigiHaptic

limitée (en général 2000 points par tour) et enfin ils sont bien plus chers et encombrants que les potentiomètres.

Le support a été usiné en un bloc afin d'assurer la coaxialité entre les deux paliers, supports de l'axe moteur. Le dernier élément à choisir est le moteur et son couple maximal.

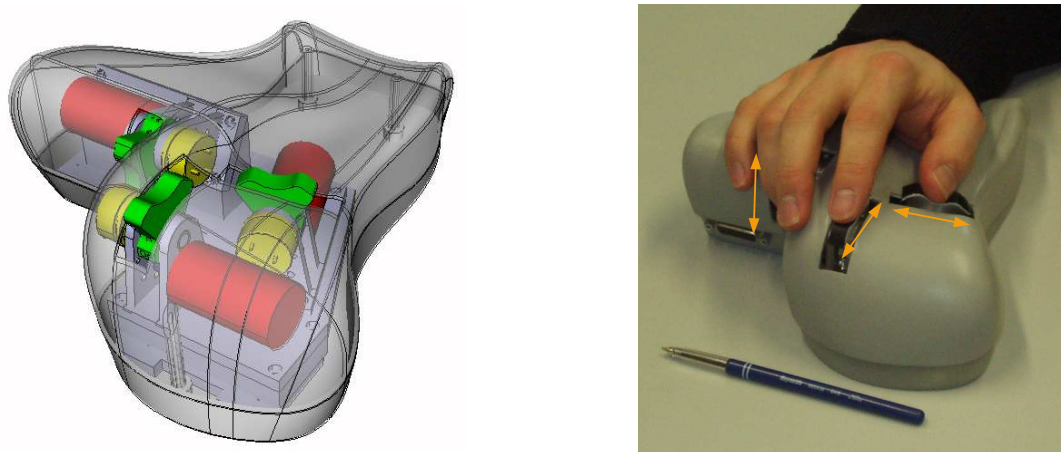


FIG. 2.8 – Le DigiHaptic avec les trois manettes et sa coque. A gauche le modèle Catia et à droite le prototype fonctionnel

### Choix des moteurs

Le premier critère de choix du moteur est le couple maximal qu'il peut fournir, puis ce sont ses qualités mécaniques et son encombrement qui entrent en jeu.

Dans les applications de retour d'effort, les moteurs travaillent à vitesse quasi nulle. Pour cette raison, l'absence de couple reluctant et la qualité des contacts glissants sont des propriétés mécaniques particulièrement importantes.

Dans la gamme commerciale disponible, nous avons choisi un moteur à courant continu Maxon [Max] (tab. 2.1) 4W, 6V pour son couple permanent maximal de  $11,3 \cdot 10^{-3} N.m$  et ses qualités mécaniques. En effet, c'est un moteur cloche à rotor sans fer qui n'a donc pas de couple reluctant. Par ailleurs, les contacts mécaniques sont faits en métaux précieux qui garantissent la propreté de passage d'une lame de collecteur à l'autre.

Nous aurions pu sous-dimensionner les moteurs compte tenu de leur utilisation en intermittence mais nous avons privilégié la sûreté de fonctionnement en respectant les caractéristiques du moteur.

TAB. 2.1 – Résumé des caractéristiques du moteur

Courant permanent max	1,07 A
Couple permanent max	$11,3 \cdot 10^{-3} N.m$
Inertie du rotor	$8,67 \cdot 10^{-7} g.m^2$

### Choix définitif du rayon des manettes

Finalement, le rayon des manettes ( $r$ ) est donné par la relation 2.1 où  $C_{motmax}$  est le couple maximal du moteur et  $N$  le rapport de réduction. Compte tenu de l'encombrement de la mécanique, nous avons fixé le rayon inférieur de la manette à  $12mm$ , ce qui donne un rapport de réduction de 4. Le rayon d'une manette vaut donc  $2cm$ .

$$r = \frac{C_{motmax} N}{F_{sat}} \quad (2.1)$$

### 2.4.2 Conception électronique

Par "conception électronique", on entend l'ensemble des moyens informatiques et électroniques qui permettent la connexion du périphérique à l'ordinateur gérant l'environnement virtuel.

Le contrôle d'un dispositif haptique doit se faire à une fréquence supérieure ou égale à 1KHz pour éviter les problèmes d'instabilités (voir §1.4). Comme Windows n'est pas un système d'exploitation temps réel, nous avons choisi QNX qui est un système d'exploitation dérivé d'Unix et orienté temps réel [QNX]. Sur cette machine opérant le système d'exploitation QNX (machine QNX) (fig. 2.9), une carte d'entrées-sorties analogiques numériques Servotogo [Ser] assure la liaison avec la carte de puissance des moteurs et la carte de conditionnement des signaux des potentiomètres. Les entrées sorties analogiques numériques de la carte ont une résolution de 13 bits. Ainsi les positions angulaires sont mesurées avec une précision de  $0.02^\circ$  ( $0.06^\circ$  en pratique à cause des bruits de mesure).

L'environnement virtuel 3D est géré par une seconde machine opérant le système d'exploitation Windows (machine Windows) et la communication entre les deux machines

est assurée en ethernet en utilisant le protocole UDP<sup>16</sup>. Nous avons choisi ce protocole non sûr car la perte d'informations n'a pas d'importance à la fréquence à laquelle nous envoyons les données (1KHz). Le choix du protocole TCP aurait pu entraîner des latences importantes lors de la réexpédition des données perdues. Par ailleurs, le choix de la liaison réseau a aussi l'avantage de rendre possible la connexion du DigiHaptic à n'importe quel système d'exploitation.

Sur la machine Windows, la mise à jour de l'affichage est faite à 25Hz alors que les gestions de collisions et calculs de forces se font à 1KHz. Une API bas niveau permet d'intégrer facilement le périphérique dans une application.

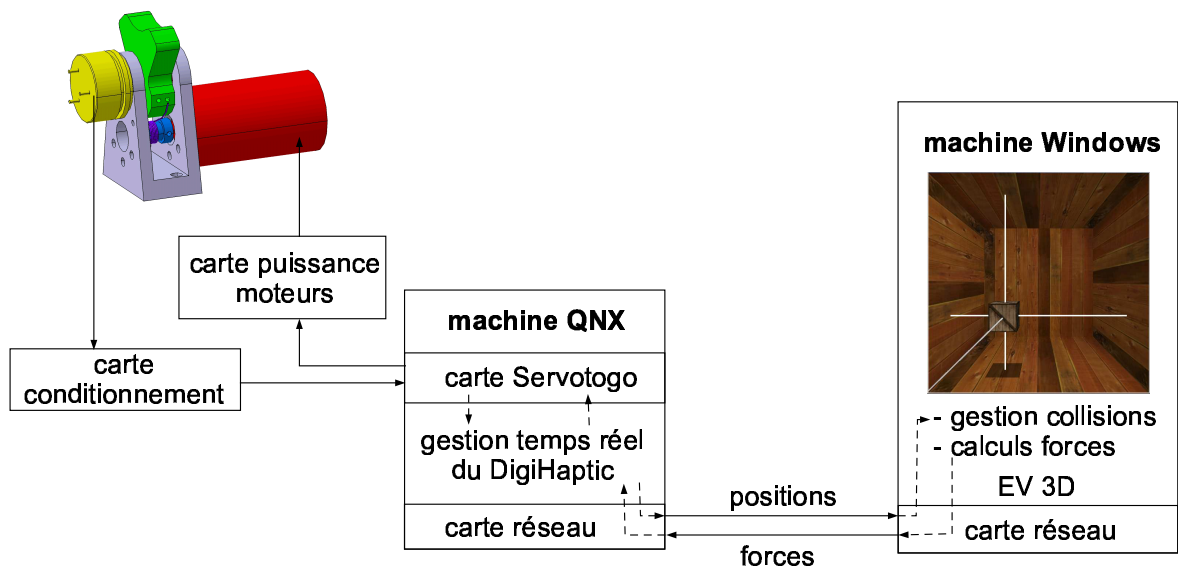


FIG. 2.9 – Représentation des connexions entre les différentes parties matérielles.

### Boucle de contrôle sous QNX

Le programme de contrôle sur la machine QNX est divisé en une boucle principale (fig. 2.10) qui s'exécute à une fréquence fixe de 1KHz et deux threads : l'un réceptionne les consignes de forces et l'autre les réglages des manettes en fonction du mode pré-programmé choisi. Le programme contrôle les tensions aux bornes de la carte de puissance des moteurs, par l'intermédiaire de la carte Servotogo.

### API Windows/Linux

L'API Windows/Linux écrite en C++ est une couche bas niveau qui permet l'intégration facile du périphérique dans tout programme. Elle inclut une classe gérant l'initialisation du périphérique, la réception des positions des manettes et l'envoi des forces (fig. 2.11) mais aussi une classe gérant les comportements pré-programmés sur la machine QNX tels que la simulation de ressorts (fig. 2.11).

<sup>16</sup>Le Virtuose 6D commercialisé par la société Haption utilise le même principe [Hapb]

```

tant que () {
    Attente du top horloge de la carte Servotogo (chaque milliseconde)
    Lecture des entrées analogiques de la carte Servotogo
    Calcul des positions angulaires des manettes
    Filtrage des positions
    Calcul des courants moteurs en fonction du mode d'utilisation choisi
    Envoi des positions des manettes sur le réseau
    Envoi des consignes de courant à la carte Servotogo
}
    
```

FIG. 2.10 – Algorithme de la boucle de contrôle

<b>DigiHaptic</b>			
manette: Lever forcesmanettes: LeverForce machine: char[20]		<b>DigiSettings</b>	
DigiHaptic(machine: char*) Init(machine: char*) GetPosVit(positions: Lever, mode: char*) SendForce(forces: LeverForce)		manset: LeverSetting machine: char[20]	
		DigiSettings(machine: char*) Init(machine: char*) SendSettings(réglages: LeverSetting)	

FIG. 2.11 – Classes de gestion du DigiHaptic

## Solution DSP

La mise en oeuvre avec la machine QNX et la carte Servotogo nous offre une solution robuste et fiable permettant de faire du développement rapide. Cette solution suppose l'utilisation d'un PC et d'une carte Servotogo. Dans l'optique de développer un prototype industriel, pour diminuer l'encombrement et obtenir de meilleures performances, nous avons essayé de remplacer la machine QNX par un DSP connecté en USB avec la machine Windows.

La mise en oeuvre a révélé un problème de latence aléatoire sur la ligne USB lors de l'envoi des forces sur le DSP et rend donc impossible l'utilisation de cette association.

Une utilisation de la future génération de PIC de Microchip[Mica] en USB2.0 pourrait résoudre ce problème. La dernière version du Delta HAPTIC device de Force Dimension et la dernière version du SPIDAR fonctionnent ainsi en USB 2.0. A défaut, il faudrait s'orienter vers une liaison de type FireWire, utilisée par la dernière version du Phantom (Phantom Omni) de Sensable.

## La carte de puissance des moteurs

Bien que moins compacts que les amplificateurs de courant fonctionnant en modulation de largeur d'impulsion (MLI) et dissipant plus de chaleur, les amplificateurs linéaires permettent une conversion fidèle de la tension d'entrée en couple moteur en s'affranchissant des bruits hautes fréquences associés aux amplificateurs MLI.

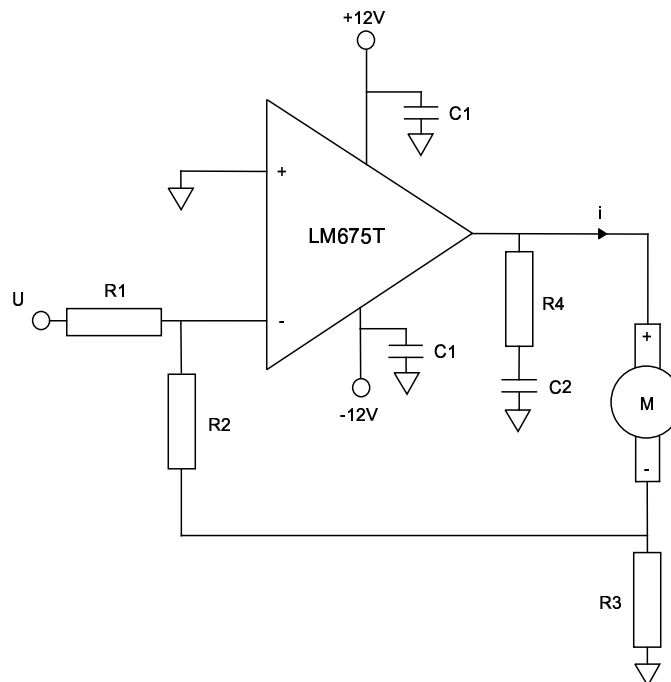


FIG. 2.12 – Schéma des amplificateurs de courant pour les moteurs

La figure 2.12 représente le schéma électronique de commande d'un moteur. Le principe est d'utiliser un amplificateur opérationnel en source de courant pour asservir le courant

qui passe dans le moteur M. Ce courant ( $i$ ) est alors proportionnel à la tension de consigne  $U$  :

$$i = - \left( 1 + \frac{R2}{R3} \right) \frac{U}{R1} \quad (2.2)$$

$R1$  doit être choisi le plus grand possible pour limiter le courant débité par le générateur de tension de consigne. Les résistances  $R2$  et  $R3$  sont ensuite choisies pour avoir le courant maximal quand  $U$  est maximal. Nous avons finalement choisi  $R1=10K\Omega$ ,  $R2=328\Omega$  et  $R3=0,33\Omega$  ( $R3$  est une résistance de puissance.  $C1$ ,  $C2$  et  $R4$  servent au filtrage).

### Les conditionneurs

Les conditionneurs permettent d'adapter la plage de tensions en sortie des potentiomètres à celle tolérée par la carte d'entrées-sorties analogiques de façon à mesurer la position angulaire de la manette avec le plus de précision possible.

Nous avons mis en place un montage multiplicateur soustracteur pour adapter le niveau de tension, associé à un montage suiveur pour assurer l'adaptation d'impédance au niveau de l'entrée de la carte d'entrées-sorties analogiques numériques (fig. 2.13).

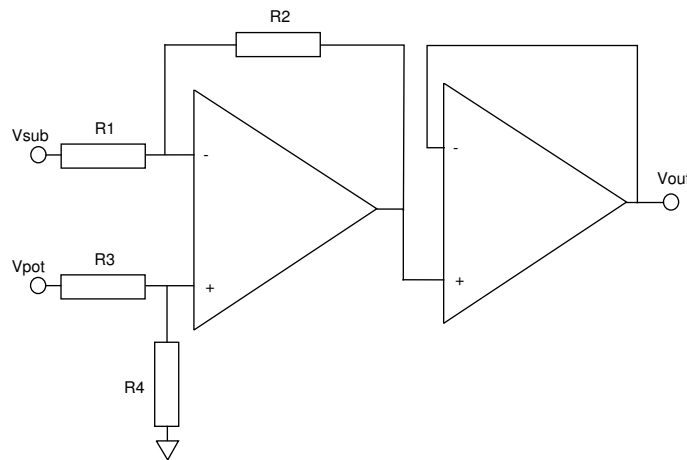


FIG. 2.13 – Schéma du conditionneur pour un potentiomètre

En prenant  $\frac{R1}{R2} = \frac{R3}{R4}$ , on obtient :

$$V_{out} = \frac{R2}{R1} (V_{sub} - V_{pot}) \quad (2.3)$$

Le principe est de ramener la tension de sortie du potentiomètre symétriquement autour de 0V, puis de multiplier par le coefficient adéquat pour obtenir une tension variant entre -10V et 10V.

La figure 2.14 montre les différents éléments électroniques assemblés dans le boîtier.

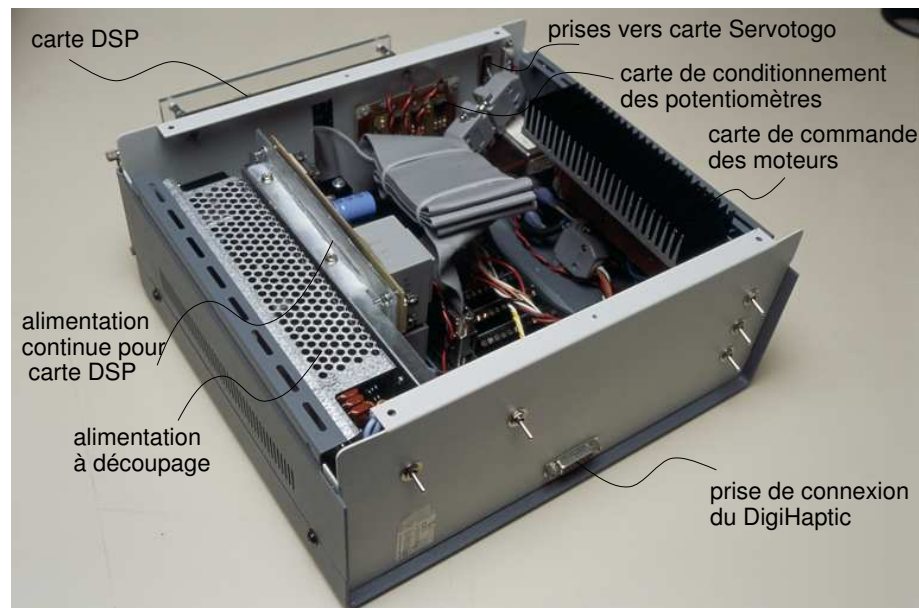


FIG. 2.14 – L'ensemble de l'électronique.

## 2.5 Identification des paramètres du système

Le "système" correspond à un ensemble élémentaire du DigiHaptic (fig. 2.7) comprenant la manette, le potentiomètre, les paliers, la poulie, le câble et le moteur. Ce système doit être identifié afin de déterminer ses performances en retour d'effort. Pour modéliser le système et en définir les paramètres, nous avons à chaque fois comparé les courbes expérimentales et théoriques de réponse indicielle en position. Une fois la correspondance globalement atteinte, nous considérons que le modèle est représentatif du phénomène réel.

L'identification du système a été faite en boucle fermée pour assurer la stabilité et éviter que la manette ne tape en bout de course et endommage le câble.

Nous commençons par une modélisation simple du système (fig. 2.15) en ne prenant en compte que le moment d'inertie global des parties tournantes ( $J$ ),  $\theta$  étant la position angulaire de la manette en radians<sup>17</sup>,  $K_\theta$  le correcteur proportionnel,  $K$  le coefficient qui lie la tension d'entrée de la carte de commande au couple s'exerçant sur la manette. Dans ce modèle comme dans ceux qui vont suivre, nous avons négligé la masse de la manette par rapport à l'inertie globale. Nous avons montré que cette hypothèse est valide en simulant le système avec et sans prise en compte de la masse. Nous obtenons une différence négligeable entre deux courbes de position obtenues.

Pour déterminer  $K$ , nous considérons que le courant du moteur  $i$  est proportionnel à  $U$  par la relation (2.4), déduite de l'expression (2.2). Par ailleurs, le couple développé par le moteur  $C_{mot}$  est proportionnel au courant de l'induit  $i$  (2.5) et le couple de la manette  $C$  est proportionnel au couple moteur suivant le rapport de réduction (2.6). Compte tenu des valeurs paramétriques, on obtient  $K = 4,18 \cdot 10^{-3} N.m.V^{-1}$ .

<sup>17</sup>La position angulaire est mesurée par rapport à la verticale

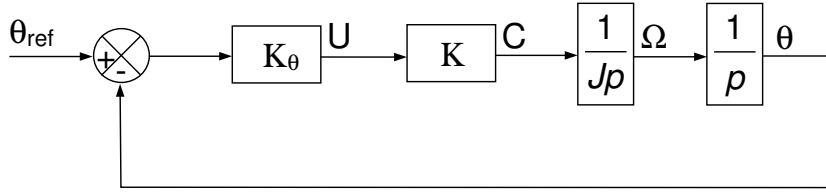


FIG. 2.15 – Modèle de base du système

$$i = \frac{1}{R1} \left( 1 + \frac{R2}{R3} \right) U \quad (2.4)$$

$$C_{mot} = k_i i \text{ avec } k_i = 10,5 \cdot 10^{-3} \quad (2.5)$$

$$C = N C_{mot} \text{ avec } N = 4 \quad (2.6)$$

$$C = K U \quad (2.7)$$

### 2.5.1 Détermination de l'inertie de la manette J

Nous pouvons écrire, à partir du schéma-blocs 2.15, l'équation différentielle correspondante (2.8). Celle-ci se résout en donnant (2.10) et (2.11) après avoir posé (2.9) et avec comme conditions initiales  $\theta(0) = 0$  et  $\dot{\theta}(0) = 0$ . Nous pouvons alors déduire que l'instant du premier extremum  $t_1$  est égal à  $\pi/\omega_0$ , temps pour lequel  $\dot{\theta}(t)$  s'annule. Finalement, nous en déduisons l'écriture de  $J$  (2.12).

$$\frac{J}{K K_\theta} \ddot{\theta}(t) + \theta(t) = \theta_{ref} \quad (2.8)$$

$$\frac{1}{\omega_0^2} = \frac{J}{K K_\theta} \quad (2.9)$$

$$\theta(t) = \theta_{ref} (-\cos(\omega_0 t) + 1) \quad (2.10)$$

$$\dot{\theta}(t) = \theta_{ref} \omega_0 \sin(\omega_0 t) \quad (2.11)$$

$$J = \frac{K K_\theta t_1^2}{\pi^2} \quad (2.12)$$

La figure 2.16 nous montre un relevé expérimental, correspondant à une réponse indicielle en position, à partir duquel nous pouvons déterminer  $J$  qui vaut environ  $1.4 \cdot 10^{-5} \text{ kg.m}^2$ .

En trait discontinu, est affiché le résultat de la simulation pour les valeurs correspondantes. Comme on pouvait s'y attendre, le modèle ne tenant compte d'aucun amortissement, la réponse du système simulé est fortement oscillante. Pour améliorer le modèle, nous nous proposons donc de déterminer les frottements secs et visqueux.

### 2.5.2 Détermination des frottements secs et visqueux

Nous reprenons pour cela le schéma 2.15 dans lequel nous ajoutons un bloc de frottements secs et visqueux (fig. 2.17) dont la sortie est égale à  $\text{sign}(\dot{\theta}(t)) (C_v |\dot{\theta}(t)| + C_r)$  où  $C_v$



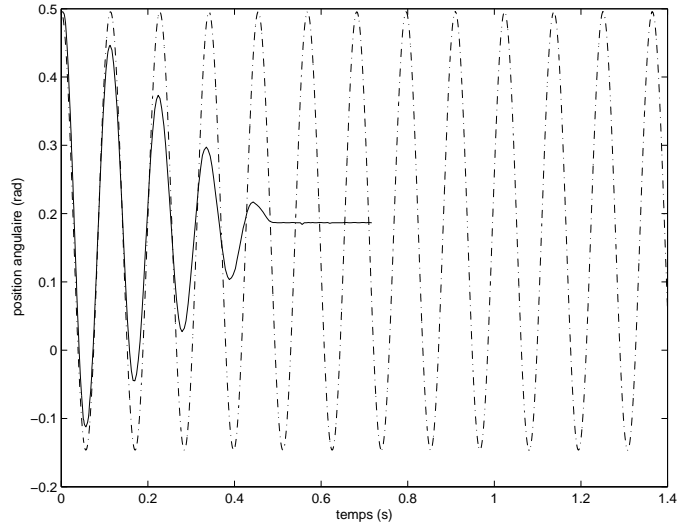


FIG. 2.16 – Position angulaire au cours du temps avec  $K_\theta = 10.0$  pour un échelon indiciel quand  $\theta_{ref}$  passe de  $0,5 \text{ rad}$  à  $0,17 \text{ rad}$  ( $10^\circ$ ). Le trait continu représente l'essai expérimental et le trait discontinu, la simulation sous Simulink.

est le coefficient de frottements visqueux et  $C_r$  le couple de frottements secs. En écrivant l'équation différentielle correspondante (2.13) que l'on peut mettre sous la forme (2.14) avec  $\omega_0^2 = K K_\theta / J$  et  $\lambda = C_v / (2 * J)$ , on trouve une solution de la forme (2.15) avec  $A$  et  $\phi$  qui restent à déterminer en fonction des conditions initiales. En prenant  $\theta(0) = 0$  et  $\dot{\theta}(0) = 0$ , on obtient (2.16) et (2.17). La solution correspond à celle trouvée en supposant que le discriminant de l'équation homogène associée est négatif. Hypothèse justifiée si l'on considère que les frottements sont faibles ( $\lambda < \omega_0$ ).

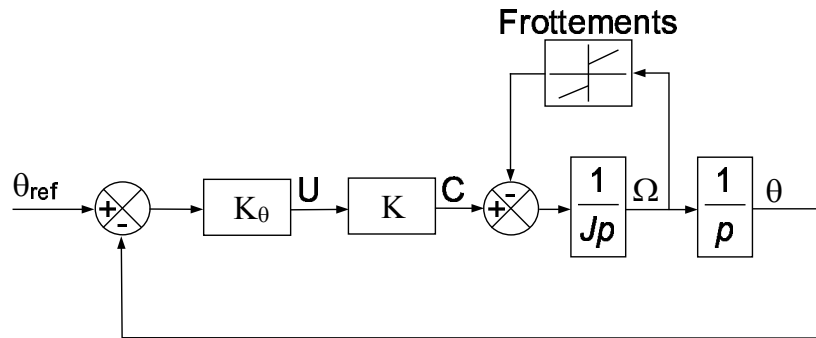


FIG. 2.17 – Modèle du système avec prise en compte des frottements secs et visqueux.

$$J \ddot{\theta}(t) + C_v \dot{\theta}(t) + K K_\theta \theta(t) = K K_\theta \theta_{ref} - C_r \quad (2.13)$$

$$\ddot{\theta}(t) + 2 \lambda \dot{\theta}(t) + \omega_0^2 \theta(t) = \omega_0^2 \theta_{ref} - \frac{C_r}{J} \quad (2.14)$$

$$\theta(t) = \left( \theta_{ref} - \frac{C_r}{J \omega_0^2} \right) \left( A e^{-\lambda t} \cos(\sqrt{\omega_0^2 - \lambda^2} t + \phi) + 1 \right) \quad (2.15)$$

$$\phi = -\arctan\left(\frac{\lambda}{\sqrt{\omega_0^2 - \lambda^2}}\right) \quad (2.16)$$

$$A = -\sqrt{\frac{\omega_0^2}{\omega_0^2 - \lambda^2}} \quad (2.17)$$

L'instant de premier extrémum  $t_1$  est celui pour lequel  $\dot{\theta}(t)$  s'annule. En dérivant 2.15, on en déduit alors la plus petite valeur de  $t$  qui annule cette fonction (2.18). Cela nous permet de calculer  $\lambda$  (2.19) et ensuite de calculer  $C_v$  (2.20).

$$t_1 = \frac{\pi}{\sqrt{\omega_0^2 - \lambda^2}} \quad (2.18)$$

$$\lambda = \sqrt{\omega_0^2 - \left(\frac{\pi}{t_1}\right)^2} \quad (2.19)$$

$$C_v = 2\lambda J \quad (2.20)$$

$C_r$  se calcule à partir de la valeur de  $\theta(t)$  quand  $t = t_1$  (2.21).

$$C_r = \frac{J\omega_0^2 \left( \theta_{ref} \sqrt{-\frac{\omega_0^2}{-\omega_0^2 + \lambda^2}} e^{-\lambda t_1} \cos\left(-\sqrt{\omega_0^2 - \lambda^2} t_1 + \arctan\left(\frac{\lambda}{\sqrt{\omega_0^2 - \lambda^2}}\right)\right) - \theta_{ref} + \theta(t_1) \right)}{\sqrt{-\frac{\omega_0^2}{-\omega_0^2 + \lambda^2}} e^{-\lambda t_1} \cos\left(-\sqrt{\omega_0^2 - \lambda^2} t_1 + \arctan\left(\frac{\lambda}{\sqrt{\omega_0^2 - \lambda^2}}\right)\right) - 1} \quad (2.21)$$

En reprenant l'essai de la figure 2.16, nous trouvons une valeur de  $C_v = 6 \cdot 10^{-6} N.m.rad^{-1}.s$  et  $C_r = 6 \cdot 10^{-4} N.m$ . Avec ces valeurs nous obtenons la simulation de la figure 2.18 qui est tout à fait satisfaisante.

Massie recommande un rapport minimal de 30 :1 entre le couple maximal et le couple de frottement sec [Mas93]. Nous obtenons un rapport 75 :1 qui respecte ce critère.

Cependant, nous avons remarqué que les valeurs de frottement secs et visqueux ne sont pas constantes au cours des essais (elles peuvent varier du simple au double). C'est ce que l'on constate en effectuant des réponses indicielles d'amplitudes différentes. Nous n'avons pas identifié les causes de ce phénomène mais nous supposons que la mémoire de forme du câble peut y contribuer. Nous avons cependant obtenu un ordre de grandeur des frottements et nous pouvons conclure que le frottement visqueux est négligeable devant le frottement sec (rapport 1 :100).

## 2.6 Commande du système

Maintenant que nous avons identifié les principaux paramètres de notre système, nous allons en réaliser la commande. Dans un premier temps, nous détaillons la méthode de filtrage de la vitesse dont nous avons besoin pour la suite. Nous réalisons ensuite la commande en position du système, qui se révèle particulièrement intéressante pour le calcul

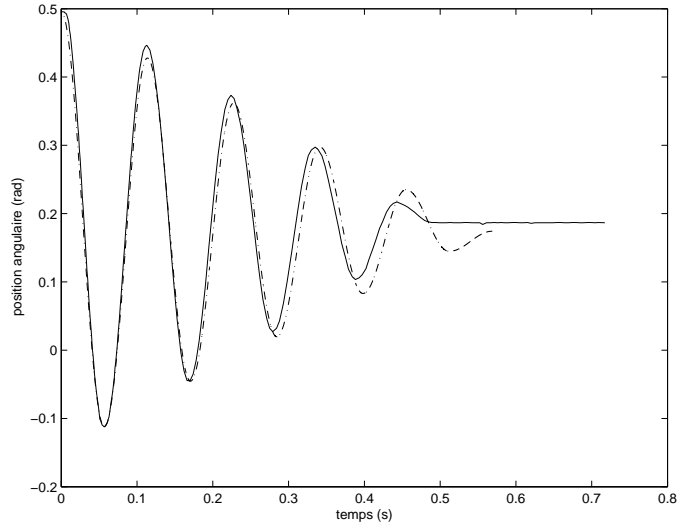


FIG. 2.18 – Position angulaire au cours du temps avec  $K_\theta = 10.0$  pour un échelon indiciel quand  $\theta_{ref}$  passe de  $0,5 \text{ rad}$  à  $0,17 \text{ rad}$  ( $10^\circ$ ). Le trait continu représente l'essai expérimental et le trait discontinu, la simulation sous Simulink.

des paramètres optimaux d'un mur virtuel. Enfin, nous montrons les différents modes de commande en impédance et nous détaillons la commande en mode élastique.

### 2.6.1 Filtrage de la vitesse

Pour la suite, nous avons besoin de la vitesse angulaire de la manette ( $\Omega$ ) qui n'est pas directement mesurée. Nous devons donc la calculer à partir des variations de position angulaires. Pour cela, nous mettons en œuvre un filtre numérique passe-bas d'ordre 1 (2.22) ayant une constante de temps ( $\tau$ ) que nous prenons au minimum dix fois supérieure à la période d'échantillonnage du système ( $Te = 1 \text{ ms}$ ) afin de garder une approche pseudo-continue du système. La fréquence de coupure correspondante vaut donc  $1/(2\pi\tau) = 16 \text{ Hz}$ . Nous n'appliquons pas de filtrage de position car celle-ci est peu bruitée. La figure 2.19 montre les vitesses angulaires filtrée et non filtrée.

$$\Omega_n = \left( \frac{\theta_n - \theta_{n-1}}{Te} + \frac{\tau}{Te} \Omega_{n-1} \right) \frac{1}{1 + \frac{\tau}{Te}} \quad (2.22)$$

Le déphasage induit par le filtrage est relativement faible compte tenu des basses fréquences de déplacement de la manette. Le déphasage observé sur la figure 2.19 a été obtenu pour des mouvements de manettes qui peuvent être considérés « haute fréquence ». Par ailleurs, nous ne pouvons pas diminuer le déphasage car il faudrait prendre une valeur plus petite pour  $\tau$  et nous avons déjà pris le plus petit possible pour garder une approche pseudo-continue du système.

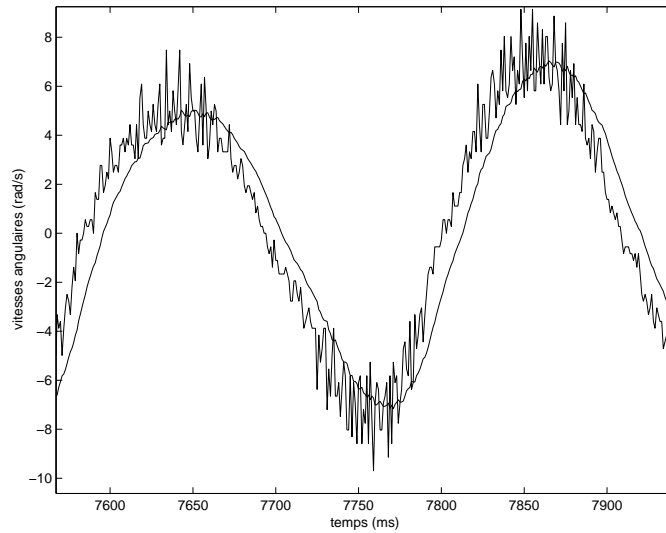


FIG. 2.19 – Vitesses angulaires filtrée et non filtrée

### 2.6.2 Commande en position

Nous allons maintenant mettre en œuvre une commande en position de la manette avec retour de vitesse qui permet de faire une régulation dans l'espace d'état avec un réglage proportionnel (fig. 2.20). Cette commande est particulièrement intéressante puisqu'elle nous permettra de régler les coefficients de raideur et d'amortissement lors de la création d'un mur virtuel, comme nous l'expliquerons plus loin.

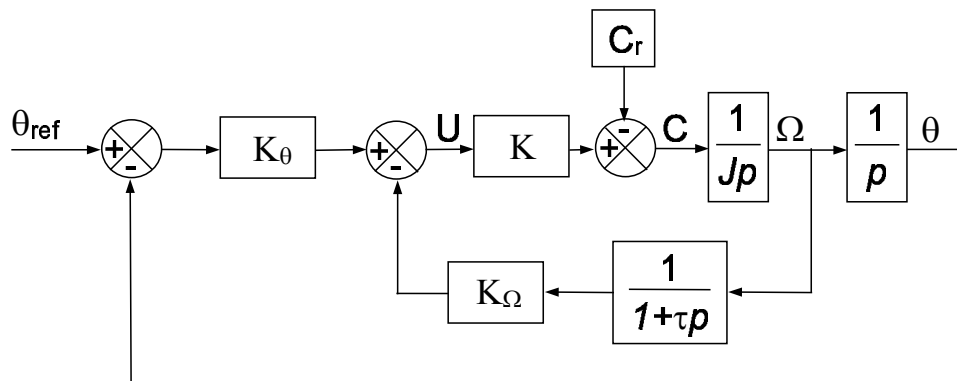


FIG. 2.20 – Schéma blocs de la commande en position avec retour de vitesse.

Dans ce modèle, nous ne prenons pas en compte les couples de frottement visqueux, étant donné qu'ils sont négligeables devant les frottements secs (voir précédemment). Quant aux frottements secs, ils sont considérés comme des perturbations; il sera néanmoins difficile de les compenser compte tenu de leur variation et de leur dépendance avec le signe de  $\Omega$ .  $\frac{1}{1+\tau p}$  représente la fonction de transfert du filtre dans le domaine continu.

Nous écrivons la fonction de transfert du système (2.23) dont nous cherchons à déterminer  $K_\theta$  et  $K_\Omega$  de manière à obtenir un système stable avec un temps de réponse satisfaisant. Plusieurs méthodes existent mais nous choisissons la méthode de Naslin [HC97]

qui donne directement des valeurs numériques pour les correcteurs suivant le type de réponse souhaitée. Selon la méthode, on définit les termes  $\omega_0$ ,  $\alpha_1$  et  $\alpha_2$  (2.24)(2.25)(2.26) en fonction des coefficients du dénominateur de la fonction de transfert. On doit avoir  $\alpha_1 = \alpha_2 = \alpha$  où  $\alpha$  est un coefficient qui dépend de la réponse souhaitée du système. On obtient donc (2.27) et (2.28).

$$\frac{\theta}{\theta_{ref}} = \frac{1 + \tau p}{\frac{J\tau}{K K_\theta} p^3 + \frac{J}{K K_\theta} p^2 + \frac{K_\Omega + \tau}{K_\theta} p + \frac{1}{K_\theta}} \quad (2.23)$$

$$\omega_0 = \frac{1}{K_\Omega + \tau} \quad (2.24)$$

$$\alpha_1 = \frac{K (K_\Omega + \tau)^2}{K_\theta J} \quad (2.25)$$

$$\alpha_2 = \frac{J}{K \tau (K_\Omega + \tau)} \quad (2.26)$$

$$K_\theta = \frac{J}{\alpha^3 K \tau^2} \quad (2.27)$$

$$K_\Omega = \frac{J}{\alpha K \tau} - \tau \approx \frac{J}{\alpha K \tau} \quad (2.28)$$

Le choix de  $\alpha$  dépend de l'instant de premier dépassement  $t_D$  souhaité ( $t_D = 2, 2/\omega_0$ ) et du pourcentage de dépassement  $D$  (exemple :  $D=6,3\%$  pour  $\alpha = 2$ ). Cependant, les temps de premier dépassement expérimentaux sont environ une fois et demie plus importants que ceux prédits et les pourcentages de dépassement sont plus faibles que prévu (environ deux fois moins importants). Nous voyons là les limites du modèle simplifié que nous avons considéré, qui néglige en particulier les frottements secs. Finalement pour avoir un bon compromis entre l'instant et le pourcentage de premier dépassement, nous avons choisi  $\alpha = 1,1$  pour lequel  $K_\theta = 0,311$  et  $K_\Omega = 25,7$ , ce qui donne  $D = 28\%$  et  $t_D = 0,035$  s. Ce dépassement n'est pas gênant car il est obtenu sans intervention de l'utilisateur. Lorsque celui-ci interagit avec le système, le dépassement est quasiment annulé.

### 2.6.3 Calcul des paramètres optimaux d'un mur virtuel

#### Calcul de la raideur et de l'amortissement virtuels

Forts de notre commande en position précédemment décrite, nous cherchons maintenant à créer un mur virtuel, simulé par un ressort ( $k$ ) et un amortisseur ( $a$ ) en parallèle (fig. 2.22), dans un environnement virtuel. Nous avons décrit dans la première partie les méthodes de stabilisation d'un mur virtuel. En pratique, la méthode la plus souvent utilisée est de fixer une raideur virtuelle puis de régler l'amortissement pour obtenir un système stable. Nous avons essayé cette méthode non-systématique qui n'a pas permis de trouver un couple de valeurs ( $k$  et  $a$ ) rendant le système stable (sans doute à cause du faible coefficient de frottement visqueux de la manette). Pour ces raisons, nous avons développé notre propre méthode de réglage des coefficients avec une approche continue du système.

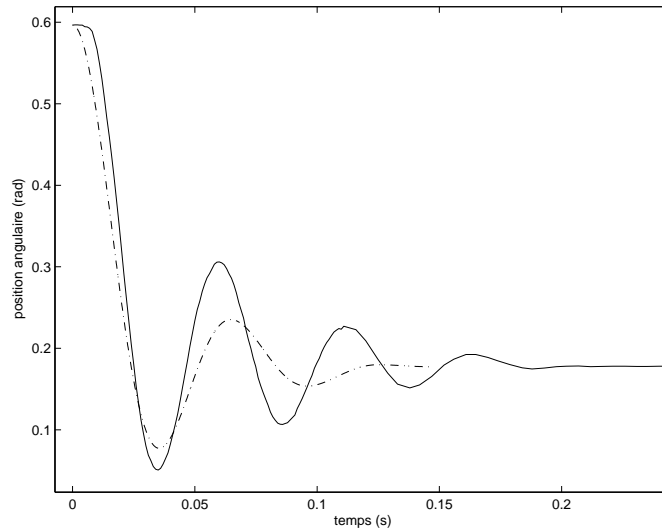


FIG. 2.21 – Commande en position de la manette avec retour de vitesse avec  $K_\theta = 25,7$ ,  $K_\Omega = 0,311$  pour un échelon indiciel quand  $\theta_{ref}$  passe de  $0,6 \text{ rad}$  à  $0,17 \text{ rad}$  ( $10^\circ$ ). Le trait continu représente l'essai expérimental et le trait discontinu, la simulation sous Simulink.

Nous proposons d'exploiter la commande en position car on remarque que le schéma bloc du mur dans l'environnement virtuel est tout à fait similaire à celui de l'asservissement de position dans l'environnement réel (fig. 2.20 et fig. 2.23).

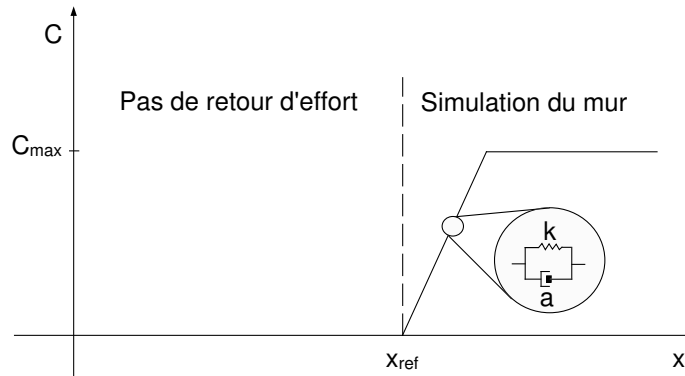


FIG. 2.22 – Représentation d'un mur virtuel.

Effectivement, sur cette figure 2.23, le système de la manette ainsi que les paramètres de raideur et d'amortissement du mur virtuel sont représentés.  $S$  représente un facteur de sensibilité qui fait correspondre la position angulaire de la manette ( $\theta$ ) au déplacement de l'objet manipulé dans les unités de longueur de l'environnement virtuel ( $x$ ).  $x_{ref}$  est la position du mur dans l'environnement virtuel. Les positions et vitesses angulaires sont transmises à l'environnement virtuel à la fréquence de 1000Hz qui calcule et renvoie les consignes de forces aux moteurs (cf. paragraphe 2.4.2). Lorsque l'objet manipulé entre dans le mur, la distance de pénétration est mesurée ( $x_{ref} - x$ ) et multipliée par la raideur du mur  $k$ . L'avancée de l'objet dans le mur est amortie par un terme dépendant de la vitesse de l'objet, qui correspond à la vitesse de la manette à la sensibilité  $S$  près et

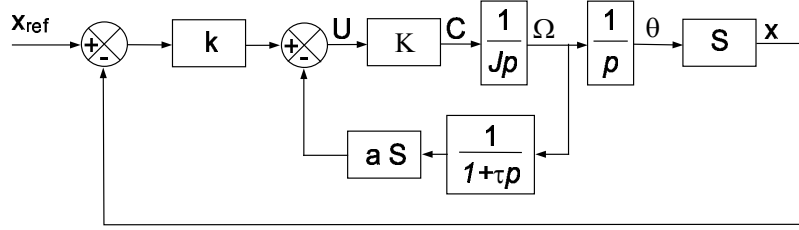


FIG. 2.23 – Schéma blocs de la simulation d'un mur virtuel.

proportionnellement au coefficient  $a$ . La différence des deux termes correspond à  $U^{18}$ . Par ailleurs, le modèle n'est activé que lorsque l'utilisateur est en contact avec le mur. Le reste du temps, les courants d'induit des moteurs sont nuls.

La fonction de transfert du système (2.29) donne après application de la méthode de Naslin les valeurs de  $k$  et  $a$  (2.30)(2.31)

$$\frac{x}{x_{ref}} = \frac{1 + \tau p}{\frac{J\tau}{SKk} p^3 + \frac{J}{SKk} p^2 + \frac{k\tau+a}{k} p + 1} \quad (2.29)$$

$$k = \frac{J}{\alpha^3 S K \tau^2} \quad (2.30)$$

$$a = \frac{J}{\alpha S K \tau} \quad (2.31)$$

Notons que contrairement aux essais précédents, l'utilisateur interagit désormais avec le système. Le doigt crée un nouveau couple perturbateur et joue l'effet d'un amortisseur en atténuant les oscillations de la manette<sup>19</sup>, ce qui nous a permis de diminuer  $\alpha$  à une valeur de 0,7 (les  $\alpha$  plus faibles donnent des instabilités au contact du mur). Cela conduit, pour une sensibilité  $S$  de 2.9, à un coefficient de raideur du mur virtuel  $k$  de 34,39 et un amortissement  $a$  de 0,168.

### Calcul de la raideur physique

Auparavant, nous avons calculé la raideur et l'amortissement dans l'environnement virtuel, maintenant nous cherchons à déterminer quelle est la raideur physique simulée sur la manette ( $R_S$ ) correspondant au coefficient  $k$  que nous venons de calculer. Pour calculer  $R_S$ , nous cherchons de quelle distance se déplace le doigt quand la force qui s'exerce sur lui passe de 0 N à la force maximale pouvant être rendue à l'extrémité de la manette. Nous connaissons les relations (2.32) et (2.33) et nous savons que le courant maximal admissible par le moteur est de 1A, nous en déduisons  $U_{max}$  qui est alors la

<sup>18</sup>Plutôt que de contrôler la tension aux bornes de la carte de commande, nous aurions pu calculer la force à renvoyer à l'extrémité de la manette à partir de la différence entre les deux termes (ce qui est plus intuitif), puis en déduire la tension  $U$  qui est proportionnelle à la force. Pour garder l'analogie avec la commande en position, nous avons privilégié la simplicité, ce qui ne change rien au problème de commande.

<sup>19</sup>Comme l'ont aussi remarqué Salcudean et al. lors de la simulation de mur sur un interface magnétique [SV97]

tension maximale à l'entrée de la carte de commande (2.35). Connaissant la relation entre  $U$  et  $\theta$  (2.36), nous en déduisons la variation d'angle correspondante  $\theta_{max}$  (2.37) et ainsi le déplacement du doigt correspondant  $d_{max}$  (2.38) en fonction du rayon de la manette  $r$  pour lequel la force est maximale (2.39).

$$C = K U \quad (2.32)$$

$$C = N k_i i \quad (2.33)$$

$$i_{max} = 1 A \quad (2.34)$$

$$U_{max} = \frac{N k_i}{K} \quad (2.35)$$

$$U = k S \theta \quad (2.36)$$

$$\theta_{max} = \frac{N k_i}{K k S} \quad (2.37)$$

$$d_{max} = \frac{N k_i r}{K k S} \quad (2.38)$$

$$F_{max} = \frac{N k_i}{r} \quad (2.39)$$

$$R_S = \frac{K k S}{r^2} \quad (2.40)$$

Pour le coefficient  $k$  précédemment calculé, nous obtenons une raideur simulée ( $R_S$ ) sur la manette de  $1050 N.m^{-1}$  (2.40). Nous sommes loin des  $15000 N.m^{-1}$  (voir sec.2.4.1) qui permettent à l'utilisateur de ne plus faire la différence entre un mur réel et un mur simulé. En comparaison, le Phantom peut simuler des raideurs de l'ordre de  $3500 N.m^{-1}$  [MS94] et le HapticMaster de l'ordre de  $10000 N.m^{-1}$  à  $50000 N.m^{-1}$  [dLLFR02]. Il faut tout de même remarquer que ces périphériques sont destinés à être manipulés avec la main. On peut supposer que la raideur minimale à partir de laquelle l'utilisateur ne fait plus la différence entre un mur réel et virtuel est moindre au niveau des doigts qu'au niveau de la main. Bien que nous n'ayons pas mené d'études expérimentales sur la qualité de ce mur, les personnes qui l'ont testé demandent en général si la manette n'est pas en fait en butée mécanique. Nous sommes donc satisfaits du résultat.

Pour augmenter  $R_S$ , il faut augmenter  $k$ . Pour augmenter ce dernier, on peut soit augmenter l'inertie  $J$  de la manette (il faut alors en concevoir une nouvelle avec un matériau plus dense que l'aluminium), soit diminuer la constante de temps  $\tau$  du filtre. Cependant, en l'état actuel du dispositif de calcul et pour garder une approche pseudo continue du système, il n'est pas possible de diminuer cette constante de temps. Nous avons donc réaliser le "meilleur" mur possible compte tenu de l'ensemble des paramètres du système.

A noter qu'il est possible d'utiliser des artifices supplémentaires pour augmenter la qualité du mur tel que la pulsion d'amortissement<sup>20</sup> qui consiste à appliquer un amortissement important juste au moment où l'utilisateur rentre dans le mur [SV97]. Une autre technique consiste à ajouter un terme intégral [Mas96]. Celui-ci croît proportionnellement au temps d'appui sur le mur.

---

<sup>20</sup>breaking pulse dans l'article



### 2.6.4 Commande en impédance du système

Que ce soit pour simuler un mur ou d'autres phénomènes physiques, nous allons voir comment la boucle de contrôle de la machine QNX (voir sec. 2.4.2) gère les forces selon les modes d'utilisations choisis.

#### Retour de force en mode isotonique

Dans ce mode d'utilisation, la force reçue par la machine QNX est directement envoyée au système. La tension appliquée à la carte de commande est donc proportionnelle à cette force (en tenant compte du rayon de la manette  $r$ ) (2.41). Nous étudierons plus en détail le mode isotonique dans le chapitre suivant.

$$U = \frac{F r}{K} \quad (2.41)$$

#### Retour de force en mode élastique

Dans ce mode élastique (voir chapitre suivant), des ressorts angulaires de raideur ( $R$ ) et d'amortissement ( $A$ ) modifiables en cours d'utilisation sont simulés sur les manettes. Le réglage de l'amortissement permet d'éviter que la manette n'entre en oscillations. Les ressorts sont directement simulés, en permanence, sur la machine QNX et ne sont pas calculés par la machine Windows. Les forces calculées dans l'environnement virtuel sont ensuite ajoutées à l'effet du ressort. La tension aux bornes de la carte de commande vaut alors (2.42).

$$U = \frac{R \theta}{K} + \frac{A \dot{\theta}}{K} + \frac{F r}{K} \quad (2.42)$$

Nous cherchons la valeur d'amortissement minimale évitant que la manette n'entre en oscillations dès que l'utilisateur la lâche à partir d'une position non nulle. Cette valeur d'amortissement dépendra de la valeur du coefficient de raideur.

Pour répondre à cette question, nous pouvons écrire l'équation différentielle du mouvement de la manette (2.43) que nous pouvons mettre sous la forme (2.44) en posant  $\omega_0$  la pulsation propre de l'oscillateur (2.45),  $\lambda$  son coefficient d'amortissement (2.46) et  $Q$  son facteur de qualité (2.47)

$$J \ddot{\theta} + A \dot{\theta} + R \theta = 0 \quad (2.43)$$

$$\ddot{\theta} + 2 \lambda \dot{\theta} + \omega_0^2 \theta = 0 \quad (2.44)$$

$$\omega_0^2 = \frac{R}{J} \quad (2.45)$$

$$\lambda = \frac{A}{2J} \quad (2.46)$$

$$Q = \frac{\omega_0}{2\lambda} \quad (2.47)$$

Si  $Q > \frac{1}{2}$ , l'oscillateur est peu amorti et oscillera beaucoup. Si  $Q = \frac{1}{2}$ , l'oscillateur est en régime critique et atteindra sa position stable le plus rapidement possible. On a alors  $A = 2\sqrt{RJ}$ . Finalement si  $Q < \frac{1}{2}$ , nous sommes en présence de frottements importants. Nous choisissons donc  $Q = \frac{1}{2}$  pour les réglages.

Le réglage des raideurs se fait à l'aide d'une interface graphique (fig. 2.24) qui permet de régler individuellement la raideur, l'amortissement et le frottement sec de chaque manette. L'amortissement est automatiquement réglé en fonction de la raideur définie. Il est ensuite possible de l'ajuster pour améliorer le résultat. Nous verrons dans le prochain chapitre l'intérêt de régler le frottement sec des manettes.

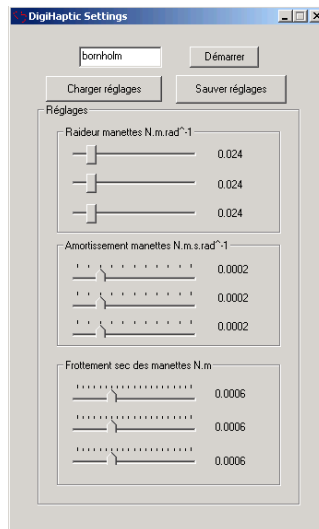


FIG. 2.24 – Interface de paramétrage des manettes.

## Simulation de murs en mode élastique

De la même manière que nous avons déterminé les paramètres optimaux d'un mur pour un contrôle en position dans l'environnement virtuel sans simulation de raideur sur les manettes, nous allons maintenant chercher à déterminer ces paramètres lors de la simulation de raideurs sur les manettes avec contrôle en vitesse dans l'environnement virtuel.

En mode élastique, une raideur est simulée sur la manette de manière à la ramener toujours en position neutre. Pour cette position neutre, la vitesse de déplacement de l'objet manipulé est nulle. Lorsque l'utilisateur appuie sur la manette dans un sens ou dans l'autre, la vitesse de l'objet croît proportionnellement au déplacement de la manette dans la direction donnée. Si l'objet manipulé entre en collision avec un mur, il est visuellement projeté sur celui-ci et la distance de pénétration est mesurée pour renvoyer un effort à l'utilisateur (voir §1.4.1 pour plus de détails).

Un point important à noter lors de la collision est qu'il ne faut plus contrôler l'objet en vitesse mais en position. En effet, quand l'objet entre en collision avec le mur, la position de la manette est non nulle. Une force de retour qui tend à ramener la manette vers sa position neutre est alors calculée en fonction de la distance de pénétration. Cependant,

lorsque la manette revient vers sa position neutre, la vitesse de déplacement est toujours de même signe puisque le signe de la position angulaire de la manette ne change pas. La distance de pénétration continue donc de croître au lieu de décroître. Ainsi lorsque l'utilisateur veut sortir du mur, le proxy reste collé au mur. Il ne se décolle du mur que lorsque la position angulaire de la manette change de signe et que la distance de pénétration s'annule.

Pour modéliser le mur, nous sommes à nouveau dans le cas d'une commande en position. Nous reprenons donc le schéma 2.23 auquel nous ajoutons la raideur  $R$  et l'amortissement  $A$  simulés sur la manette. Lorsqu'il y a collision, la manette contrôle la position  $x$  de l'objet dans le mur. La position du mur correspond à  $x_{ref}$  et la distance de pénétration ( $x_{ref} - x$ ) sert à la simulation du mur défini par sa raideur  $k$  et son amortissement  $a$ . La sensibilité  $S$  est superflue et peut être prise égale à 1. Elle est maintenue pour des raisons pratiques de programmation.

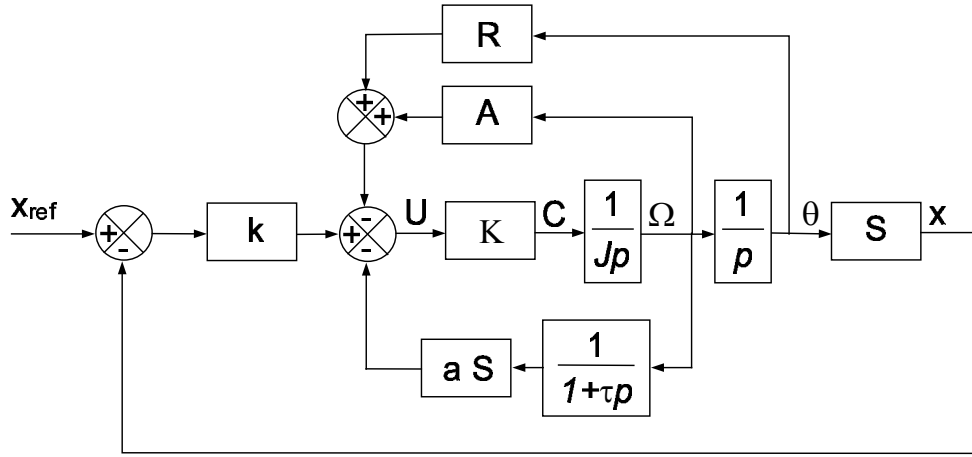


FIG. 2.25 – Schéma blocs de la simulation d'un mur virtuel avec simulation de raideurs sur les manettes.

Nous cherchons, pour la fonction de transfert correspondante en boucle fermée (2.48), les paramètres  $a$  et  $k$  qui rendent le système stable et rapide pour  $k$  le plus grand possible.

$$\frac{x}{x_{ref}} = \frac{1 + \tau p}{\frac{J\tau}{KkS}p^3 + \left(\frac{J}{KkS} + \frac{A\tau}{Sk}\right)p^2 + \left(\tau + \frac{a}{k} + \frac{A}{kS} + \frac{R\tau}{kS}\right)p + \frac{R}{Sk} + 1} \quad (2.48)$$

Nous appliquons à nouveau la méthode de Naslin, qui donne les coefficients suivants :

$$\omega_0 = \frac{1 + \frac{R}{kS}}{\tau + \frac{a}{k} + \frac{A}{kS} + \frac{R\tau}{kS}} \quad (2.49)$$

$$\alpha_1 = \frac{\left(\tau + \frac{a}{k} + \frac{A}{kS} + \frac{R\tau}{kS}\right)^2}{\frac{J}{KkS} + \frac{A\tau}{Sk}} \quad (2.50)$$

$$\alpha_2 = \frac{\left(\frac{J}{KkS} + \frac{A\tau}{Sk}\right)^2}{\left(\tau + \frac{a}{k} + \frac{A}{kS} + \frac{R\tau}{kS}\right) \frac{J\tau}{KkS}} \quad (2.51)$$

Ce qui donne pour  $k$  et  $a$  :

$$k = \frac{K^3 A^3 \tau^3 + 3 \tau^2 J K^2 A^2 + 3 J^2 K A \tau + J^3}{\alpha^3 K J^2 \tau^2 S} \quad (2.52)$$

$$a = -\frac{-J^3 \alpha^2 - 2 J^2 A \tau K \alpha^2 - A^2 \tau^2 K^2 \alpha^2 J + K^3 A^3 \tau^3 + 3 \tau^2 J K^2 A^2}{\alpha^3 J^2 K \tau S} \\ - \frac{3 J^2 K A \tau + J^3 + \alpha^3 K J^2 \tau A + \alpha^3 K J^2 \tau^2 R}{\alpha^3 J^2 K \tau S} \quad (2.53)$$

Pour obtenir un bon compromis entre raideur du mur et stabilité, nous avons choisi  $\alpha = 1, 2$ .

### Simulation de vibrations

Le DigiHaptic peut principalement simuler des forces. Nous montrons ici comment produire des vibrations. Ces vibrations peuvent être simulées seules ou être surimposées aux forces générées.

Les vibrations ont une fréquence  $f$  et une amplitude  $A$ . L'équation (2.54) donne un exemple de vibration où  $t$  est le temps et  $B$  une constante à modifier en fonction du résultat souhaité. Il est possible d'imaginer des composées de fonctions trigonométriques pour obtenir des résultats différents.

$$U = A \sin(B f t) \quad (2.54)$$

Nous avons par ailleurs simulé des crans sur les manettes, avec possibilité de régler l'angle entre deux crans et la hauteur des crans.

## 2.7 Conclusion

Nous avons vu les motivations qui nous ont poussées à proposer le concept original de la séparation des degrés de liberté sur un périphérique haptique à trois degrés de liberté. En concevant le périphérique, nous avons cherché à rendre l'utilisation des degrés de liberté la plus intuitive possible en respectant les isomorphismes d'orientation et de direction.

À partir des contraintes ergonomiques, nous avons ensuite réalisé le périphérique après les étapes de conception mécanique et électronique. Nous verrons, après les étapes d'évaluation, les problèmes ergonomiques relevés par les utilisateurs. Ceux-ci nous ont amenés à développer une nouvelle version que nous présenterons alors. Lors des étapes de conception mécanique et électronique, nous avons à chaque fois cherché la simplicité et l'efficacité. Les solutions adoptées font du DigiHaptic un système robuste et fiable. La séparation des degré de liberté a l'avantage de simplifier considérablement la complexité mécanique de l'interface haptique et ainsi de diminuer nettement les coûts de fabrication. Ces travaux ont fait l'objet de deux publications [CCSP03][CPCS03b].

Nous avons ensuite identifié les paramètres d'inertie et de frottements secs et visqueux d'un bloc élémentaire du DigiHaptic. Cette identification nous a montré que les frottements visqueux sont négligeables par rapport aux frottements secs. La faiblesse des frottements visqueux, si elle est un avantage pour la simulation des environnements libres,

rend néanmoins instable la simulation de murs avec raideurs importantes. Compte tenu de ce phénomène, nous avons développé une méthode originale de recherche des paramètres optimaux d'un mur virtuel, qui fait abstraction du frottement visqueux de l'interface.

Enfin, nous avons montré comment réaliser la commande en impédance du système en mode isotonique et élastique. Pour ce dernier, nous avons vu comment appliquer le bon coefficient d'amortissement pour rendre la manette stable et nous avons ensuite étudié comment réaliser des murs en mode élastique. Nous avons alors mis en évidence que l'objet manipulé doit être contrôlé en position lors d'une collision et nous avons calculé les paramètres optimaux du mur pour rendre le système stable.

Comme nous maîtrisons à présent la commande du périphérique, nous allons désormais nous intéresser aux différents modes d'utilisation du périphérique. Ceux-ci vont montrer le champ d'applications du DigiHaptic.



# Chapitre 3

## Interaction avec le DigiHaptic

### Sommaire

---

<b>3.1</b>	<b>Introduction</b>	<b>63</b>
<b>3.2</b>	<b>Tâches</b>	<b>64</b>
3.2.1	Manipulation	64
3.2.2	Navigation	64
<b>3.3</b>	<b>Interaction avec le DigiHaptic dans les tâches de manipulation</b>	<b>66</b>
3.3.1	Mode isotonique	67
3.3.2	Mode élastique	69
3.3.3	Passage entre mode isotonique et mode élastique	72
3.3.4	Résumé des modes d'utilisation	73
3.3.5	Combinaison mode isotonique et mode élastique	73
<b>3.4</b>	<b>Interaction avec le DigiHaptic dans les tâches de navigation</b>	<b>78</b>
3.4.1	Métaphore de contrôle de la caméra	78
3.4.2	Calcul du retour de force sans glissement sur la surface	80
3.4.3	Glissement sur les surfaces	80
<b>3.5</b>	<b>Le retour d'effort en mode élastique</b>	<b>80</b>
3.5.1	Introduction	80
3.5.2	Analyse	81
3.5.3	Mode maître	82
3.5.4	Mode esclave	83
3.5.5	Comparaison des modes maître et esclave	83
<b>3.6</b>	<b>Conclusion</b>	<b>84</b>

---

### 3.1 Introduction

Ce chapitre commence par une description des tâches de manipulation d'objets 3D et de navigation en environnement 3D. Nous verrons notamment quels types de périphériques peuvent être utilisés pour chacune des tâches. Cela nous permettra de voir les applications possibles du DigiHaptic.

Nous verrons ensuite les modes d'utilisation du DigiHaptic pour la manipulation d'objets 3D avec les applications de démonstration développées dans chacun des cas. Plus particulièrement, nous étudierons les modes élastiques et isotoniques pour la translation et la rotation d'objets et nous verrons les possibilités de passer d'un mode à l'autre et d'utiliser un mode hybride isotonique élastique.

Nous continuerons sur l'étude du DigiHaptic pour les tâches de navigation en environnement 3D avec retour d'effort. Nous verrons notamment comment rendre les forces lors d'une collision de la caméra avec l'environnement.

Enfin, nous discuterons du retour d'effort en mode élastique pour rendre des forces de collisions entre objets déformables.

## 3.2 Tâches

### 3.2.1 Manipulation

Une tâche de manipulation nécessite l'utilisation de six degrés de liberté : l'utilisateur doit pouvoir contrôler la position et l'orientation des objets de la scène. Le périphérique doit ainsi posséder de préférence au moins six degrés de liberté.

Les périphériques isotoniques sont généralement utilisés pour ce type de tâche car les gestes de l'utilisateur sont directement reproduits dans l'environnement virtuel, ce qui rend leur utilisation intuitive. Pour la manipulation en environnement 3D, ces périphériques ont cependant l'inconvénient d'avoir un espace de travail physique limité (voir §2.2.1), ce qui limite l'espace de travail dans l'environnement virtuel. Pour dépasser cette limite, il est possible d'utiliser les périphériques élastiques qui permettent de travailler dans un environnement virtuel de taille illimitée grâce au contrôle en vitesse. Ces périphériques sont cependant moins intuitifs à utiliser.

La manipulation peut également être faite avec retour d'effort pour permettre à l'utilisateur de ressentir le poids des objets ou leur forme. Dans ce cas, seuls les périphériques isotoniques à retour d'effort sont utilisés.

### 3.2.2 Navigation

Une tâche de navigation en environnement 3D, qui suppose un contrôle du point de vue de la caméra, peut se décomposer en deux étapes : la recherche de chemin (wayfinding) et le contrôle de point de vue. La première étape est un processus cognitif qui amène l'utilisateur à déplacer le point de vue dans la scène (processus moteur). A ce propos, Darken et Sibert ont montré que l'utilisation de cartes de navigation pouvait améliorer les performances de l'utilisateur [DS93] [DS96].

Au début des années 90, les efforts se sont concentrés sur la recherche des meilleures métaphores de navigation. Mackinlay et al. ont proposé une métaphore pour les périphériques 2D qui consiste à désigner une cible dans la scène pour déplacer le point de vue [MCR90]. Une adaptation logarithmique de la vitesse est réalisée suivant la distance à la cible. Ware et al. ont, quant à eux, proposé des métaphores pour les périphériques 6ddl



isotoniques<sup>21</sup> [WO90]. Ils définissent trois métaphores : le *globe oculaire dans la main*, la *scène dans la main* et le *contrôle de véhicule*<sup>22</sup>. La métaphore du *globe oculaire dans la main* consiste à manipuler le périphérique comme une caméra réelle et à le placer dans la scène, en utilisant un contrôle en position. Cela occasionne une fatigue de l'utilisateur tout en limitant le volume de travail accessible dans l'environnement virtuel. La métaphore de la *scène dans la main* consiste à déplacer la scène avec le périphérique : si par exemple le périphérique est déplacé vers le haut, la scène est déplacée vers le haut ; si le périphérique est orienté vers la droite, la scène également. Le contrôle en position utilisé fait apparaître les mêmes problèmes que la précédente métaphore. En ce qui concerne la métaphore du *contrôle de véhicule*, l'utilisateur contrôle la scène comme s'il se déplaçait dans un véhicule. Les déplacements sont inversés par rapport à la métaphore précédente. Ici, un contrôle en vitesse est utilisé avec un périphérique isotonique, ce qui pose des problèmes proprioceptifs (voir §1.2.3).

Pausch et al. ont proposé une métaphore de navigation qui permet d'interagir avec une version miniature de la scène [PBBW95]. D'autres auteurs ont proposé d'utiliser une pédale pour naviguer [BMH98].

La souris reste le périphérique le plus utilisé pour la navigation, en particulier pour la navigation en environnements 3D pour les applications de bureau. Les browsers existants comme Cosmoplayer [Cos], dans le cas de la visualisation de scènes VRML, proposent différents modes qui permettent d'utiliser les deux degrés de liberté de la souris pour déplacer ou orienter le point de vue en vitesse. Le contrôle en vitesse avec un périphérique isotonique rend la navigation non intuitive et difficile.

Compte tenu du faible nombre de degrés de liberté de la souris, la navigation est souvent réalisée en combinaison avec le clavier, comme c'est le cas dans certains jeux 3D pour le contrôle des déplacements du personnage<sup>23</sup>. Le clavier est utilisé pour déplacer le personnage à vitesse constante alors que la souris sert au contrôle en position de l'orientation de la vue du personnage.

Pour les métaphores de navigation qui utilisent un contrôle en vitesse, les périphériques isométriques et élastiques sont à privilégier (voir §1.2.3). Zhai et al. ont ainsi développé une métaphore de navigation, similaire à la métaphore de *contrôle de véhicule*, utilisant deux joysticks [ZKSS99]. Paton et Ware ont développé le Bbat, un périphérique élastique à six degrés de liberté pour contrôler le point de vue de la caméra en vitesse [PW94] en utilisant deux mains. Boeck et al. ont proposé une extension de la métaphore du *globe oculaire dans la main* en utilisant un PHANToM avec simulation d'une raideur dans toutes les directions et contrôle en vitesse [BC02].

Bien que la navigation nécessite six degrés de liberté [WO90], il est souvent difficile de contrôler tous ces degrés de liberté. Hachet a développé une interface six degrés de liberté, baptisée CAT, pour la navigation en environnements 3D collaboratifs qui permet de réaliser les translations en mode isométrique avec contrôle en vitesse et les rotations en mode isotonique avec contrôle en position [Hac03]. Il est ainsi facile de contrôler finement les rotations et les translations.

<sup>21</sup> Le périphérique qu'ils ont utilisé est une souris volante.

<sup>22</sup> Respectivement *eyeball in hand*, *scene in hand* et *flying vehicle control* dans l'article [WO90].

<sup>23</sup> Ici, le personnage reste au sol lors de la navigation.

Nous venons de voir qu'il existe de nombreuses métaphores de navigation en environnements 3D. Afin de pouvoir les comparer et choisir celle qui convient le mieux à une tâche, il est nécessaire de disposer d'outils permettant de les évaluer. Bowman et al. ont ainsi proposé une méthodologie pour l'évaluation des métaphores de navigation [BKH98]. Ils proposent d'évaluer une métaphore de navigation d'après les critères suivants :

- la rapidité de la navigation pour atteindre une cible,
- la précision de la navigation à proximité d'une cible,
- la conscience de localisation de l'utilisateur (la connaissance qu'a l'utilisateur de sa position et son orientation à l'intérieur de l'environnement pendant et après la navigation),
- la facilité d'apprentissage (la capacité pour un utilisateur novice d'utiliser la métaphore),
- la facilité d'utilisation (la complexité et la charge cognitive d'une métaphore pour l'utilisateur),
- la récolte d'informations (la capacité de l'utilisateur d'obtenir activement des informations de l'environnement lors de la navigation),
- la présence (l'impression qu'a l'utilisateur d'être immergé dans l'environnement pendant qu'il navigue),
- le confort (éviter que l'utilisateur ne soit malade, ait des vertiges ou des nausées).

Les deux premiers critères sont quantitatifs. Les autres critères sont qualitatifs.

Les périphériques isotoniques et élastiques sont utilisés en navigation. Les périphériques isotoniques sont meilleurs pour le contrôle en position et donc bien adaptés à la métaphore du globe oculaire dans la main ou celle de la scène dans la main. Ces métaphores ne permettent cependant pas de se déplacer dans un environnement de taille infinie. Les périphériques élastiques, meilleurs pour le contrôle en vitesse, sont donc mieux adaptés pour la métaphore du contrôle de véhicule par exemple, où l'utilisateur n'est pas limité dans ses déplacements.

Dans chacun des cas, il est envisageable de rendre les forces de collision lorsque la caméra manipulée par l'utilisateur entre en collision avec l'environnement. Si le rendu de ces forces de collision est facile à mettre en œuvre avec des périphériques haptiques isotoniques, comment rendre ces forces dans le cas d'un périphérique élastique ? La littérature ne contient aucune référence sur le rendu de forces de collisions dans les tâches de navigation. Pour le DigiHaptic, nous verrons comment rendre les forces de collision en navigation en mode élastique.

### 3.3 Interaction avec le DigiHaptic dans les tâches de manipulation

Le DigiHaptic peut être utilisé soit en mode isotonique ou en mode élastique. Pour ces deux modes, nous nous limitons ici aux tâches de manipulation. Nous allons voir dans chacun des cas les possibilités d'utiliser le DigiHaptic pour les rotations et les translations.

Nous étudierons également comment rendre les forces en mode isotonique. Le rendu de force en mode élastique sera discuté plus tard. Nous donnerons à chaque fois des exemples de leur mise en œuvre.

### 3.3.1 Mode isotonique

En mode isotonique, les manettes servent à faire un contrôle en position des objets manipulés. Compte tenu du débattement restreint de chaque manette de notre périphérique isotonique absolu (4 cm), le volume de travail est limité à une portion de l'écran. Pour un control display modéré de  $3^{24}$ , on obtient un volume projeté à l'écran de 12 cm de côté.

Nous associons un repère cartésien à l'environnement virtuel, dans lequel chaque manette contrôle le déplacement de l'objet manipulé suivant un axe tel que décrit en section 2.3.1. Les isomorphismes d'orientation et de direction que nous avons choisi permettent d'une part une utilisation plus intuitive des degrés de liberté séparés et d'autre part la réalisation du retour d'effort.

Alors qu'avec un périphérique à degrés de liberté assemblés, le vecteur force calculé dans l'environnement virtuel est directement envoyé sur l'unique effecteur afin d'être fidèlement reproduit, le résultat est moins immédiat avec un périphérique à degrés de liberté séparés. La solution que nous avons adoptée consiste à projeter le vecteur force dans le repère cartésien associé à l'environnement virtuel puis à envoyer chaque composante de cette projection à la manette correspondante. De cette façon, la direction et le sens de chaque composante du vecteur force sont identiques à ceux des déplacements des manettes et des doigts correspondants. Cela est rendu possible par l'isomorphisme d'orientation et de direction de chaque manette.

Si nous nous ramenons en 2D, la figure 3.1 nous montre l'exemple d'un utilisateur qui manipule un pointeur<sup>25</sup> et entre en collision avec un mur. Une force de réaction normale est alors calculée puis projetée sur les axes "x" et "y" du repère cartésien. La composante  $\vec{F}_x$  est envoyée sur le pouce et la composante  $\vec{F}_y$  sur l'annulaire. La perception qui en résulte n'est pas déroutante pour l'utilisateur puisqu'il y a correspondance entre le retour visuel et kinesthésique : le pouce et l'annulaire vont être bloqués dans le sens du mur et ne pourront se déplacer que dans l'autre sens. Bien que nous n'ayons pas mené d'études approfondies sur la question, nous pouvons dire, lors d'une manipulation en aveugle, que l'utilisateur sent si le mur est rectiligne ou non, "monte" ou "descend". L'utilisateur est donc capable de détecter les changements de sens de la force sur les doigts mais aussi les variations de force. Il ne semble cependant pas capable de distinguer entre différentes inclinaisons de mur, à moins qu'elles ne soient franches (mur orienté à  $10^\circ$  et  $80^\circ$  par exemple). L'exercice est bien entendu plus facile quand uniquement deux des trois doigts sont sollicités.

En manipulation, six degrés de liberté sont nécessaires pour positionner et orienter un objet dans l'espace. Comme le DigiHaptic n'en a que trois, il n'est possible de manipuler

---

<sup>24</sup>Cette valeur de control display a été obtenue de manière empirique. Ce n'est pas le résultat d'une étude systématique de l'influence du control display sur les performances de l'utilisateur.

<sup>25</sup>Le pouce est utilisé pour déplacer le pointeur suivant l'axe "x" et l'annulaire suivant l'axe "y"

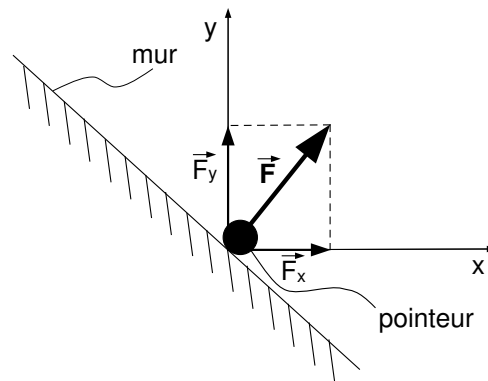


FIG. 3.1 – Projection du vecteur force dans le repère cartésien.

que trois des six degrés de liberté à la fois. Il se pose alors la question du passage des trois premiers ddl aux trois suivants : si l'utilisateur manipule les trois ddl en translation, peut-il ensuite utiliser ces trois ddl en rotation ? La réponse est négative avec un périphérique isotonique absolu. En effet, lors du passage d'un mode à l'autre, les manettes ne sont pas forcément à leur position de référence (0 degré) et il va donc s'appliquer une rotation ou une translation instantanée non voulue par l'utilisateur. Pour remédier à ce problème, il faudrait prendre comme nouvelle position de référence la position actuelle des ddl. Dans ce cas, l'utilisateur peut se trouver bloqué en limite de l'espace de travail du périphérique, ce qui est vrai quelque soit le périphérique isotonique absolu. Nous pouvons donc dire qu'avec un périphérique isotonique absolu, l'utilisation des degrés de liberté tour à tour en translation et en rotation n'est pas possible pour la manipulation d'objets avec contrôle en position.

Nous avons mis en oeuvre le mode isotonique avec retour de force dans différentes démonstrations. La première (voir fig. 3.2), écrite en C++/OpenGL, illustre la simulation de murs dont les paramètres optimaux ont été déterminés dans la section 2.6.3. L'utilisateur contrôle la position du cube dans la boîte et ressent les forces de collision lorsqu'il entre en collision avec un mur. Les murs sont ici codés "en dur" dans l'application et une détection de collision cube/plan est réalisée.

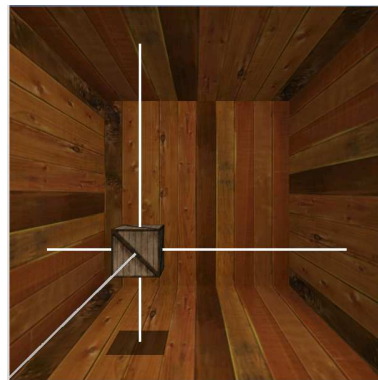


FIG. 3.2 – Contrôle en position du cube avec retour de force sur les murs.

La seconde démonstration (fig. 3.3) a été réalisée en utilisant la librairie e-Touch [eT]. Celle-ci permet de décrire facilement un environnement 3D composé d'objets élémentaires tels que des sphères, parallélépipèdes ou cylindres, et de gérer le retour d'effort entre un pointeur et ces objets. Le Phantom est pris en charge par défaut et nous avons développé un driver adapté pour connecter le DigiHaptic à l'environnement. La gestion de collision et le retour de force permettent ici d'appréhender des objets avec des surfaces non planes.

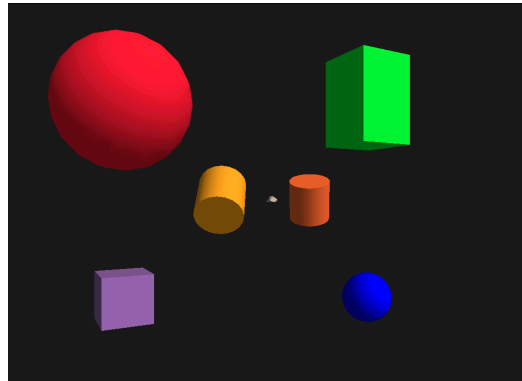


FIG. 3.3 – Contrôle en position du pointeur avec retour de force sous e-Touch.

Les deux premières démonstrations gèrent les collisions entre objets rigides. La dernière démonstration (fig. 3.4) permet la gestion de collisions entre objets déformables grâce à la bibliothèque Spore [DMC02] pour laquelle nous avons également développé un driver adapté. Dans cet environnement, l'utilisateur manipule un pointeur représenté par une sphère rouge et peut interagir avec le drap tenu par les quatre briques rouges. Les forces perçues sont bien plus faibles que celle renvoyées lors de la simulation de murs dans les démonstrations précédentes. Les forces ressenties lors de l'interaction avec le drap ne perturbent pas l'utilisateur car il y a correspondance entre le mouvement des doigts, celui du pointeur et les forces renvoyées.

### 3.3.2 Mode élastique

En mode élastique, des ressorts de raideurs ajustables sont simulés sur chaque manette (voir §2.6.4) et un contrôle en vitesse est réalisé dans l'environnement virtuel. Celui-ci permet d'utiliser le périphérique en environnements de taille illimitée pour la manipulation d'objets ou la navigation (comme nous le verrons plus loin). Le contrôle en vitesse fait correspondre une vitesse de déplacement dans l'environnement virtuel à la position de la manette. En général, cette relation est proportionnelle. Le réglage du coefficient de proportionnalité se fait en ajustant la vitesse de déplacement pour de faibles déplacements de la manette. La vitesse obtenue pour de plus grands déplacements de la manette peut alors paraître trop lente par rapport à celle attendue par l'utilisateur. Dans ce cas, il est possible d'utiliser une fonction polynômiale de degré trois pour laquelle les vitesses de déplacements seront faibles pour de petits déplacements de la manette et nettement plus important pour des déplacements de manettes eux mêmes plus importants (fig. 3.5). Rutledge et al. proposent, pour les périphériques isométriques, une fonction définie par morceaux dont la courbe à une allure de courbe similaire [RS90].



FIG. 3.4 – Contrôle en position du pointeur avec retour de force sous Spore.

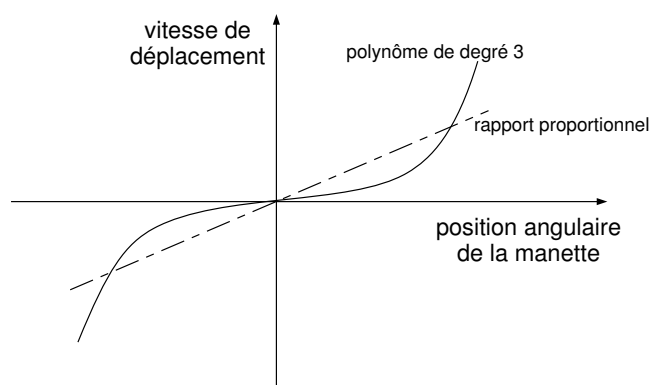


FIG. 3.5 – Rapport entre position angulaire et vitesse de déplacement pour deux fonctions.

Le passage du mode translation au mode rotation se fait sans discontinuité après avoir relâché les manettes puisque celles-ci se replacent toujours en position nulle sous l'effet du ressort, ce qui est vrai pour tout périphérique élastique. Nous pouvons donc dire qu'avec un périphérique élastique ou isométrique pour la manipulation d'objets en vitesse, l'utilisation des degrés de liberté tour à tour en translation et rotation est possible.

Nous avons mis en oeuvre ce mode dans une démonstration de manipulation d'objets. Dans l'application, l'utilisateur peut charger un objet au format OBJ et le manipuler en translation ou rotation. La rotation de l'objet est obtenue en utilisant les quaternions pour définir l'axe et l'angle de rotation qui correspondent à la combinaison des rotations de chacune des manettes (voir annexe A). Le format OBJ a l'avantage d'être simple à lire, les informations sont en effet décrites sous forme de points et de polygones à afficher. Le passage du mode rotation au mode translation se fait en cliquant sur un bouton de la souris. Il est prévu de rajouter un ou deux boutons sur le DigiHaptic au niveau du majeur.

Pour les translations ou les rotations d'objets en mode élastique, les relations entre les déplacements des doigts et les mouvements de l'objet manipulé sont identiques à ceux du mode isotonique (voir §2.3.1).

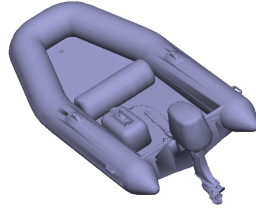


FIG. 3.6 – Application de démonstration de manipulation d'objets en mode élastique (exemple d'objet manipulé).

### Problème du frottement sec

En mode élastique, les manettes sont toujours ramenées en position neutre et la vitesse de déplacement dans l'EV est proportionnelle à l'angle mesuré sur la manette correspondante. La position neutre correspond à zéro degré. La manette ne revient cependant jamais exactement à zéro degré à cause du frottement sec de l'ensemble du système et ainsi l'objet manipulé garde une vitesse de déplacement lorsque l'utilisateur lâche les manettes.

Pour y remédier, nous avons mis en place le calcul d'une zone morte ( $\theta_{zm}$ ) autour de la position neutre fonction du couple de frottement sec ( $C_r$ ) et de la raideur de la manette ( $R$ ) :

$$\theta_{zm} = \frac{C_r}{R} \quad (3.1)$$

La vitesse de déplacement des objets est alors modifiée pour prendre en compte la zone morte (fig. 3.7). Tous ces réglages sont intégrés dans l'API du DigiHaptic. L'ajustement

du couple de frottements secs peut être effectué via l'interface graphique présentée en section 2.6.4 si des phénomènes de dérive apparaissent.

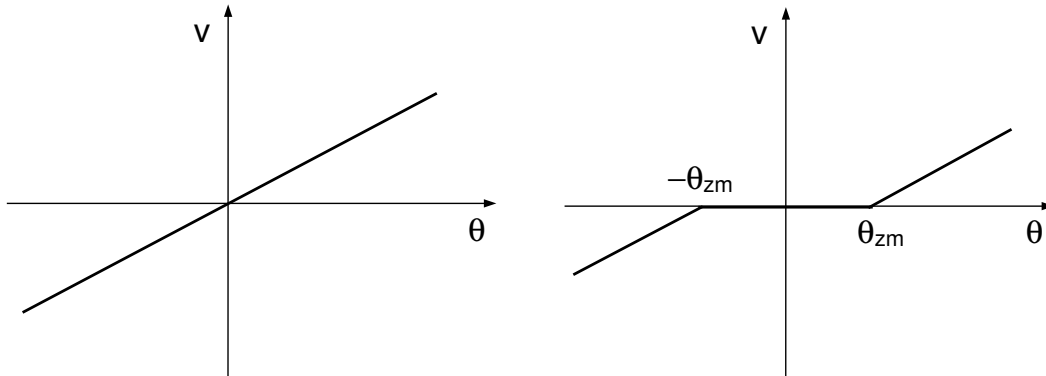


FIG. 3.7 – Vitesse de déplacement sans prise en compte du frottement sec (gauche) et avec prise en compte (droite).

### Question du retour d'effort

Le retour d'effort en mode isotonique ne pose pas de problème, le vecteur force calculé dans l'environnement virtuel est directement renvoyé à un facteur près sur l'effecteur du périphérique. Par contre, renvoyer des forces en mode élastique pose la question de la perception des forces par l'utilisateur. En effet, en mode élastique, le périphérique a toujours une force de rappel qui ramène l'effecteur en position neutre. Comment renvoyer la force calculée dans l'environnement virtuel pour que l'utilisateur puisse faire la distinction entre la force de rappel du ressort et celle de l'environnement virtuel ? C'est ce que nous verrons dans le paragraphe 3.5.

### 3.3.3 Passage entre mode isotonique et mode élastique

Pour utiliser les modes isotonique et élastique dans une même application, nous pouvons imaginer passer d'un mode à l'autre. Si les manettes sont en mode élastique, l'utilisateur contrôle la vitesse de déplacement de l'objet qu'il manipule. Pour passer en mode isotonique, l'utilisateur appuie sur un bouton après avoir relâché les manettes. Une fois en mode isotonique, il contrôle la position de l'objet dans un cube de travail dont le centre est la position de l'objet au moment du changement de mode. En mode isotonique, la position nulle des manettes correspond au centre du cube de travail.

Dans ce cas (passage de mode élastique à isotonique), les manettes sont en position neutre lors du changement de mode et il n'y a donc pas de discontinuité dans le mouvement du cube. Si l'on souhaite maintenant passer du mode isotonique au mode élastique, les manettes ne sont pas forcément en position nulle et l'objet manipulé aura par conséquent une translation instantanée non voulue, le temps que les manettes reviennent en position nulle. Ce phénomène va s'accompagner d'une application brutale des ressorts sur les manettes, ce qui est également gênant.



Il est envisageable d'attendre que les manettes reviennent en position neutre avant de bouger l'objet mais cette technique peut perturber l'utilisateur qui garderait ses doigts appuyés sur les manettes et les empêcherait ainsi de revenir en position nulle. L'objet resterait alors immobile. Généralement, on peut dire qu'avec un périphérique à retour de force, l'utilisation des degrés de liberté tour à tour en modes isotonique et élastique n'est pas possible.

### 3.3.4 Résumé des modes d'utilisation

La figure 3.8 résume les possibilités de passages entre mode isotonique et mode élastique, contrôle des translations et contrôle des rotations d'objets.





	DigiHaptic en mode isotonique absolu	DigiHaptic en mode élastique
translations d'objets		
rotations d'objets		

FIG. 3.8 – Représentation des possibilités de passage entre les modes isotoniques et élastique, contrôle des translations et rotations d'objets.

En mode isotonique, nous avons vu qu'il n'est pas possible de passer du contrôle des translation au contrôle des rotations. Nous n'utiliserons donc jamais le périphérique en mode isotonique pour le contrôle des rotations. En mode élastique, le passage de l'un à l'autre est possible. Le passage entre mode isotonique et élastique n'est possible que dans un sens. Nous allons voir dans le prochain paragraphe comment utiliser les deux modes dans une application.

En complément du tableau de la figure 3.8, nous pouvons représenter les degrés de liberté contrôlés dans chacun des cas (fig. 3.9). Pour le contrôle des translations, les trois degrés de liberté séparés du DigiHaptic (traits discontinus) utilisés soit en mode isotonique ou en mode élastique contrôlent (doubles traits) les translations des trois degrés de liberté intégrés de l'objet manipulé (traits continus). De plus il y a isomorphisme d'orientation et de direction pour chaque degré de liberté. Pour le contrôle des rotations, seul le mode élastique est utilisé. Dans ce cas, nous avons les isomorphismes d'orientation, de direction et de déplacement pour le pouce et l'index mais pas pour l'annulaire. Pour ce dernier, dont l'axe de rotation est identique à celui de l'index, l'absence isomorphisme rend son utilisation beaucoup moins intuitive.

### 3.3.5 Combinaison mode isotonique et mode élastique

Nous nous plaçons dans le cas d'une tâche de pointage où l'on souhaite contrôler un pointeur en position avec un périphérique isotonique à retour de force. Nous souhaitons par ailleurs avoir la possibilité de réaliser cette tâche dans un espace de travail de grandes

	linéaire			rotatif			
	X	Y	Z	rX	rY	rZ	
P							R
dP							dR
F							T
dF							dT
	1 10 100 Inf	1 10 100 Inf	1 10 100 Inf	1 10 100 Inf	1 10 100 Inf	1 10 100 Inf	

FIG. 3.9 – Résumé des modes d'utilisation du DigiHaptic (voir §1.2.1).

dimensions sans avoir recours au débrayage par l'usage d'un bouton. Le problème est par conséquent de s'affranchir de l'espace de travail limité des périphériques isotoniques.

L'idée est de déplacer le volume de travail accessible dans l'environnement virtuel en utilisant le mode élastique. Plutôt que d'utiliser exclusivement le mode isotonique ou le mode élastique, nous imaginons utiliser les deux modes à la fois en définissant une plage d'utilisation du mode isotonique et à l'extrémité de celle-ci définir la zone élastique (fig. 3.10).

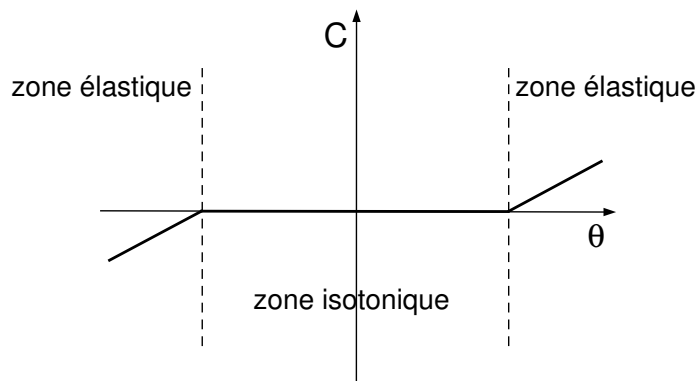


FIG. 3.10 – Courbe du couple sur la manette en fonction de sa position.

### Métaphore du cube simple avec contrôle hybride isotonique élastique

Avec un tel mode d'utilisation, il n'y a plus de discontinuité lors du passage d'un mode à l'autre. Le mode isotonique définit alors un cube de travail isotonique dans l'environnement virtuel, représenté par le cube transparent (fig. 3.11). Le cube transparent représente à tout moment le volume de travail accessible par l'utilisateur en mode isotonique. La position nulle des manettes correspond à tout moment au centre du cube de

travail isotonique. Lorsque le pointeur rouge contrôlé par l'utilisateur est dans le volume de travail isotonique, le cube transparent n'est pas représenté. Il pourrait en effet gêner la vue du pointeur et des objets à manipuler. La longueur de l'arête du cube peut être définie suivant le control display optimal du périphérique. Dès que le pointeur sort du cube<sup>26</sup>, il est visuellement projeté sur la face correspondante du cube. Sur la figure, nous avons volontairement représenté le pointeur à l'extérieur du cube par une sphère rouge transparente, afin d'indiquer la direction de déplacement du cube. A l'usage, la sphère rouge transparente n'est pas représentée. Ainsi le pointeur rouge ne quitte jamais le cube transparent. Lorsque le pointeur touche la paroi du cube, la manette correspondante passe en mode élastique et permet le déplacement en vitesse du cube. Plus l'utilisateur appuie sur la manette en mode élastique, plus la vitesse de déplacement est élevée. Lorsque l'utilisateur a placé le volume de travail isotonique à l'endroit désiré, il peut alors repasser en mode isotonique en relâchant la manette correspondante. Nous appelons cette métaphore : *métaphore du cube simple avec mode hybride isotonique élastique*.

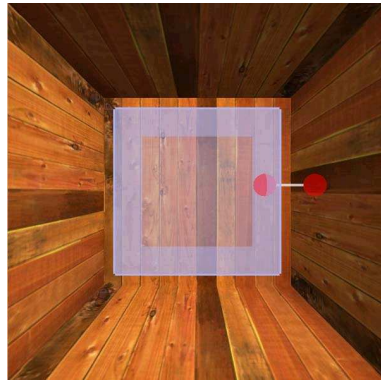


FIG. 3.11 – Métaphore du cube simple avec mode hybride isotonique élastique.

Lorsque l'utilisateur touche une paroi du cube, la manette correspondante va passer en mode élastique alors que les deux autres manettes vont rester en mode isotonique. Il y a alors une différence de perception entre les trois manettes qui peut perturber l'utilisateur. Pour éviter ce problème, nous avons développé les métaphores du *cube glissant avec mode hybride isotonique élastique*, du *cube collant avec mode hybride isotonique élastique*, de la *sphère glissante avec mode hybride isotonique élastique* et de la *sphère collante avec mode hybride isotonique élastique*.

Ces métaphores ont été mises en applications sur le DigiHaptic. Elles sont valables pour tout périphérique haptique que l'on souhaite utiliser dans des espaces de travail de grandes dimensions, sans avoir besoin de recourir au débrayage du périphérique par l'intermédiaire d'un bouton. Nos recherches bibliographiques n'ont mis à jour aucune de ces métaphores. L'idée générale du passage entre mode isotonique et élastique est évoquée très brièvement dans l'article [HS93]. La mise œuvre n'est pas expliquée.

---

<sup>26</sup>Il est préférable de définir un temps minimal au delà duquel le cube transparent apparaît, afin d'éviter qu'il n'apparaisse par intermittence.

### Métaphore du cube avec mode hybride isotonique élastique

L'idée est de faire passer toutes les manettes en mode élastique lorsque le pointeur touche une des faces du cube. La position angulaire de la manette correspondant au passage entre mode isotonique et mode élastique est alors calculée dynamiquement, contrairement à la situation précédente où elle était fixe. Lorsque le pointeur sort du cube, nous calculons le point d'intersection entre la face correspondante et le segment défini par le centre du cube et le centre du pointeur (fig. 3.12). Le pointeur est ensuite projeté sur la face correspondante du cube au point d'intersection. La direction définie par ce point d'intersection et le centre du pointeur correspond à la direction de déplacement du cube. La vitesse de déplacement est proportionnelle à la norme de ce vecteur. Celui-ci est également envoyé sur le périphérique. Les manettes passent alors en mode élastique.

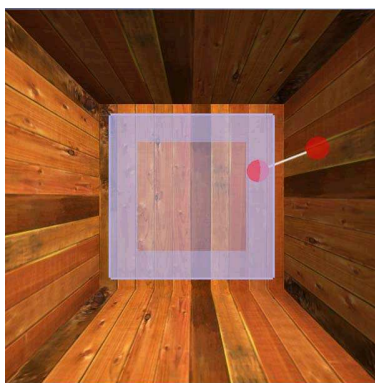


FIG. 3.12 – Métaphore du cube glissant avec mode hybride isotonique élastique.

L'utilisateur a maintenant les mêmes sensations sur chacune des manettes. Quand l'utilisateur change l'orientation du vecteur, le pointeur va glisser sur la paroi du cube. Si l'utilisateur déplace le cube de travail horizontalement et souhaite ensuite le déplacer verticalement, il va devoir soit glisser sur les parois du cube pour passer d'une face à l'autre, soit aller à l'intérieur du cube pour changer de face.

Pour éviter la perte de temps correspondante, nous avons défini la métaphore du *cube collant avec mode hybride isotonique élastique*. Dès que l'utilisateur entre en contact avec une des parois du cube, le pointeur reste collé à la surface tant que l'utilisateur est en mode élastique. Cela consiste simplement à sauvegarder la position du point de projection dès qu'il y a contact et tant qu'il y a contact. Les directions de déplacement peuvent alors être plus inclinées puisque la direction du vecteur de déplacement n'est plus définie par le centre du cube et la position du pointeur (fig. 3.13).

La première métaphore présentée nous a conduit à définir le volume de travail sous forme de cube. Nous avons ensuite imaginé définir le volume de travail sous forme de sphère, qui est une forme géométrique plus simple.

### Métaphore de la sphère avec mode hybride isotonique élastique

Nous avons défini la métaphore de la sphère par analogie à celle du cube. L'utilisation est identique, seule la forme du volume de travail change. La sphère étant plus régulière

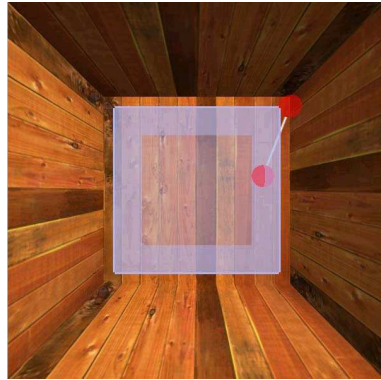


FIG. 3.13 – Métaphore du cube collant avec mode hybride isotonique élastique.

que le cube (il n’y a pas d’arêtes), nous pensons que la métaphore est plus intuitive. Elle évite aussi les discontinuités dans le mouvement du pointeur lors du passage d’une face à l’autre pour la métaphore du cube.

Lorsque l’utilisateur sort de la sphère isotonique, celle-ci apparaît (fig. 3.14). Le point d’intersection entre la sphère transparente et le segment défini par le centre de la sphère transparente et le centre de la sphère rouge, est calculé. Cela définit la position où le pointeur rouge est projeté. Le vecteur défini par le point d’intersection et le centre du pointeur transparent correspond à la direction de déplacement de la sphère transparente. Ce vecteur correspond également au vecteur force qui est envoyé sur le périphérique haptique. Dans ce cas, le périphérique passe en mode élastique (plus l’utilisateur cherche à sortir de la sphère transparente, plus la force appliquée sera importante) et un contrôle en vitesse du déplacement de la sphère transparente est réalisé. Le degré d’opacité de la sphère est proportionnel à la norme de ce vecteur pour que l’utilisateur ait un retour visuel supplémentaire sur la vitesse de déplacement (comme pour les métaphores précédentes). Cela permet également de réaliser une apparition progressive de la sphère.

De la même façon que pour le cube, nous pouvons définir la métaphore de la *sphère collante avec mode hybride isotonique élastique*.

Ces métaphores ont l’avantage de permettre à l’utilisateur de contrôler les objets en mode isotonique. Ce mode est en effet généralement utilisé dans les tâches de manipulation car les gestes réalisés correspondent à ceux de la vie quotidienne. Ces métaphores ont aussi l’avantage de toujours faire ressentir à l’utilisateur les forces en mode isotonique. En effet, lors du déplacement de la sphère isotonique, lorsque le pointeur entre en contact avec un objet, les forces de contact ramènent le périphérique en mode isotonique et les forces sont alors ressenties dans ce mode.

Les métaphores nécessitent d’être évaluées quantitativement et qualitativement dans des tâches de manipulation et comparée aux métaphores classiques de débrayage avec boutons, pour voir dans quelles mesures elles peuvent améliorer les performances de l’utilisateur. La métaphore qui nous semble la plus intéressante est celle de la *sphère glissante avec mode hybride isotonique élastique*. Le choix des retours visuels sera à évaluer lors des expérimentations. Entre autre, il faudra répondre à la question de la visualisation du

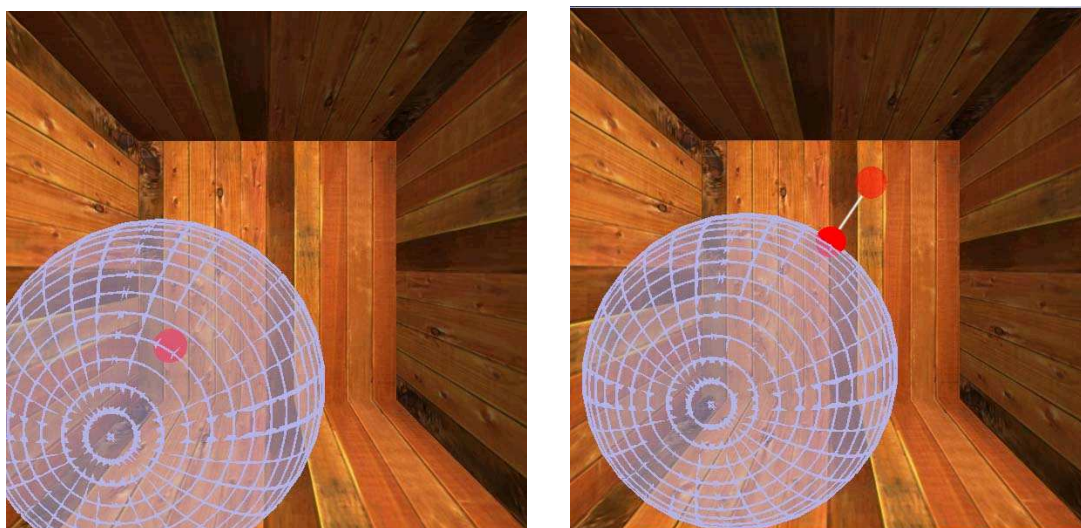


FIG. 3.14 – A gauche, le pointeur manipulé par l'utilisateur est à l'intérieur de la sphère (normalement la sphère n'est pas représentée dans cette situation). Les manettes sont en mode isotonique avec contrôle en position du pointeur. A droite lorsque l'utilisateur touche les parois de la sphère isotonique, celle-ci apparaît progressivement et les manettes passent en mode élastique avec contrôle en vitesse de la sphère transparente. Le pointeur transparent à l'extérieur de la sphère, non représenté en pratique, indique ici la direction de déplacement.

volume de travail lors de son déplacement en mode élastique.

Ces métaphores sont particulièrement intéressantes pour tous les périphériques isotoniques à retour d'effort où le volume de travail est limité. Pour les périphériques isotoniques à volume de travail limité sans retour d'effort tels que le DigiTracker développé au laboratoire [MPC04], des métaphores similaires peuvent être envisagées lorsque les limites du périphérique sont atteintes.

Nous venons de voir l'utilisation du DigiHaptic pour les tâches de manipulation en environnement 3D. Nous allons maintenant étudier comment utiliser le DigiHaptic dans les tâches de navigation en environnement 3D.

## 3.4 Interaction avec le DigiHaptic dans les tâches de navigation

Dans cette section, nous étudions comment utiliser le DigiHaptic dans des tâches de navigation en environnement 3D. Compte tenu du faible volume de travail du périphérique en mode isotonique, nous utilisons uniquement le mode élastique pour cette tâche.

### 3.4.1 Métaphore de contrôle de la caméra

L'idée est de contrôler trois des six degrés de liberté de la caméra avec le DigiHaptic en associant chaque manette à un degré de liberté de la caméra.

Nous utilisons la métaphore du *contrôle de véhicule* expliquée en section 3.2.2 en utilisant seulement trois degrés de liberté. Nous avons une nouvelle fois cherché à avoir les isomorphismes d'orientation et de direction pour chacun des degrés de liberté de la caméra. Ainsi, l'index qui a un mouvement d'avancer/reculer est utilisé pour avancer/reculer la caméra, le pouce qui a un mouvement gauche/droite oriente la caméra sur la gauche ou la droite (contrôle du lacet) et l'annulaire qui a un mouvement haut/bas sert à orienter la caméra en haut ou en bas (contrôle du tangage) mais peut, dans un autre mode, servir à translater la caméra suivant la direction haut/bas (fig. 3.15). Si l'annulaire sert à la translation de la caméra, son vecteur visée sera toujours orthogonal à la verticale de l'environnement 3D.

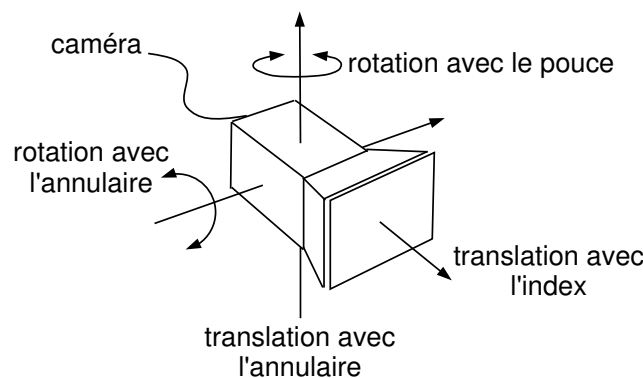


FIG. 3.15 – Mouvements des doigts et mouvements associés de la caméra. L'annulaire sert soit à la translation de la caméra suivant la direction haut/bas, soit au contrôle du tangage.

Pour mettre en œuvre cette métaphore de contrôle de la caméra, nous avons développé une application de navigation avec retour de force quand celle-ci vient en collision sur un mur. Écrire une application de contrôle de caméra dans un environnement 3D n'est pas en soit difficile. La difficulté est de décrire cet environnement pour qu'il soit graphiquement riche tout en restant fluide. Une autre difficulté est la gestion des collisions entre la caméra et l'environnement.

Décrire l'environnement avec des primitives OpenGL aurait été laborieux. Afin d'éviter le travail de création de l'environnement et disposer d'une gamme importante de mondes virtuels, nous avons choisi de créer un moteur capable de charger les fichiers au format BSP que l'on trouve dans la plupart des jeux de type First Person Shoot (FPS) comme Doom ou Quake. Le principe des arbres BSP, leur création, le rendu de la scène et la détection de collisions sont décrites en annexe B.

L'application est capable de détecter une collision de la caméra avec l'environnement et dans ce cas de mesurer la distance de pénétration de la caméra. Nous pouvons ainsi appliquer la méthode de retour de force en mode élastique décrite au §2.6.4. Nous renvoyons alors une force sur une, deux ou trois manettes suivant la façon dont la caméra arrive en collision sur le mur.

### 3.4.2 Calcul du retour de force sans glissement sur la surface

Lorsqu'il y a collision avec un mur, la force de réaction du mur se décompose en une force normale ( $\vec{R}_N$ ) et une force tangentielle ( $\vec{R}_T$ ). La force tangentielle correspond au frottement statique qui empêche la caméra de glisser sur la surface. La force de retour (calculée suivant la méthode décrite au §2.6.4), orientée selon le vecteur visée de la caméra, est envoyée sur l'index responsable du déplacement de la caméra dans cette direction (fig. 3.16).

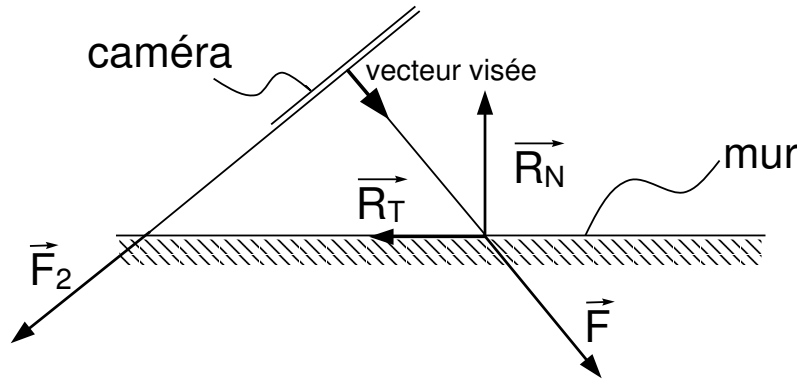


FIG. 3.16 – Collision de la caméra avec un mur.

Le rayon lancé sur le côté de la caméra va également rentrer en collision avec le mur. Celui-ci va générer une force  $\vec{F}_2$  qui sera renvoyée sur le pouce ou l'annulaire en fonction du rayon en collision.

### 3.4.3 Glissement sur les surfaces

Avec la méthode précédente, quand l'utilisateur arrive avec la caméra sur un plan, la caméra s'arrête et un retour de force est généré. Il faut alors faire marche arrière pour continuer sa route. Il serait plus naturel de glisser sur la surface plus ou moins rapidement suivant l'angle d'attaque du plan.

Soit  $\vec{vVector}$  le vecteur représentant le vecteur de déplacement souhaité par l'utilisateur (le vecteur orthogonal au plan de la caméra),  $\vec{mur}$  le vecteur orthogonal au plan du mur et  $\vec{sVector}$  le vecteur de glissement de la caméra.  $\vec{sVector}$  est orthogonal à  $\vec{mur}$  et correspond à la projection de  $\vec{vVector}$  sur le plan du mur. Ainsi  $\vec{sVector}$  est égal à :

$$\vec{sVector} = \vec{vVector} - \frac{\vec{vVector} \cdot \vec{mur}}{\vec{mur} \cdot \vec{mur}} \vec{mur} \quad (3.2)$$

## 3.5 Le retour d'effort en mode élastique

### 3.5.1 Introduction

Habituellement le retour de forces est uniquement utilisé avec les périphériques isotoniques tels que le Phantom. Pour les périphériques élastiques, nous proposons de rendre





FIG. 3.17 – Capture écran de l'application de navigation.

les forces calculées dans l'environnement virtuel [CPCS03a][CC03].

On peut toutefois noter que les joysticks à retour de forces permettent de rendre des forces calculées dans les jeux [JL99]. Cependant les forces rendues correspondent à des effets pré-programmés dans le périphérique qui sont mis à jour à basse fréquence et non à des forces directement calculées dans le jeu puis envoyées au périphérique à haute fréquence.

Le retour de force en mode élastique a également été mis en oeuvre sur le Magic Wrist, périphérique magnétique à retour de force utilisé pour le positionnement précis/grossier en téléopération [HS93]. Compte tenu des actionneurs utilisés, le périphérique n'autorise qu'un faible débattement de l'effecteur qui est utilisé en modes isotonique/élastique. Le mode isotonique est défini autour de la position centrale de l'effecteur et le mode élastique est défini aux extrémités de l'espace de travail. On a ainsi un mode d'utilisation similaire à celui décrit en section 3.3.5. Quand l'utilisateur est en mode élastique avec contrôle en vitesse et qu'il reçoit des forces du bras esclave, la force de retour ramène le dispositif en zone isotonique et les forces sont donc ressenties dans ce mode.

Nous nous intéressons au retour de force en mode exclusivement élastique. Nous avons vu comment simuler un mur en mode élastique dans le chapitre précédent dans le cas de la collision entre corps rigides. Nous allons maintenant voir comment simuler des forces qui correspondent à des phénomènes qui génèrent des forces plus faibles, comme c'est le cas dans le cas de la collision entre corps déformables.

### 3.5.2 Analyse

Sur le DigiHaptic en mode élastique, la force  $F$  à l'extrémité de la manette est proportionnelle ( $k_0$ ) à son déplacement (l'angle de la manette  $\theta$ ) comme décrit dans l'équation (3.3). Une position de référence où  $\theta = 0$ , encore appelée position neutre, est placée en général à équidistance des limites de la manette. Cela permet de toujours ramener la manette dans sa position neutre.

$$F = k_0 \theta \quad (3.3)$$

Pour introduire les forces calculées par l'environnement virtuel, il est possible de les insérer dans le coefficient de raideur pour obtenir une raideur fonction de la force (3.4) ou d'ajouter un terme à côté du terme élastique (3.5).

$$F = \text{fonction}(\text{force}) \times \theta \quad (3.4)$$

$$F = k_0 \theta + \text{fonction}(\text{force}) \quad (3.5)$$

Les équations (3.4) et (3.5) décrivent deux modes que nous avons respectivement appelés maître et esclave.

### 3.5.3 Mode maître

Les résultats qui suivent sont l'application des travaux réalisés sur un contrôleur de vitesse avec retour de force sur la raideur, appliqué au contrôle d'une grue pour sentir les forces exercées à l'extrémité du godet [PLS96].

Les contraintes pour définir la fonction (3.4) sont les suivantes :

- Besoin d'un terme constant de raideur pour garder une raideur lorsqu'aucune force n'est appliquée.
- Les forces positives doivent augmenter la raideur et les négatives la diminuer. Les forces positives sont celles qui s'opposent au déplacement de l'objet manipulé quand  $\theta$  est positif.
- Plus l'objet manipulé arrive rapidement sur un obstacle, plus la variation de force doit être importante.

Ainsi la fonction (3.4) nécessite l'ajout d'un terme constant et d'un terme dépendant du signe des forces rendues et proportionnel à la distance entre la position neutre de la manette et la position à l'instant de collision.

Soit  $k$  la raideur variable,  $k_0$  la raideur constante,  $f$  la force calculée par l'environnement virtuel et  $a$  un facteur d'échelle entre l'environnement virtuel et le périphérique.  $F$  est la force à l'extrémité de la manette et  $\theta$  la position angulaire de celle-ci.

$$k = k_0 + a \text{ signe}(\theta) f \quad (3.6)$$

$$F = (k_0 + a \text{ signe}(\theta) f) \theta \quad (3.7)$$

Le terme  $a f$  définit la plus simple des lois. Il est en effet possible d'employer des lois logarithmiques ou exponentielles pour jouer sur les effets qualitatifs suivant la tâche. Par ailleurs, la même zone morte peut être appliquée dans ce mode que celle définie en section 3.3.2.

Avec l'équation (3.7), l'objet manipulé ne peut pas bouger quand la manette est en position neutre même quand des forces sont appliquées sur l'objet. Par exemple, l'utilisateur ne peut pas ressentir le poids d'un objet s'il ne bouge pas. Ainsi la position de l'objet manipulé dépend uniquement de la volonté de l'utilisateur de bouger cet objet, c'est pourquoi nous avons appelé ce mode maître.

### 3.5.4 Mode esclave

Ce mode consiste à ajouter un terme dépendant de la force au terme élastique. Ce terme doit être indépendant de  $\theta$  sinon nous nous replaçons dans le cas du mode maître. Nous proposons d'ajouter un terme proportionnel à la force calculée dans l'environnement virtuel (3.8).  $a f$  est le terme qui est envoyé à un périphérique isotonique. L'équation (3.8) correspond donc à la somme de la force de rappel du ressort du mode élastique et de la force qui serait envoyée au périphérique en mode isotonique.

$$F = k_0 \theta + a f \quad (3.8)$$

Comme en mode maître,  $a f$  est le plus simple des termes qui peut être ajoutée en mode esclave mais on peut imaginer définir des fonctions qui atténuent ou augmentent les forces calculées par l'environnement virtuel suivant leur amplitude.

Avec ce mode, l'objet manipulé peut maintenant bouger quand la manette est en position neutre et que des forces non nulles sont exercées sur l'objet. C'est pour cette raison que nous avons appelé ce mode esclave.

Les constantes  $k_0$  et  $a$  doivent être choisies de manière à obtenir un bon rapport entre l'effet élastique et les forces ajoutées de telle sorte que l'utilisateur soit en mesure de discriminer les forces de l'environnement virtuel de l'effet élastique.

Le terme  $a$  est un facteur d'échelle qui dépend de l'environnement virtuel. Si l'environnement virtuel calcule des forces énormes,  $a$  devra être petit (ex : manipulation d'avions). Il devra être grand si l'environnement virtuel calcule de faibles forces (exemple : manipulation de molécules).  $k_0$  doit également être choisi en fonction du couple maximal applicable sur les manettes, pour être en mesure de rendre des forces quand les limites de la manette sont atteintes.

A nouveau, la même zone morte que celle définie au §3.3.2 peut être appliquée.

### 3.5.5 Comparaison des modes maître et esclave

Après avoir discuté des spécifications pour les modes maître et esclave, nous pouvons en dégager les caractéristiques principales suivantes :

- En mode maître, les forces ne peuvent pas être ressenties lorsque les manettes sont en position neutre. En dehors de la position neutre, les forces sont principalement ressenties sur le doigt contrôlant la direction de déplacement la plus rapide. Une force donnée n'est pas rendue de la même façon quelque soit l'angle de la manette.
- En mode esclave, les forces sont rendues de la même façon quelque soit l'angle, même en position neutre. Il est donc plus facile de discriminer la force générée par l'environnement virtuel de la force de rappel du ressort.

Les deux modes ont été mis en oeuvre dans la même démonstration que celle présentée en section 3.3.1 avec Spore pour le mode isotonique. Pour cette démonstration, les sensations sont qualitativement meilleures pour le mode esclave.

Nous venons de proposer deux solutions pour rendre les forces calculées par l'environnement virtuel en mode élastique. Ces méthodes sont particulièrement intéressantes pour la simulation d'environnements non rigides. Dans le cas où l'environnement est complètement rigide, nous utilisons la méthode de simulation de murs en mode élastique présentée au paragraphe 2.6.4.

## 3.6 Conclusion

Dans ce chapitre, nous avons passé en revue les modes d'utilisation du DigiHaptic en lien avec le type de tâche. Le DigiHaptic peut être utilisé dans deux principaux modes : le mode isotonique et le mode élastique. Le mode isotonique permet de faire un contrôle en position des objets et le mode élastique, un contrôle en vitesse. Nous avons regroupé les tâches en deux familles : les tâches de manipulation et les tâches de navigation.

Nous avons dans un premier temps analysé l'utilisation du DigiHaptic dans les tâches de manipulation. En mode isotonique, nous avons vu l'importance des isomorphismes d'orientation et de direction de chacune des manette pour appliquer le vecteur force calculé par l'environnement sur les manettes et nous avons conclu que le périphérique ne peut être utilisé que pour réaliser des translations. En mode élastique, nous avons vu qu'il est possible d'utiliser le périphérique aussi bien pour les translations ou les rotations d'objets et qu'il est possible de passer de l'un à l'autre sans problème de discontinuité dans le mouvement de l'objet manipulé.

Pour passer du mode isotonique au mode élastique, nous avons observé des problèmes de discontinuité dans le mouvement de l'objet manipulé. Nous avons donc proposé différentes métaphores de cubes et de sphères qui permettent de réaliser un contrôle hybride isotonique élastique. Les objets sont manipulés en mode isotonique dans un volume de travail correspondant à un cube ou une sphère. L'utilisateur peut déplacer le volume de travail isotonique dans l'environnement en utilisant le mode élastique. Ces métaphores, applicables à tout périphérique haptique, sont particulièrement intéressantes quand le volume de travail du périphérique est restreint.

Nous nous sommes ensuite intéressés aux tâches de navigation en environnement 3D. Nous avons présenté la métaphore de navigation avec le DigiHaptic qui respecte à nouveau les isomorphismes d'orientation et de direction. Nous avons ensuite montré comment renvoyer des forces sur les manettes du périphérique lorsque la caméra entre en collision avec l'environnement. C'est la première fois qu'un périphérique haptique est utilisé en navigation avec retour de force actif.

Nous avons enfin étudié les possibilités de retour de force en mode élastique pour la simulation de collisions entre corps déformables. Ainsi le mode que nous définissons comme "esclave" apparaît comme une bonne solution pour permettre à l'utilisateur de discriminer la force de rappel du ressort de la force rendue dans l'environnement virtuel. Ce travail, applicable à tout périphérique à retour de force, a fait l'objet de deux publications [CPCS03a] [CC03].

Nous allons à présent évaluer certaines de ces tâches. En particulier, nous nous intéressons au contrôle des translations en mode élastique et à la navigation en mode élastique.

Pour cela, nous allons évaluer et comparer le DigiHaptic en mode élastique à la SpaceMouse, dans une application de suivi de trajectoires 3D et dans une application de navigation dans un tunnel 3D.



# Chapitre 4

## Evaluation du DigiHaptic

### Sommaire

---

<b>4.1</b>	<b>Introduction</b>	<b>87</b>
<b>4.2</b>	<b>Outils d'évaluation</b>	<b>88</b>
4.2.1	Loi de Fitts	88
4.2.2	Bandes passantes	90
4.2.3	La loi de suivi de trajectoires	91
4.2.4	Coordination	92
4.2.5	Apprentissage	93
<b>4.3</b>	<b>Présentation des évaluations</b>	<b>93</b>
<b>4.4</b>	<b>Evaluation en suivi de trajectoires 3D</b>	<b>94</b>
4.4.1	Outils	94
4.4.2	Méthode	95
4.4.3	Résultats et discussion	97
4.4.4	Questionnaires	101
4.4.5	Conclusion	101
<b>4.5</b>	<b>Evaluation en navigation avec retour d'effort</b>	<b>102</b>
4.5.1	Méthode	102
4.5.2	Résultats	104
4.5.3	Discussion et conclusion	111
<b>4.6</b>	<b>DigiHaptic version 2</b>	<b>112</b>
4.6.1	Présentation de la V2	112
4.6.2	Autre design	112
<b>4.7</b>	<b>Conclusion</b>	<b>113</b>

---

## 4.1 Introduction

Avant de présenter nos deux expérimentations d'évaluation, nous allons faire un résumé bibliographique des outils d'évaluation des périphériques.

Card et al. proposent d'évaluer les périphériques selon deux critères : l'expressivité (le périphérique transmet l'intention de l'utilisateur avec exactitude) et l'efficacité (le périphérique transmet l'intention de l'utilisateur de façon appropriée à la tâche) [CMR91].

En ce qui concerne l'expressivité, un problème peut survenir quand le nombre d'éléments de l'espace d'arrivée ne correspond pas à l'ensemble des éléments de l'espace de départ. L'utilisateur peut alors spécifier des valeurs qui n'existent pas dans l'espace d'arrivée ou à l'inverse être incapable de spécifier certaines valeurs de l'espace d'arrivée. Pour prendre l'exemple de la souris, si la résolution est très mauvaise, l'utilisateur sera incapable de pointer un certain nombre de pixels à l'écran et nous pouvons parler dans ce cas de perte d'expressivité.

Quant à l'efficacité, les auteurs définissent plusieurs critères basés sur des considérations de mesures de performances et des considérations pragmatiques :

- La rapidité de réalisation de la tâche. Par exemple, dans une tâche de pointage, le temps nécessaire pour pointer un objet. Celui-ci va directement dépendre de la bande passante du périphérique.
- Précision de réalisation de la tâche mesurée par le nombre d'erreurs.
- Temps d'apprentissage.
- Préférences de l'utilisateur.
- Fatigue de l'utilisateur.
- Temps pour saisir le périphérique. Par exemple, le temps pour mettre un gant de données est beaucoup plus long que le temps nécessaire pour saisir une souris.
- Encombrement du périphérique.
- Coût.

Nous allons étudier plus en détail certains de ces paramètres.

L'évaluation de périphériques est un sujet difficile puisqu'il met en jeu des utilisateurs humains. Les critères mesurés le plus souvent pour évaluer les périphériques sont la rapidité et la précision. La rapidité est mesurée par le temps de réalisation de la tâche. La précision est en général mesurée par le taux d'erreurs. Par exemple, pour une tâche de pointage, c'est le pourcentage de fois où le pointeur sélectionne en dehors de la cible.

L'évaluation des périphériques informatiques est généralement faite dans des tâches de pointage en environnements 2D, pour lesquels des outils quantitatifs existent. Nous verrons également qu'il est possible d'évaluer les périphériques dans des tâches de suivi de trajectoires.

## 4.2 Outils d'évaluation

### 4.2.1 Loi de Fitts

Très peu d'outils quantitatifs existent en recherche sur les interfaces homme-machine. La loi de Fitts est une des rares exceptions [Fit54]. Paul Fitts a en effet montré qu'il existe une relation formelle qui modélise le compromis vitesse/précision lors des mouvements de pointage. La loi prédit que le temps  $T$  nécessaire pour pointer une cible de largeur  $W$



située à une distance  $A$  suit une relation logarithmique (eq. 4.1).

$$T = a + b \log_2 \left( \frac{A}{W} + c \right) \quad (4.1)$$

où  $a$  et  $b$  sont des constantes déterminées empiriquement et  $c$  vaut 0 (pour la loi de Fitts originelle), 0.5 (Welford [Wel68]) ou 1 (Shannon [MAc89]). La formulation de Shannon est généralement adoptée car elle montre les taux de corrélation les plus importants [MAc89]. Le facteur  $\log_2 \left( \frac{A}{W} + c \right)$  est appelé indice de difficulté de la tâche (ID) et décrit le degré de difficulté de la tâche : plus l'ID est grand, plus la tâche est complexe.

La loi de Fitts a montré sa robustesse et sa précision dans de nombreuses tâches. Elle est particulièrement appréciable pour l'étude des périphériques informatiques. Sans la loi de Fitts, le temps de réalisation d'une tâche n'a de sens que dans des conditions expérimentales bien spécifiques. Avec la loi de Fitts, il est possible de calculer un indice de performance indépendamment des conditions expérimentales.

La loi de Fitts a le défaut de n'avoir été définie que pour des tâches de pointage monodimensionnelles alors que la plupart des tâches de pointages se font en 2D. Dans ce cas, quelle largeur considérer pour l'objet si celui-ci n'est pas circulaire ? MacKenzie propose de prendre pour la largeur de l'objet la plus petite de ses dimensions (en particulier pour un objet rectangulaire :  $\min(W, H)$ ) ou encore de calculer une largeur  $W'$  suivant l'angle d'attaque de l'objet  $\theta$  [MB92] (fig. 4.1).

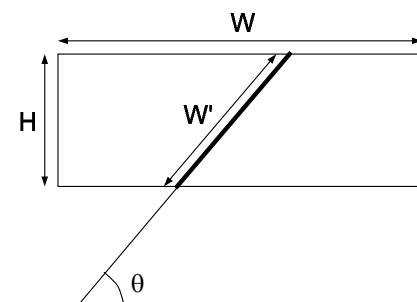


FIG. 4.1 – Choix de la largeur de l'objet pour l'application de la loi de Fitts dans les tâches 2D (d'après [MB92]).

Un autre défaut est de ne prendre en compte que les essais réussis. Un taux d'erreur est généralement reporté à côté des résultats de la loi de Fitts. Pour remédier à ce problème, Douglas, Kirkpatrick et MacKenzie proposent une variante de la loi de Fitts tenant compte des erreurs de pointage [DKM99]. Pour cela, ils définissent un nouvel indice de difficulté  $ID_e$  (eq. 4.2) où  $W_e$  est la largeur effective de la cible qui correspond à la largeur de la distribution des coordonnées de sélection pendant une séance d'essais (eq. 4.3), en prenant en compte tous les essais.  $x_i$  et  $y_i$  correspondent aux coordonnées du point sélectionné au  $i$ -ème essai des  $n$  essais.

$$ID_e = \log_2 \left( \frac{A}{W_e} + c \right) \quad (4.2)$$

$$W_e = 4.133 \times SD \quad (4.3)$$

$$SD = \sqrt{\frac{\sum_{i=1}^n [(x_i - \bar{x})^2 + (y_i - \bar{y})^2]}{n - 1}} \quad (4.4)$$

L'indice de performance est également modifié de manière à prendre en compte les performances de rapidité et de précision. Son nom a d'ailleurs changé pour devenir *débit*<sup>27</sup>. Il est calculé suivant la relation (4.5) où MT est le temps moyen de pointage en secondes pour tous les essais fait dans les mêmes conditions.

$$debit = \frac{ID_e}{MT} \quad (4.5)$$

Ces résultats sont utilisés dans la norme d'évaluation des périphériques de pointage ISO 9241-9.

Pour savoir si une tâche est difficile ou non suivant l'indice de difficulté, Card et al. définissent une sorte d' "étalon de difficulté". Dans leur échelle de difficulté, le pointage d'un mot avec une souris est une tâche relativement facile [CMR91]. En prenant la longueur moyenne d'un mot égale à 5,5 caractères, soit 2,3 cm et en prenant comme distance moyenne de pointage 9,7 cm, cela donne un indice de difficulté de 2,4 bits. Les tâches avec un indice de difficulté plus petit sont considérées faciles et les autres difficiles.

## 4.2.2 Bandes passantes

MacKenzie et al. [MKS01] ont étudié les débits de plusieurs périphériques de pointage selon la méthode décrite ci-dessus et ont trouvé les résultats du tableau 4.1, où il apparaît que la souris est le périphérique de pointage le plus efficace.

TAB. 4.1 – Bandes passantes des périphériques

Périphérique	débit (bits/sec)
souris	4,9
trackball	3,0
joystick	1,8
touchpad	2,9

Un périphérique est associé à un groupe de muscles qui le contrôle. La bande passante du groupe musculaire va déterminer la limite de la bande passante maximale du périphérique. Un périphérique bien conçu doit pouvoir exploiter toute la bande passante du groupe musculaire auquel il est associé.

Nous avons reporté dans le tableau 4.2 les bandes passantes de différents membres d'après les résultats obtenus par Balakrishnan et al. [BM97] et la norme ISO 9241-9, qui constituent les données les plus récentes à ce sujet. Les différences observées viennent sans doute des méthodes utilisées. En comparant ces données à celles du tableau 4.1, nous observons que la souris a une bande passante supérieure à chacun des membres. Cette différence peut s'expliquer par le fait que la souris met en jeu des mouvements de l'épaule, du bras et du poignet. La bande passante résultante est alors une combinaison des bandes passantes de chacun des membres.

<sup>27</sup> *Throughput* en anglais.

TAB. 4.2 – Bandes passantes des membres selon Balakrishnan et al. [BM97] et la norme ISO 9241-9.

membres	Bande passante (bits/sec)	
	Balakrishnan	ISO 9241-9
doigts	2,96	3
poignet	4,08	2
avant-bras	4,14	

### 4.2.3 La loi de suivi de trajectoires

Inspirée de la loi de Fitts, la loi de suivi de trajectoires<sup>28</sup>, présentée pour les trajectoires 2D, a été proposée par Accot et Zhai pour toutes les tâches où l'utilisateur doit se déplacer à l'intérieur d'un tunnel [AZ97] (fig. 4.2).

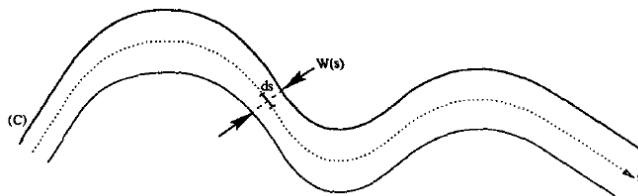


FIG. 4.2 – Exemple de suivi de trajectoire [AZ99].

Ces chercheurs ont montré que le temps de suivi de la trajectoire ( $T_C$ ) est proportionnel à l'indice de difficulté de la trajectoire ( $ID_C$ ) (eq. 4.6), où  $a$  et  $b$  sont des constantes.  $1/b$  est appelé indice de performance du parcours.  $ID_C$  est calculé en intégrant l'inverse de la largeur du parcours  $W(s)$  le long du parcours, avec  $s$  l'abscisse curviligne du parcours (eq. 4.7).

$$T_C = a + b \times ID_C \quad (4.6)$$

$$ID_C = \int_C \frac{ds}{W(s)} \quad (4.7)$$

Cela donne par exemple pour un chemin rectiligne de longueur  $A$  et de largeur  $W$ ,  $ID_C = A/W$  et pour un chemin circulaire de rayon  $R$  et de largeur  $W$ ,  $ID_C = 2\pi R/W$ . Précisons que la largeur du chemin à prendre en compte pour calculer l'indice de difficulté est la largeur du chemin moins la largeur du pointeur [NKK04].

Accot et al. ont ainsi pu comparer la souris, la tablette graphique, la trackball, le touchpad et trackpoint pour différents types de chemins [AZ99]. Ils ont montré que la souris et la tablette graphique sont les plus performantes.

<sup>28</sup>Steering law en anglais

#### 4.2.4 Coordination

La mesure de la coordination a pour but de mesurer la capacité des utilisateurs à contrôler plusieurs degrés de liberté en même temps. Plus les degrés de liberté seront coordonnés, plus le mouvement apparaîtra comme continu.

Zhai et al. ont proposé une méthode pour mesurer la coordination des périphériques six degrés de liberté [ZM98]. La méthode consiste à supposer que la coordination la plus importante est atteinte quand le chemin parcouru entre deux points est la ligne droite (fig. 4.3). La coordination est donc calculée ainsi :

$$Taux_{coordination} = \frac{\text{Longueur du chemin parcouru} - \text{Longueur du chemin le plus court}}{\text{Longueur du chemin le plus court}} \quad (4.8)$$

Plus  $Taux_{coordination}$  est proche de zéro, plus la coordination des degrés de liberté est importante. Ainsi, cette méthode mesure la faiblesse de la coordination.

Cette méthode donne une mesure globale de la coordination. Il est en effet possible d'obtenir un bon coefficient de coordination en utilisant un degré de liberté à la fois et en réalisant de petits déplacements suivant chaque degré de liberté. La distance supplémentaire parcourue sera alors petite comparé à la distance du chemin le plus court, pourtant à aucun moment les degrés de liberté n'auront été coordonnés.

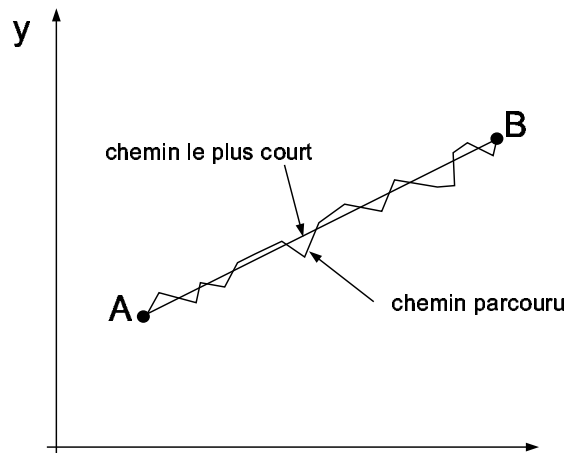


FIG. 4.3 – Illustration de la coordination de deux degrés de liberté (d'après [ZM98]).

Jacob et al. proposent de mesurer la coordination en regardant pour chaque intervalle de temps s'il y a eu un mouvement suivant tous les degrés de liberté [JSMM94]. Cela permet de déterminer un pourcentage de mouvements coordonnés. Si cette méthode permet de savoir localement s'il y a eu coordination, elle ne permet cependant pas de connaître l'efficacité de la coordination : est-ce que la coordination s'est faite de manière à se rapprocher de l'objectif ?

Pour prendre en compte la simultanéité et l'efficacité de coordination, Masliah et al. ont proposé une méthode de mesure appelée m-metric [MM00]. Le résultat est le produit d'un terme représentant la simultanéité de coordination et d'un autre terme représentant l'efficacité de coordination.

La simultan  it   est obtenue en calculant la fonction de r  duction d'erreur normalis  e. Suivant chaque degr   de libert  , la diff  rence entre la position    atteindre et la position actuelle est mesur  e    chaque instant. La variation de cette quantit   est calcul  e et norm  e par la distance totale parcourue suivant le degr   de libert   consid  r  . La plus petite variation suivant tous les degr  s de libert   est ensuite prise en compte pour chaque intervalle de temps. La somme de ces variations donne la valeur de simultan  it   de contr  le.

Le coefficient d'efficacit   est obtenu en comparant la r  duction d'erreur sur un intervalle    la r  duction d'erreur optimale sur cet intervalle. La somme de ces valeurs norm  es donne un coefficient compris entre 0 et 1. Une valeur proche de 1 indique une synchronisation efficace des degr  s de libert  .

### 4.2.5 Apprentissage

Le temps d'apprentissage d'un p  riph  rique correspond au temps    partir duquel les performances de l'utilisateur en terme de temps de r  alisation de la t  che et nombre d'erreurs n'  voluent plus. Ce temps d'apprentissage peut   tre plus ou moins long suivant le p  riph  rique.

Card et al. ont   tudi   le temps d'apprentissage pour la souris avec contr  le de position, le joystick avec contr  le de vitesse et le clavier [CEB78]. Ils ont mesur   le temps de pointage en fonction du nombre d'essais et ont v  rifi   qu'il suit la loi suivante :

$$T_N = T_1 N^{-\alpha} \quad (4.9)$$

o    $T_1$  est le temps moyen de pointage pour le premier groupe d'essais,  $T_N$  le temps moyen de pointage du n-i  me groupe d'essais,  $N$  le num  ro du groupe d'essais et  $\alpha$  une constante    d  terminer. Ainsi la facilit   d'apprentissage d'un p  riph  rique est caract  ris  e par les deux constantes  $T_1$  et  $\alpha$ . Plus  $\alpha$  est important, plus l'apprentissage am  liore les performances de l'utilisateur. Pour la t  che qu'ils ont propos  e, Card et al. montrent ainsi que l'effet de l'apprentissage pour la souris et le clavier am  liore davantage les performances de l'utilisateur que pour le joystick (tab. 4.3).

TAB. 4.3 – Coefficients des courbes d'apprentissage de diff  rents p  riph  riques (d'apr  s [CEB78]).

P��riph��rique	$T_1$	$\alpha$
Souris	2,20	0,13
Joystick	2,19	0,08
Clavier	3,86	0,15

## 4.3 Pr  sentation des   valuations

Apr  s avoir con  u et r  alis   le DigiHaptic, nous avons cher  h      l'  valuer dans des t  ches de suivi de trajectoire et de navigation en environnements 3D avec retour de force

pour, d'une part, voir si la coordination des degrés de liberté était facilement maîtrisable pour des utilisateurs novices et, d'autre part, analyser quel type de tâche permet d'exploiter au mieux la séparation des degrés de liberté.

Nous avons vu dans le chapitre précédent que le DigiHaptic en mode isotonique ne peut être utilisé que dans des environnements virtuels de tailles relativement limitées, à moins d'utiliser les métaphores que nous avons développées. Pour cette raison, nous avons uniquement évalué le DigiHaptic en mode élastique. Dans les deux tâches présentées, la SpaceMouse a servi de périphérique témoin. Celle-ci est en effet l'un des seuls périphériques élastiques commercialisés avec plus de deux degrés de liberté. La comparaison du DigiHaptic à la SpaceMouse permet ainsi d'évaluer l'influence de la séparation des degrés de liberté sur les périphériques élastiques.

La tâche de suivi de trajectoire a été choisie d'une part parce qu'il s'agit de contrôler un pointeur en translation et d'autre part parce qu'elle nécessite un contrôle permanent de la trajectoire. Une tâche de manipulation aurait requis le contrôle à la fois des translations et des rotations des objets. Le passage des translations aux rotations ne nous aurait pas permis d'évaluer la séparation des degrés de liberté à part entière. Quant à une tâche de pointage, l'absence de contrainte imposée à l'utilisateur pour atteindre la cible ne nous aurait pas permis d'évaluer précisément la coordination des degrés de liberté.

Pour la tâche de navigation, nous avons choisi un tunnel 3D plutôt qu'une tâche de navigation en environnement libre afin de disposer du maximum de données quantitatives pour évaluer l'effet de la séparation des degrés de liberté.

## 4.4 Evaluation en suivi de trajectoires 3D

Dans cette tâche, l'utilisateur manipule un pointeur sous différentes représentations le long de chemins rectilignes orientés dans différentes directions [CPC04]. Le DigiHaptic est utilisé en mode élastique et il est comparé à la SpaceMouse. Comme celle-ci est un périphérique élastique, nous cherchons à évaluer uniquement l'effet de la séparation des degrés de liberté. Nous avons aussi choisi la SpaceMouse pour son utilisation répandue dans les logiciels de CAO 3D, comme Catia [Cat] et 3D Studio Max [DDM], où elle est utilisée pour contrôler le point de vue de la caméra et manipuler les objets de la scène. C'est donc un périphérique élastique de référence.

### 4.4.1 Outils

#### Mesure de la coordination

Comme le DigiHaptic utilise des degrés de liberté séparés, nous souhaitons savoir si l'utilisateur les manipule séparément ou simultanément et comparer cette coordination à celle de la SpaceMouse qui a ses trois degrés de liberté assemblés.

Nous utilisons pour cela l'outil de mesure de la coordination proposé par Zhai et al. [ZM98] (§4.2.4).

## Loi de suivi de trajectoire

Afin de pouvoir comparer les périphériques, nous utilisons la loi de suivi de trajectoires proposée par Accot et al. [AZ97] (voir §4.2.3). Si la loi de suivi de trajectoire est vérifiée, nous pourrions ainsi comparer les indices de performances des deux périphériques.

### 4.4.2 Méthode

#### Présentation de la tâche

Il s'agissait de suivre des chemins dans un environnement 3D. Dans les expériences, on demandait aux utilisateurs de venir placer un pointeur 3D (anneau ou balle) au début d'un chemin puis de le parcourir.

Il y avait trois chemins rectilignes orientés dans différentes directions que nous appelons respectivement "A" (chemin qui va de l'avant bas de l'écran vers l'arrière haut de l'écran), "B" (chemin qui va de l'avant bas gauche de l'écran vers l'arrière haut droit de l'écran) et "C" (chemin qui va du haut du côté gauche vers le bas du côté droit) et trois représentations que nous nommons respectivement "1" (fil avec anneau), "2" (tube vide avec balle) et "3" (tube plein avec balle) (fig.4.4). L'anneau est orienté automatiquement perpendiculairement au chemin.

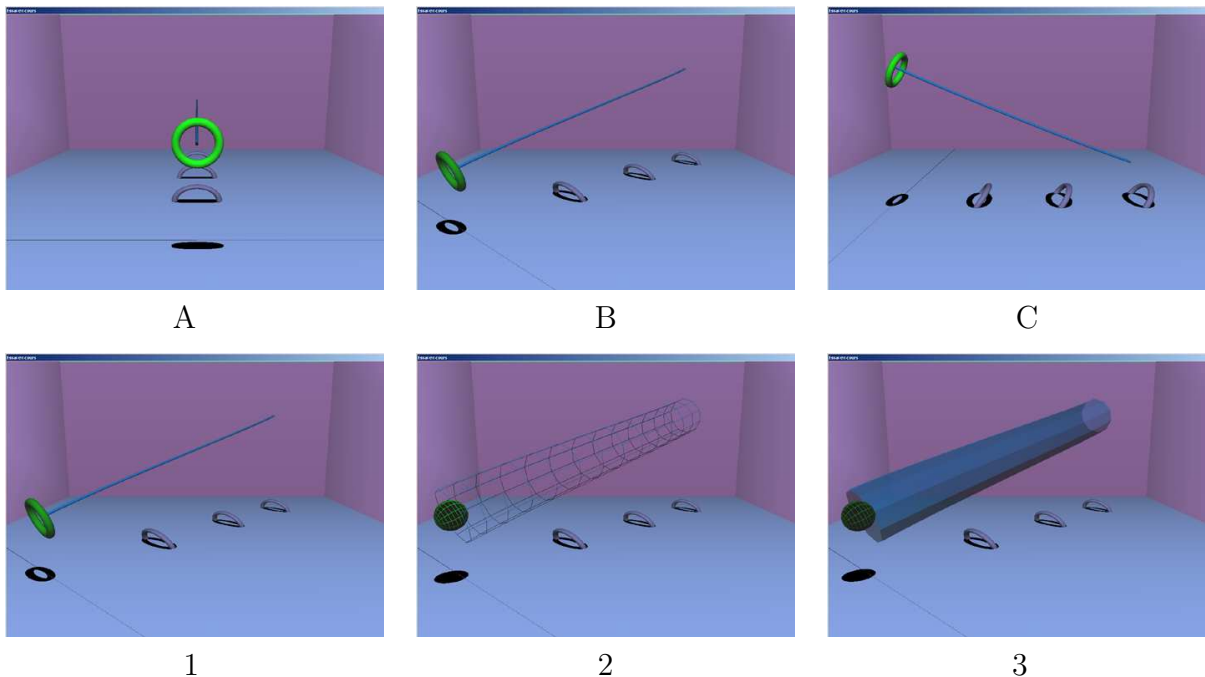


FIG. 4.4 – Les trois chemins droits (haut) et les trois orientations des chemins (bas)

Nous avons souhaité commencer par tester des chemins faciles d'avant d'essayer de plus compliqués. Nous voulions évaluer l'influence de la séparation des degrés de liberté au delà du temps d'apprentissage. C'est pour cette raison que nous avons choisi trois chemins rectilignes. L'orientation des chemins a été choisie pour manipuler deux ou trois degrés de liberté à la fois sur le périphérique et comparer ainsi les performances d'utilisation de

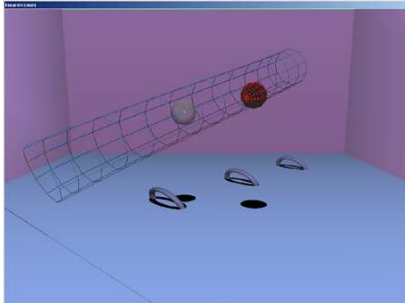


FIG. 4.5 – Le pointeur en rouge à l’extérieur du chemin et son fantôme en gris à l’intérieur du chemin pendant un essai après que l’utilisateur soit sorti.

anneau + fil	balle + tube	couleur du pointeur
		verte
		noire
		rouge

FIG. 4.6 – La couleur du pointeur dépend de sa position par rapport au centre du chemin

deux et trois degrés de liberté. Les différentes représentations servent à s’assurer que les résultats n’en dépendent pas.

Les expériences ont été conduites sur un PC de bureau. Lors de la création de l’environnement 3D, plusieurs types d’indices visuels ont été choisis comme la perspective linéaire, l’élimination des parties cachées, l’ombre du pointeur 3D et les anneaux au sol qui aident à percevoir la profondeur. Les anneaux au sol sont placés à 33%, 66% et 100% de la longueur du chemin et donne un indice visuel qui renseigne sur la hauteur du chemin. Nous n’avons pas utilisé de lunettes stéréoscopiques pour cette expérience.

Au début de chaque essai, le pointeur 3D (anneau ou balle) apparaît à une position aléatoire en rouge. On demande au sujet de l’amener au début du chemin en superposant le pointeur 3D à son fantôme en s’aidant de la ligne droite représentée au sol et de l’ombre du pointeur 3D au sol. Quand le pointeur 3D est en correspondance avec son fantôme, l’utilisateur commence à parcourir le chemin.

Pendant l’expérience, quand le centre du pointeur coïncide avec le centre du chemin, le pointeur devient vert. Ensuite, plus le pointeur s’écarte du centre du chemin, plus il devient sombre jusqu’à devenir noir quand il touche le bord du chemin (il y a alors contact) et rouge quand il sort des limites du chemin (fig. 4.6). Le fantôme représenté en gris transparent indique la dernière position du pointeur 3D à l’intérieur du chemin avant que l’utilisateur ne quitte celui-ci. Il aide l’utilisateur à savoir où il doit ramener le pointeur de façon à continuer le parcours du circuit (fig.4.5).

Les utilisateurs avaient pour consigne de minimiser leur temps de parcours et d’ajuster leur vitesse pour éviter de sortir du chemin.

## Sujets

Un total de 23 sujets a participé à l’étude. La plupart des participants (19) étaient droitiers et 4 étaient gauchers. Les participants ont été aléatoirement séparés en deux groupes, un pour chaque périphérique. Chaque sujet est passé sur un seul des périphériques. Tous les sujets ont utilisé leur main droite sur chacun des périphériques. Aucun des sujets n’était familier avec la tâche en question ou les périphériques étudiés.



## Procédure

Les sujets étaient assis en face de l'écran à une distance d'environ 70 cm. Les tests commencèrent après neuf essais d'entraînement. L'expérience complète durait une heure pour chaque sujet. Chaque combinaison de chemin et de représentation a été testée avec neuf essais, les essais étant présentés aléatoirement.

Les essais se déroulèrent comme décrit précédemment. La fin d'un essai était indiquée par un court bip et une boîte de dialogue indiquait le nombre restant d'essais à réaliser. Pendant ce temps, les sujets pouvaient prendre une pause avant de continuer en appuyant sur une touche.

Le nombre de contacts, les sorties, le temps de parcours, la longueur totale parcourue et les écarts par rapport au centre du chemin étaient enregistrés tous les 20 à 30 ms. Pour chaque essai, les enregistrements commencèrent après que le sujet avait placé le pointeur au début du chemin.

## Matériel

L'application correspondante à la tâche précédemment décrite a été développée en OpenGL sous Visual C++. Le moniteur avait une taille de 21 pouces réglé à une résolution de 1024x768. Comme périphérique d'entrée, nous avons utilisé une SpaceMouse Classic et un DigiHaptic utilisé en mode élastique. Pour ce dernier, des raideurs fixes ont été réglées sur les manettes pour tous les sujets. Aucun retour de force supplémentaire n'a été utilisé sur le périphérique. Enfin, pour assurer une comparaison équitable des deux périphériques, nous avons réglé les sensibilités de chaque périphérique avec deux sujets expérimentés sur chacun d'entre eux. Nous avons procédé à différents réglages de control display (voir partie 1 chapitre 2) et nous avons retenu celui qui demandait le moins de temps pour parcourir l'ensemble des circuits.

### 4.4.3 Résultats et discussion

#### Temps de parcours du circuit

L'indice de difficulté (ID) tel qu'il est décrit en partie 1 chapitre 1 dépend du chemin et de sa représentation. L'ID correspond au rapport de la longueur du chemin ( $A$ ) par sa largeur ( $W$ ) (table 4.4).  $W$  est la largeur du chemin libre pour bouger le pointeur, c'est donc par exemple pour la sphère et le tube, le diamètre du tube moins le diamètre de la sphère. Toutes les longueurs prises en compte correspondent aux dimensions des objets en unités OpenGL.

TAB. 4.4 – Tableau de correspondance entre l'ID des chemins et leurs représentation

	<b>1</b>	<b>2</b>	<b>3</b>
<b>A</b>	77	12	12
<b>B</b>	100	15	15
<b>C</b>	72	11	11

Comme les diamètres des tubes plein et vide ainsi que les diamètres des sphères sont égaux pour chaque chemin, les index de difficulté sont égaux pour les représentations "2" et "3". Pour la première représentation, le diamètre interne de l'anneau est plus faible que celui de la balle, ce qui explique l'ID plus important.

Inévitablement, il arrive que les participants sortent des limites du chemin. Dans les travaux de Zhai, tous les essais comportant des erreurs (sorties du chemin) sont ignorés, ce qui biaise les résultats. En effet, si nous prenons un périphérique rapide mais imprécis et un périphérique lent mais précis et que nous considérons seulement le temps de parcours, le second périphérique sera toujours considéré comme moins performant, même si les taux d'erreurs sont faibles pour celui-ci. Dans l'expérience, quand l'utilisateur sort du chemin, il doit ramener le pointeur à l'endroit où il est sorti, ce qui occasionne une pénalité de temps. De cette façon, nous obtenons une mesure globale qui permet de tenir compte de la rapidité et de la précision des sujets.

Ainsi, nous avons utilisé tous les essais pour tester la steering law<sup>29</sup>. Pour le DigiHaptic, la steering law est bien suivie avec de bons coefficients de régression. En voici les résultats pour chaque représentation (en ms) :

$$1 : T = -68174 + 1110 \times ID \quad r^2 = 0.999 \quad (4.10)$$

$$2 : T = -61416 + 6521 \times ID \quad r^2 = 0.999 \quad (4.11)$$

$$3 : T = -58621 + 6277 \times ID \quad r^2 = 0.999 \quad (4.12)$$

Nous constatons que l'indice de performance est meilleur pour l'anneau et le fil (représentation "1") alors que les deux autres représentations obtiennent des indices de performance équivalents. Par ailleurs, les chemins nécessitant l'utilisation de deux degrés de liberté<sup>30</sup> (chemins "A" et "C") ou trois degrés de liberté (chemin "B") suivent tous la steering law. On peut en déduire que l'utilisation de deux ou trois doigts n'a pas d'influence sur le temps de parcours, cela représente la même difficulté pour l'utilisateur.

En ce qui concerne la SpaceMouse, la steering law n'est pas respectée quelque soit la représentation du circuit :

$$1 : T = -24993 + 495 \times ID \quad r^2 = 0.744 \quad (4.13)$$

$$2 : T = -17382 + 2474 \times ID \quad r^2 = 0.666 \quad (4.14)$$

$$3 : T = -18266 + 2524 \times ID \quad r^2 = 0.745 \quad (4.15)$$

Pour analyser quelles en sont les raisons, nous allons calculer dans la prochaine section le coefficient de coordination pour chaque périphérique et chemin.

Les résultats d'une analyse de variance (ANOVA) correspondent aux précédentes analyses. L'analyse inclut comme condition les périphériques (DigiHaptic et SpaceMouse) avec mesures répétées sur les facteurs que sont les chemins et les représentations : 3 (chemins) \* 3 (représentations). Les chemins étaient répartis en trois catégories : A, B et C (fig. 4.4). Les représentations aussi étaient réparties en trois catégories : 1, 2 et 3 (fig. 4.4).

---

<sup>29</sup>Loi de suivi de trajectoire en français

<sup>30</sup>Pour le DigiHaptic, le chemin "A" nécessite l'utilisation de l'index et de l'annulaire, le chemin "B" l'utilisation du pouce, de l'index et de l'annulaire et le chemin "C" l'utilisation du pouce et de l'annulaire.

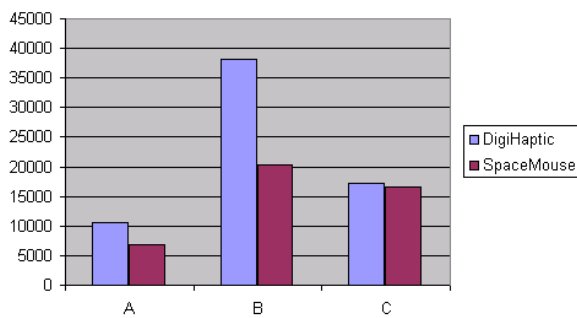


FIG. 4.7 – Temps moyen de parcours (ms) pour chaque chemin et périphérique

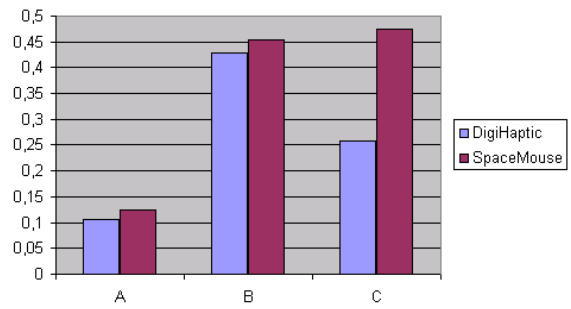


FIG. 4.8 – Coefficient de coordination pour la SpaceMouse et le DigiHaptic suivant le chemin

L'analyse de variance pour le temps moyen de parcours a révélé un effet significatif des périphériques ( $F(1,21)=5,734$ ,  $p < 0.0260$ ). Globalement, les tests montrent que les sujets accomplissent la tâche plus rapidement avec la SpaceMouse que le DigiHaptic. Le chemin a un effet significatif sur la performance ( $F(2,42)=70.291$ ,  $p < 0.0000$ ). Les résultats sur les chemins montrent que les chemins "B" et "C" demandent plus de temps que le chemin "A", que ce soit pour le DigiHaptic ou la SpaceMouse mais pour tous les chemins, ils sont meilleurs pour la SpaceMouse. Des effets significatifs ont également été trouvés pour l'interaction croisée périphérique \* chemin ( $F(2,42)=13.664$ ,  $p < 0.0000$ ) et chemin \* représentation ( $F(4,84)=4.247$ ,  $p < 0.0035$ ).

## Coordination

Pour mesurer le coefficient de coordination, nous avons utilisé la définition présentée en 4.4.1. Chaque fois, nous avons pris la longueur parcourue par l'utilisateur du début à la fin du chemin en incluant la longueur parcourue en dehors du chemin lors d'une sortie. Pour chaque chemin et représentation, le coefficient de coordination de la SpaceMouse est toujours plus grand que celui du DigiHaptic, ce qui signifie que la coordination des degrés de liberté est moins bonne sur la SpaceMouse. Il n'y a pas de grandes différences entre les coefficients de coordination calculés pour les représentations de chaque chemin.

Le chemin "A" est le plus coordonné des trois. Pour les chemins "A" et "B", les résultats sont similaires pour les deux périphériques alors que pour le chemin "C", il y a une grande différence entre les deux. Il est surprenant que le coefficient de coordination de la SpaceMouse pour la chemin "C" soit plus important que celui du chemin "B" alors que celui-ci est plus facile. Nous pouvons émettre l'hypothèse suivante : le chemin "C" est responsable du non respect de la loi de suivi de trajectoire pour la SpaceMouse. Les utilisateurs ont du mal à appuyer sur l'effecteur de la SpaceMouse tout en allant sur la droite.

Si nous comparons les chemins "A" et "C" qui demandent tous deux l'utilisation de deux degrés de liberté, le chemin "A" obtient de meilleurs résultats pour le DigiHaptic que pour la SpaceMouse. Nous pouvons supposer que cela est dû à la plus faible pente du chemin "A" : dès que l'utilisateur est engagé, il n'a besoin que de contrôler très peu la manette de l'annulaire. Pour le chemin "B", le coefficient de coordination est le plus faible parce que le chemin nécessite l'utilisation de trois degrés de liberté à la fois.

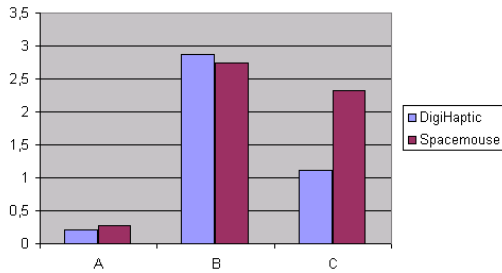


FIG. 4.9 – Nombre moyen de sorties pour chaque chemin et périphérique

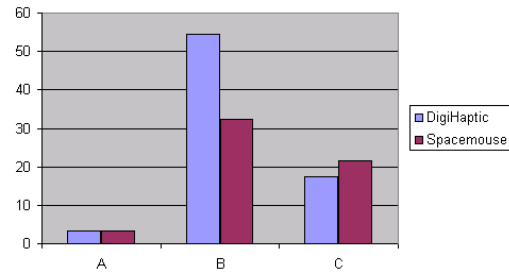


FIG. 4.10 – Nombre moyen de contacts pour chaque chemin et périphérique

## Sorties

La figure 4.9 montre le nombre moyen de sorties pour chaque chemin (A, B et C) avec chaque périphérique. L'analyse de variance pour le nombre moyen de sorties indique qu'il y a un effet principal du chemin ( $F(2,42)=76.811$ ,  $p<0.0000$ ). Pour les chemins "A" et "B", le nombre moyen de sorties est similaire entre les deux périphériques. Globalement, les sujets sortent plus avec le chemin "B" puisqu'il est plus difficile de coordonner trois degrés de liberté à la fois. Une fois de plus, on observe une différence importante entre le DigiHaptic et la SpaceMouse pour le chemin "C". En effet, il y a un effet significatif périphérique \* chemins ( $F(2,42)=6.090$ ,  $p<0.0047$ ). Ces résultats sont cohérents avec les résultats précédents de mesure de coordination.

## Contacts

La figure 4.10 représente le nombre moyen de contacts pour chaque chemin et périphérique. L'ANOVA sur le nombre moyen de contacts montre qu'il y a un effet significatif du chemin ( $F(1,42)=48.444$ ,  $p<0.0000$ ). Des effets significatifs d'interaction entre les périphériques et les chemins ont été trouvés ( $F(2,42)=6.017$ ,  $p<0.0050$ ). Finalement, il n'y a pas d'interaction entre les périphériques, les chemins et les représentations.

Pour le chemin "B", le nombre moyen de sorties pour le DigiHaptic et la SpaceMouse sont similaires alors que le nombre moyen de contacts est plus important pour le DigiHaptic. On aurait pu s'attendre à ce que le nombre de sorties soit proportionnel au nombre de contacts. Cela implique que le DigiHaptic permet un meilleur contrôle lorsque le bord du chemin est atteint et que le pointeur vire au noir.

Les nombres moyens de contacts pour le chemin "C" sont similaires pour le DigiHaptic et la SpaceMouse alors qu'on observe que le nombre moyen de sorties est plus important pour la SpaceMouse que le DigiHaptic. Ces résultats suggèrent un problème de contrôle en vitesse en plus du problème de coordination pour la SpaceMouse. Nous pouvons émettre comme hypothèse que le problème de contrôle en vitesse vient des faibles amplitudes de déplacement de l'effecteur du périphérique.

#### 4.4.4 Questionnaires

Nous avons demandé aux utilisateurs de juger la facilité d'utilisation, la fatigue et la performance globale de chacun des périphériques. Chaque critère était jugé entre 0 à 5. Nous avons regroupé 0 et 1 dans la catégorie que nous appelons *mauvais*, 2 et 3 de la catégorie *moyenne*, 4 et 5 de la catégorie *bonne*.

La facilité d'utilisation est considérée comme *moyenne* pour le DigiHaptic et la SpaceMouse (66% des réponses pour le DigiHaptic et 72% pour la SpaceMouse). Le DigiHaptic est considéré comme non-fatigant (58% des utilisateurs le place dans la catégorie *bonne*) alors que la SpaceMouse est considérée fatigante par 45% des utilisateurs (catégorie *mauvais*). Finalement pour la performance globale, 45% des utilisateurs place le DigiHaptic dans la catégorie *bonne* et 50% dans la catégorie *moyenne*. La SpaceMouse est considérée *moyenne* par 80% des utilisateurs.

Globalement le DigiHaptic est mieux jugé que la SpaceMouse. Nous n'avons pas pu demandé aux utilisateurs de comparer les périphériques puisqu'ils passaient sur l'un ou sur l'autre.

#### 4.4.5 Conclusion

Le premier résultat est que la conception du DigiHaptic permet la coordination simultanée des trois degrés de liberté. Par ailleurs, nous avons vu que la coordination de deux ou trois degrés suit toujours la steering law.

Nous avons vu que la séparation des degrés de liberté augmente le temps de parcours même si celle-ci permet un meilleur contrôle du pointeur 3D. Pour les chemins où le nombre de contacts est similaire entre le DigiHaptic et la SpaceMouse, le nombre de sorties est plus important avec cette dernière. Par ailleurs, pour les chemins où le nombre de contacts est plus important avec le DigiHaptic, le nombre de sorties reste similaire entre les deux périphériques. Ainsi la difficulté de coordination des degrés de liberté est compensée par de meilleures corrections et ajustements de trajectoires.

Le coefficient de coordination est meilleur avec le DigiHaptic qu'avec la SpaceMouse. Nous pensons que cela est dû à la relation directe qui existe entre les mouvements des doigts et ceux du pointeur, c'est-à-dire aux isomorphismes de direction et d'orientation. Ces résultats peuvent aussi s'expliquer par le fait que le DigiHaptic est plus élastique que la SpaceMouse. Ainsi le DigiHaptic offre un retour proprioceptif plus riche qui permet un meilleur contrôle. Cette hypothèse est appuyée par les résultats de Poulton qui indiquent que pour des tâches de suivi de cible, les contrôles en vitesse sont meilleurs en ce qui concerne l'ajustement de la position [Pou74].

Le problème observé sur la SpaceMouse entre les axes "x" et "y" nécessite davantage d'analyses pour être finement compris. Nous avons toutefois noté que les utilisateurs laissent leur poignet immobile. Or, pour le mouvement en question, les sujets ont à accomplir un mouvement de translation avec l'effecteur qui n'est pas facilement réalisable à cause de l'affordance du périphérique qui entraîne un mouvement involontaire de rotation. Comparé à la SpaceMouse, on peut noter que, pour toutes les configurations de degrés de liberté testées, le coefficient de coordination est proportionnel à la difficulté de coordination.

## 4.5 Evaluation en navigation avec retour d'effort

Nous présentons ici l'évaluation et la comparaison du DigiHaptic (en mode élastique) à la SpaceMouse dans une tâche de navigation en environnement 3D. Nous utilisons les deux périphériques pour le contrôle en vitesse de trois degrés de liberté de la caméra. Nous cherchons à étudier l'effet de la séparation des degrés de liberté sur les performances de l'utilisateur (temps de parcours et nombre de contacts). Nous souhaitons également étudier l'influence du retour de force lors des collisions.

### 4.5.1 Méthode

#### Présentation de la tâche

La tâche consiste à parcourir le tunnel 3D décrit figure 4.11 en contrôlant trois degrés de liberté de la caméra. La caméra peut se déplacer suivant les trois dimensions du tunnel et les collisions sont détectables sur l'ensemble de la paroi du tunnel (sol, plafond, murs).

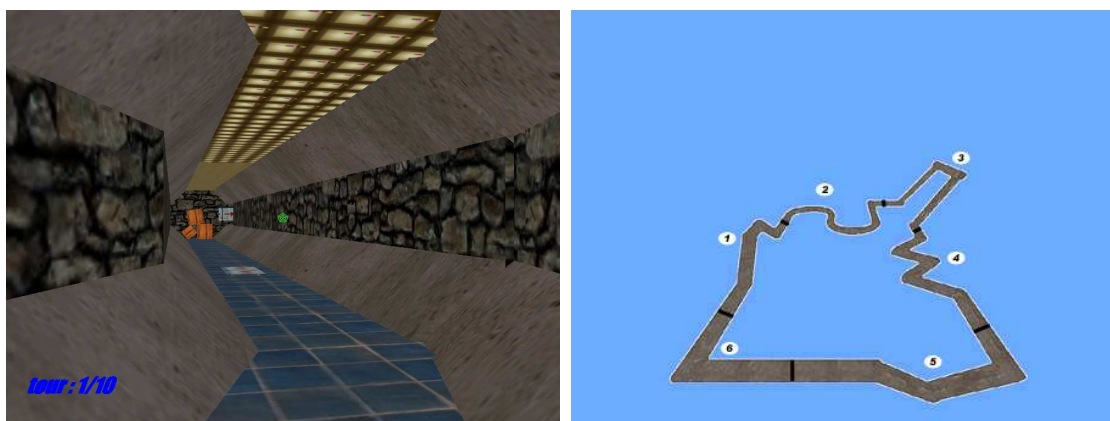


FIG. 4.11 – Capture écran de l'intérieur du tunnel et de l'ensemble du circuit.

Pour le DigiHaptic, le pouce contrôle le lacet et l'annulaire le tangage alors que l'index contrôle l'avancée de la caméra. Pour la SpaceMouse, les mêmes degrés de liberté sont contrôlés (fig. 4.12).

Le circuit est divisé en six parties de formes différentes. Nous pourrions ainsi analyser les performances des utilisateurs sur chacune des parties. L'utilisateur démarre au début du segment 1 et parcourt le circuit dans le sens des aiguilles d'une montre.

Le circuit a été dessiné avec un éditeur de cartes BSP puis chargé dans l'application de navigation décrite dans la partie précédente. Le dessin du circuit, l'application de textures et la gestion des collisions sont ainsi beaucoup plus faciles à gérer.

Des textures différentes pour le sol, le plafond et les murs permettent à l'utilisateur de garder la notion de la verticale en différenciant le haut du bas. Lors de l'avancée dans le tunnel, des flèches au sol et sur les murs, indiquant la direction à suivre, sont toujours visibles afin d'éviter que l'utilisateur ne fasse demi-tour. Au niveau de chaque coude, des caisses en bois sont représentées pour que l'utilisateur voit bien le virage et l'anticipe. Un

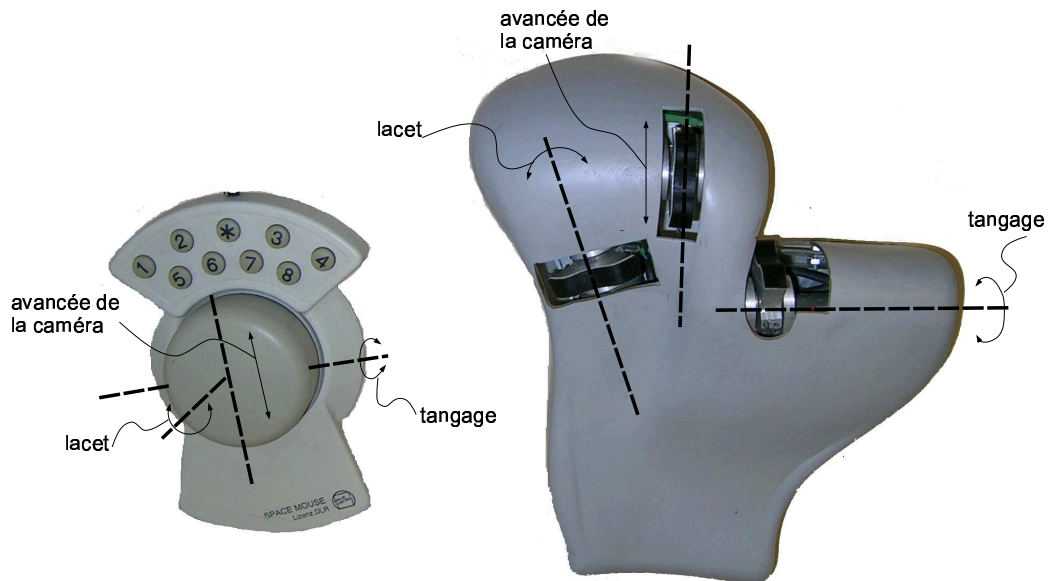


FIG. 4.12 – Degrés de liberté contrôlés par le DigiHaptic et la SpaceMouse.

point vert au centre de l'écran aide l'utilisateur à positionner la caméra. Enfin le nombre de tours réalisés est affiché en bas de l'écran en bleu.

Lors d'une collision avec la paroi du tunnel, la caméra est ralentie ou arrêtée suivant l'angle de collision avec le mur. Dans le cas de l'utilisation du DigiHaptic avec retour d'efforts, la force est uniquement envoyée sur l'index. L'index contrôle en effet l'avancée de la caméra, il est donc naturel de renvoyer la force de collision sur ce doigt pour stopper la caméra. Nous aurions pu renvoyer des forces sur le pouce et l'annulaire également comme présenté au paragraphe 3.4.2. Pour cette expérimentation, nous avons préféré commencer par évaluer le retour d'effort sur un doigt afin de s'assurer que les utilisateurs comprennent bien le retour d'effort et pour éviter que les résultats ne soient biaisés par un retour d'effort sur trois doigts mal compris. Le retour d'effort sur trois doigts aurait par ailleurs aidé l'utilisateur à replacer la caméra pour repartir.

## Sujets

Un total de 30 sujets a participé à l'étude. Tous les sujets, dont 26 étaient droitiers et 4 gauchers, ont utilisé leur main droite sur chacun des périphériques. Les sujets gauchers étaient habitués à utiliser la souris de la main droite et n'ont pas éprouvé de difficulté particulière à utiliser le DigiHaptic ou la SpaceMouse. Leur âge moyen est de 25 ans, le plus jeune avait 19 ans et le plus âgé 38 ans. La majorité des sujets joue régulièrement aux jeux vidéos 3D. Les participants ont été aléatoirement séparés en deux groupes de 15 personnes. Le premier groupe testait SpaceMouse(SM)/DigiHaptic sans retour de force(DH) (groupe 1) et le second groupe testait SpaceMouse(SM)/DigiHaptic avec retour de force(DHRE) (groupe 2). Afin d'éviter l'effet d'ordre, la moitié des sujets de chaque groupe commençait par le DigiHaptic et l'autre par la SpaceMouse. Aucun des sujets n'était familier avec la tâche en question ou les périphériques étudiés.

## Procédure

Les sujets étaient assis en face de l'écran à une distance d'environ 70 cm. Les tests commençaient par une phase d'entraînement durant laquelle l'utilisateur se familiarisait avec le périphérique et l'environnement. Les sujets pouvaient s'entraîner aussi longtemps qu'ils le souhaitaient jusqu'à ce qu'ils se sentent prêts. D'une manière générale chaque sujet réalisait environ cinq tours d'entraînement. Les sujets réalisaient ensuite les dix tours de circuit après avoir eu comme consigne d'aller le plus vite possible en touchant le moins possible les parois du tunnel.

Après avoir rempli un questionnaire, les sujets recommençaient les mêmes étapes avec le second périphérique.

Le temps de parcours, le nombre de contacts ainsi que la longueur totale parcourue étaient enregistrés toutes les 30 ms environ.

## Matériel

Le moniteur avait une taille de 21 pouces réglé à une résolution de 1024x768. Comme périphérique d'entrée, nous avons utilisé une SpaceMouse Classic et un DigiHaptic utilisé en mode élastique. Pour ce dernier des raideurs fixes ont été réglées sur les manettes pour tous les sujets. Enfin, pour assurer une comparaison équitable des deux périphériques, nous avons réglé les sensibilités de chaque périphérique avec deux sujets expérimentés sur chacun d'entre eux. Nous avons procédé à différents réglages et nous avons retenu celui qui demandait le moins de temps pour parcourir l'ensemble des circuits.

### 4.5.2 Résultats

Nous avons utilisé le test paramétrique  $t$  de Student avec une valeur significative  $p = 0,05$ .

Avant de traiter les données, nous avons classé les sujets suivant leur temps total de parcours du circuit et le nombre total de contacts (fig. 4.13). A partir du temps moyen et du nombre de contacts moyen, nous avons classé les sujets en quatre catégories : rapide et précis, rapide et non-précis, lent et précis, et lent et non-précis.

Afin de diminuer la variance des résultats, nous n'avons pas pris en compte les résultats des sujets qui se situent en dehors du cercle, que nous avons défini sur la figure 4.13. Cela correspond à deux sujets dans chaque groupe.

### Temps de parcours

Nous pouvons constater que les temps moyens de parcours du circuit sont très proches pour le DigiHaptic et la SpaceMouse (fig. 4.14) et qu'ils ont tendance à décroître au fil des tours.

Les temps moyens par tour pour chaque périphérique du premier groupe sont quasiment identiques et la différence n'est pas significative. Elle n'est pas non plus significative pour le second groupe même si le temps moyen de parcours est plus faible pour la SpaceMouse (tab. 4.5).



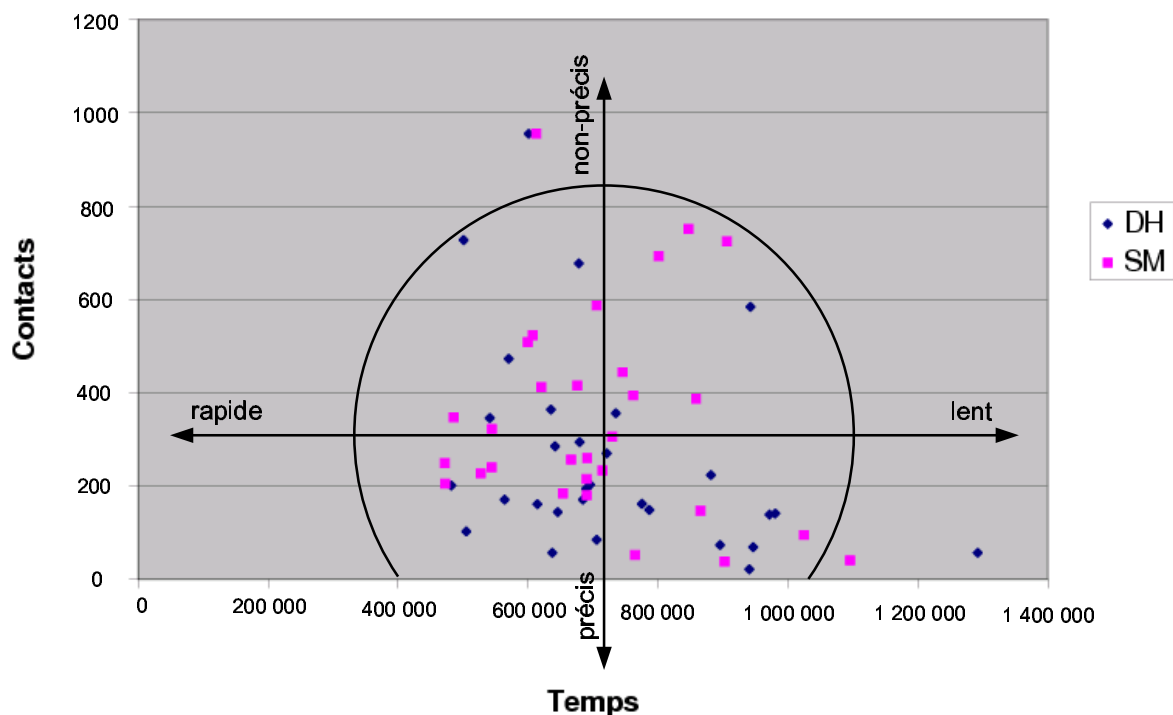


FIG. 4.13 – Position des sujets en fonction de leur temps total de parcours et du nombre total de contacts.

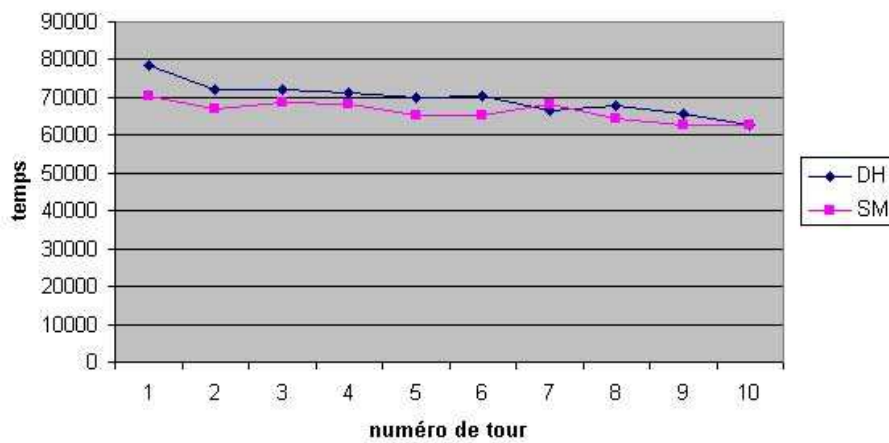


FIG. 4.14 – Temps moyen de parcours du circuit (ms) pour l'ensemble des sujets, par tour de circuit, pour le DigiHaptic et la SpaceMouse.

TAB. 4.5 – Temps moyen par tour

groupe 1	SM	70 705 ms	$p(\text{SM},\text{DH})=0,83$
	DH	69 995 ms	
groupe 2	SM	66 982 ms	$p(\text{SM},\text{DHRE})=0,12$
	DHRE	73 745 ms	

### Nombre de contacts

En ce qui concerne le nombre moyen de contacts par tour, nous remarquons qu'il est toujours plus important pour la SpaceMouse (fig. 4.15).

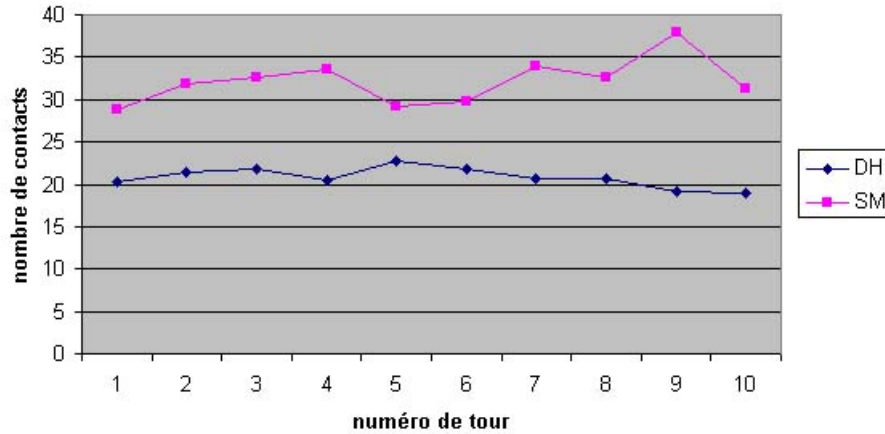


FIG. 4.15 – Nombre moyen de contacts pour l'ensemble des sujets, par tour de circuit, pour le DigiHaptic et la SpaceMouse.

Cette différence est confirmée pour chacun des groupes où la différence du nombre moyen de contacts est significative (tab. 4.6).

TAB. 4.6 – Nombre de contacts moyen par tour

groupe 1	SM	36	p(SM,DH)=0,02
	DH	24	
groupe 2	SM	31	p(SM,DHRE)=0,04
	DHRE	22	

### Temps de contact

Pour les sujets du premier groupe, le temps de contact est plus faible avec la Space-Mouse mais la différence n'est pas significative. En revanche la différence est significative pour ceux du second groupe en faveur du DigiHaptic. Nous nous attendions à ce résultat puisque le retour de force s'oppose à la pénétration dans le mur.

TAB. 4.7 – Temps moyen de contact

groupe 1	SM	186 ms	p(SM,DH)=0,12
	DH	201 ms	
groupe 2	SM	201 ms	p(SM,DHRE)=0,03
	DHRE	176 ms	

### Distance parcourue

La distance moyenne parcourue dans chaque groupe est inférieure pour le DigiHaptic (tab. 4.8). Même si la différence est significative pour le premier groupe, ce n'est pas le cas pour le second. Il est donc difficile de conclure.

TAB. 4.8 – Distance moyenne parcourue.

groupe 1	SM	7 596	p(SM,DH)=0,05
	DH	7 344	
groupe 2	SM	7 270	p(SM,DHRE)=0,47
	DHRE	7 171	

### Résultats par zones

Nous avons traité les résultats par zone afin d'analyser l'influence de la forme du parcours. Concernant les temps de parcours par zones (tab. 4.9), les résultats sont équivalents d'un périphérique à l'autre et d'un groupe à l'autre. Cependant les résultats ne sont pas significatifs et ne permettent pas de conclure.

TAB. 4.9 – Temps de parcours total (ms) par zone, par groupe et par périphérique.

zone	groupe1			groupe2		
	DH	SM	p	DHRE	SM	p
1	96 855	100 377	0,60	100 508	106 377	0,30
2	162 123	160 495	0,84	166 738	167 109	0,96
3	161 217	158 454	0,79	176 531	154 067	0,06
4	128 607	125 544	0,59	146 392	132 207	0,08
5	97 641	103 036	0,41	108 966	101 221	0,24
6	36 065	38 475	0,33	40 957	41 390	0,84

En revanche, les résultats sont plus significatifs concernant le nombre de contacts par zone. Nous pouvons constater que dans tous les cas de figure, le nombre de contacts pour le DigiHaptic est moins important que pour la SpaceMouse. Par ailleurs, les résultats sont significatifs dans les deux groupes pour les zones 2 et 4 : la zone avec les deux arcs de cercles et celle avec les chicanes rapprochées, particulièrement difficiles.

### Nombre de degrés de liberté utilisés

Nous avons souhaité déterminer le nombre de degrés de liberté utilisés simultanément pour le DigiHaptic et la SpaceMouse. Pour cela, nous nous sommes inspirés de l'article de Jacob et al. dans lequel une analyse similaire est réalisée pour des périphériques isotoniques [JSMM94].

A partir des données recueillies toutes les 30 ms sur l'ensemble des sujets des deux groupes, nous avons regardé pour chaque intervalle de temps le nombre de degrés de

TAB. 4.10 – Nombre de contacts totaux par zone, par groupe et par périphérique.

zone	groupe1			groupe2		
	DH	SM	p	DHRE	SM	p
1	41	53	0,08	32	54	0,04
2	34	70	0,00	33	69	0,04
3	60	79	0,25	56	87	0,09
4	35	69	0,01	46	83	0,02
5	46	64	0,08	42	66	0,02
6	9	15	0,12	9	17	0,08

liberté modifiés. Nous avons considéré qu'il y a changement suivant un degré de liberté si la variation d'angle est supérieure à  $0,15^\circ$  pour le DigiHaptic et 3 unités pour la SpaceMouse. Pour cette dernière, la position de chaque degré de liberté est représentée entre -300 et 300. Ces seuils sont compris dans des intervalles ( $0,07^\circ - 0,3^\circ$  pour le DigiHaptic et 1 unité - 6 unités pour la SpaceMouse) où les résultats varient peu.

Nous pouvons constater que pour la plupart du temps, aucun degré de liberté n'est modifié pour la SpaceMouse. Les utilisateurs ont ensuite tendance à modifier deux ou trois degrés de liberté plutôt qu'un seul. Nous pensons que cela est dû au faible débattement de l'effecteur de la SpaceMouse qui rend difficile l'exécution d'un mouvement suivant un seul degré de liberté à la fois (fig. 4.16).

En ce qui concerne le DigiHaptic, les utilisateurs essaient d'utiliser le moins de degrés de liberté. Ainsi ils préfèrent ajuster un seul degré de liberté à la fois. Les différences observées entre les deux périphériques pour chaque cas sont toutes significatives ( $p < 0,000$ ) à l'exception de deux degrés de liberté utilisés simultanément ( $p = 0,16$ ).

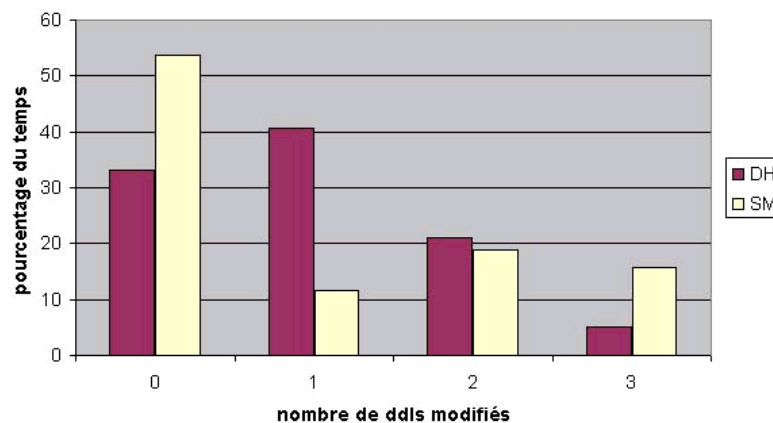


FIG. 4.16 – Pourcentage de temps suivant le nombre de degrés de liberté modifiés.

## Questionnaires

Nous avons ensuite posé une série de questions aux participants (tab. 4.11 et tab. 4.12). En ce qui concerne la difficulté globale du circuit, elle a été jugée moyenne pour chacun des périphériques. Les parties 2, 3 et 4 sont considérées comme les plus difficiles. Par ailleurs, le circuit est jugé plutôt plaisant.

La vitesse de déplacement dans le circuit est jugée assez rapide pour chacun des périphériques. Les réglages de sensibilité ont en effet été réglés par deux utilisateurs expérimentés. Il est important qu'un périphérique ne soit pas jugé plus sensible qu'un autre, sinon les résultats seraient biaisés.

L'utilisation du DigiHaptic est globalement jugée plus facile que celle de la SpaceMouse. La difficulté d'apprentissage est considérée comme moyenne et la durée d'apprentissage, plutôt rapide. Plus précisément, les sujets ont considéré que le DigiHaptic est rapide à prendre en main, que l'utilisation de deux degrés de liberté à la fois se fait aisément mais que l'utilisation de trois degrés de liberté à la fois demande un effort cognitif beaucoup plus important. Les sujets doivent en effet réfléchir au doigt qui contrôle le degré de liberté correspondant de la caméra. L'utilisation du pouce pour contrôler le lacet ne pose de problème pour personne. En revanche, certains utilisateurs confondent les degrés de liberté contrôlés par l'index et l'annulaire lorsqu'ils ne sont pas utilisés. La confusion disparaît lorsqu'un de ces deux degrés de liberté est utilisé. Par ailleurs, les participants estiment que cette confusion peut disparaître rapidement avec l'entraînement. Certains sujets confondent aussi le sens de déplacement de l'annulaire<sup>31</sup>. Le débattement de l'effecteur de la SpaceMouse est également jugé trop faible par rapport aux débattements des manettes du DigiHaptic, ce qui a pour effet un contrôle moins efficace de la vitesse de déplacement.

Concernant la fatigue d'utilisation, les participants ont jugé le DigiHaptic comme moyennement ou peu fatigant. Les critiques ont principalement portées sur la manette de l'annulaire dont la position n'est pas jugée ergonomique. Quelques douleurs peuvent apparaître au niveau de l'avant-bras ou du poignet. Pour la SpaceMouse, les sujets se divisent en deux catégories : ceux qui la trouve douloureuse à cause des tensions musculaires dans les doigts pour maintenir l'effecteur stable et ceux qui trouvent le périphérique non-fatigant. Il semble que les sujets de la première catégorie adoptent une mauvaise technique malgré les conseils d'usage prodigués. Les sujets se plaignent alors de douleurs dans la main, le poignet et l'avant-bras.

La dé-coordination<sup>32</sup> pour la SpaceMouse est jugée moyennement difficile. Le principal problème est d'avancer dans le tunnel sans faire pivoter en même temps la caméra puisque les deux mouvements sont liés. La coordination<sup>33</sup> est jugée relativement facile pour le DigiHaptic en dépit des problèmes cités plus haut.

Enfin, le retour d'effort est considéré comme une aide dans la navigation pour savoir si la caméra a touché les parois du tunnel.

Finalement, 70 % des sujets ont préféré le DigiHaptic. Cette préférence peut être expliquée par l'attrait suscité par un nouveau périphérique, mais aussi par le fait que les

---

<sup>31</sup>Cela semble particulièrement vrai chez les sujets qui inversent la souris dans les jeux 3D

<sup>32</sup>Être capable d'utiliser un degré de liberté indépendamment des autres

<sup>33</sup>Être capable d'utiliser tous les degrés de liberté à la fois

TAB. 4.11 – Résultats des questionnaires pour la SpaceMouse en pourcentages.

		SM						
circuit	difficile	0	6,5	<b>48</b>	<b>26</b>	13	6	facile
	ennuyeux	0	0	<b>36</b>	<b>25</b>	29	11	plaisant
vitesse de déplacement	lente	0	6,5	13	<b>39</b>	<b>32</b>	10	rapide
	utilisation difficile	3	<b>35</b>	<b>35</b>	6	19	0	facile
	apprentissage difficile	0	23	<b>35</b>	<b>23</b>	19	0	facile
	lent	0	13	17	<b>20</b>	<b>47</b>	3	rapide
	périphérique fatigant	<b>23</b>	<b>19</b>	13	3	<b>23</b>	<b>19</b>	pas fatigant
	dé-coordination difficile	7	21	<b>41</b>	<b>24</b>	7	0	facile

TAB. 4.12 – Résultats des questionnaires pour le DigiHaptic en pourcentages.

		DH+DHRE						
circuit	difficile	0	6,5	<b>39</b>	<b>39</b>	13	3	facile
	ennuyeux	0	10	10	<b>40</b>	<b>37</b>	3	plaisant
vitesse de déplacement	lente	0	0	13	<b>39</b>	<b>45</b>	3	rapide
	utilisation difficile	6	16	<b>29</b>	<b>16</b>	<b>32</b>	0	facile
	apprentissage difficile	7	10	<b>30</b>	<b>13</b>	<b>37</b>	3	facile
	lent	3	20	6,7	<b>20</b>	<b>43</b>	7	rapide
	périphérique fatigant	3	9,7	<b>29</b>	<b>13</b>	<b>32</b>	13	pas fatigant
	coordination difficile	16	9,7	<b>23</b>	<b>19</b>	<b>32</b>	0	facile

utilisateurs réalisent plus facilement ce qu'ils veulent avec le DigiHaptic.

### 4.5.3 Discussion et conclusion

Pour conclure, nous avons remarqué que le temps de parcours du circuit pour les deux périphériques étaient équivalents et que les différences de temps de parcours n'étaient pas significatives. Le réglage de la sensibilité des périphériques n'est pas en cause puisqu'il est jugé similaire pour chaque périphérique. Nous ne pouvons donc pas conclure à la supériorité d'un périphérique du point de vue de la rapidité. Les résultats des questionnaires indiquent toutefois que l'on peut espérer aller plus vite avec le DigiHaptic après une plus longue période d'apprentissage.

Du point de vue précision, la différence du nombre de contacts est significative, en faveur du DigiHaptic, pour chaque groupe. Le DigiHaptic permet donc un meilleur contrôle de la caméra, en particulier dans les trajectoires qui ne demandent l'utilisation que de deux degrés de liberté à la fois et qui sont jugées particulièrement difficiles (zones 2 et 4). Dans ces zones, qui ne requièrent que l'utilisation du pouce et de l'index, l'ajustement de la trajectoire doit être permanent. Pour les autres zones, qui nécessitent l'utilisation de l'annulaire, les résultats sont moins significatifs car les utilisateurs ont tendance à confondre l'utilisation de l'index et de l'annulaire et la position de la manette de l'annulaire n'est pas assez ergonomique. Ainsi, l'isomorphisme entre les degrés de liberté du périphérique et ceux de la caméra pour l'index et l'annulaire ne sont pas évidents pour tous les utilisateurs. Les moins bons résultats obtenus pour la SpaceMouse peuvent en partie s'expliquer par le fait qu'une partie des utilisateurs trouvent le périphérique particulièrement fatigant. Nous pensons également que la possibilité de contrôler un degré de liberté à la fois pour le DigiHaptic permet un contrôle plus précis de la caméra.

Le retour d'effort permet de réduire le temps de contact comme nous pouvions nous y attendre. Même s'il n'est pas considéré comme indispensable, c'est un plus pour l'utilisateur.

Du point de vue de la fatigue musculaire, la SpaceMouse est plus fatigante que le DigiHaptic. Celle-ci demande en effet plus de contractions musculaires pour contrôler l'effecteur qui a peu de débattement comparé au DigiHaptic.

En conclusion, le résultat de cette étude nous permet de penser que le DigiHaptic, préféré des utilisateurs, demande un peu plus de temps pour la prise en main mais permet d'obtenir des résultats meilleurs que ceux de la SpaceMouse. La séparation des degrés de liberté sur les périphériques dans les tâches de navigation semble donc tout à fait pertinente.

En perspective, nous pensons évaluer le retour d'effort sur les trois degrés de liberté pour voir les différences avec le retour d'effort sur un seul degré de liberté. Par ailleurs nous imaginons utiliser le retour d'effort pour guider l'utilisateur dans le tunnel : plus l'utilisateur s'approche d'une paroi, plus une force importante tend à l'écartier de celle-ci.

Nous envisageons également évaluer des critères plus qualitatifs tels que la récolte d'informations et la notion de présence dans l'environnement.

## 4.6 DigiHaptic version 2

Nous avons demandé aux utilisateurs lors des essais expérimentaux quels sont les défauts ergonomiques du DigiHaptic, en terme de mobilité des doigts et de fatigue musculaire.

La principale remarque concerne le manque d'appui de la paume de la main qui peut entraîner des mouvements involontaires entre les doigts et qui est fatigante lors d'utilisations prolongées. La seconde remarque porte sur la mobilité de l'annulaire : certains utilisateurs ont du mal à utiliser cette manette. Enfin la coque provoque une cassure au niveau du poignet qui peut entraîner un inconfort.

### 4.6.1 Présentation de la V2

Nous avons donc proposé une nouvelle ergonomie et un nouveau design (fig. 4.17 et 4.18). Pour cela, nous avons travaillé avec une maquette en clay qui nous a permis de modifier la forme à souhaits. La maquette a ensuite été scannérisée en 3D. Le modèle obtenu a été retravaillé et a servi à produire la coque en stéréo-lithographie. Une boule au niveau de la paume de la main permet de reposer la main et il n'y a plus de cassure au niveau du poignet. Les manettes ont été re-dessinées pour être plus pratiques à utiliser. Nous avons en particulier modifié la forme de la manette de l'annulaire.

Nous avons évalué cette version de manière informelle lors de la conférence EuroHaptics 2004. Les remarques des utilisateurs nous ont montré que l'ergonomie est nettement améliorée. Certains utilisateurs se sont cependant plaints de la position de la manette de l'annulaire. Des efforts restent donc à accomplir pour améliorer l'ergonomie de celle-ci.



FIG. 4.17 – La version 2 du DigiHaptic avec et sans la coque.

### 4.6.2 Autre design

Comme il est difficile de passer de la translation à la rotation d'objets avec la version actuelle, nous imaginons utiliser un DigiHaptic dans chaque main : l'un pour effectuer les translations et l'autre pour les rotations.

Nous imaginons également utiliser deux DigiHaptic orientés à 90°, un pour chaque main. L'orientation à 90° rendrait la position de la main encore moins fatigante, puisque





FIG. 4.18 – Placement de la main sur le DigiHaptic.

cette position correspond à la position naturelle de la main au repos lorsque les bras sont le long du corps. Le fait de travailler avec les mains en vis à vis donnerait également à l'utilisateur l'impression de manipuler directement l'objet virtuel dans ses mains. Le changement d'orientation nécessitera de repenser l'isomorphisme des manettes.

## 4.7 Conclusion

Dans une première partie de ce chapitre, nous nous sommes intéressés principalement aux outils quantitatifs d'évaluation des périphériques. Ces outils ont pour but de comparer les performances des périphériques afin de déterminer quels sont les périphériques les plus appropriés suivant le type de tâche (principalement pointage et suivi de trajectoire).

Nous avons vu que la loi de Fitts est un des seuls outils quantitatifs pour l'évaluation des performances des utilisateurs dans les tâches de pointage. Il permet ainsi de comparer les performances des périphériques par le calcul d'un indice de performance. Les dernières évolutions de la loi permettent de prendre en compte à la fois la rapidité et la précision des utilisateurs dans des tâches de pointage 2D. La loi de Fitts trouve aussi son utilité pour la mesure des bandes passantes des groupes musculaires. La loi de suivi de trajectoire, issue de la loi de Fitts, permet quant à elle de mesurer les performances des utilisateurs et des périphériques dans les tâches de suivi de trajectoire. Nous avons également vu que la mesure du taux de coordination des degrés de liberté renseigne sur la capacité des utilisateurs à aller facilement entre deux points en utilisant une ligne droite. Enfin, nous avons vu que le temps d'apprentissage suit une courbe caractéristique différente pour chaque périphérique. Cela constitue également un critère de comparaison des périphériques.

Nous avons ensuite présenté les deux expérimentations réalisées sur le DigiHaptic, comparé à la SpaceMouse. Dans la tâche de suivi de trajectoire, le DigiHaptic demande plus de temps pour réaliser la tâche que la SpaceMouse. La précision obtenue avec le DigiHaptic est cependant meilleure que celle obtenue avec la SpaceMouse. La séparation des degrés de liberté, ainsi que les déplacements plus importants des manettes du DigiHaptic comparés au déplacements de l'effecteur de la SpaceMouse, permettent d'expliquer ce résultat. Par ailleurs nous avons vu que les performances du DigiHaptic sont isotropiques comparées à

celles de la SpaceMouse. Ces travaux ont fait l'objet d'une publication [CPC04].

Pour la tâche de navigation avec retour d'effort, les temps de réalisation de la tâche sont équivalents entre les deux périphériques. Le DigiHaptic se montre cependant nettement plus précis que la SpaceMouse. Par ailleurs, le retour d'effort n'a pas permis de montrer une amélioration des performances des utilisateurs.

A travers ces deux expérimentations de suivi de trajectoire et de navigation, nous pouvons conclure que le DigiHaptic est globalement mieux adapté, en termes de performances, pour la navigation. Nous pouvons expliquer cette différence par la nature des tâches.

Le suivi de trajectoire est en effet une tâche nettement plus intégrale que la navigation : lors du suivi de trajectoires, l'utilisateur doit à tout moment coordonner les trois manettes pour ajuster sa trajectoire. Cette coordination est délicate puisque l'utilisateur manipule indépendamment chaque degré de liberté, ce qui occasionne une perte de temps importante. La séparation des degrés de liberté permet cependant d'ajuster beaucoup plus finement la trajectoire du pointeur, ce qui limite le nombre de contacts et de sorties du chemin.

La navigation est une tâche plus séparable. Nous avons vu que les utilisateurs limitent le nombre de degrés de liberté utilisés sur chacun des périphériques. La séparation des degrés de liberté du DigiHaptic permet alors un meilleur contrôle de la trajectoire et donc une meilleure précision. De plus, la coordination des degrés de liberté séparés ne constitue pas une perte de temps pour l'utilisateur puisque les temps de parcours sont similaires sur le DigiHaptic et la SpaceMouse.

Nous pouvons donc confirmer les résultats de Jacob et al. [JSMM94] qui ont montré que les périphériques séparables sont mieux adaptés aux tâches séparables. Nous montrons de plus que ce résultat reste valable si l'utilisateur a la possibilité de coordonner simultanément les degrés de liberté séparés, ce qui n'était pas réalisable dans l'expérimentation de Jacob.

Suite aux questionnaires des deux évaluations, nous avons utilisé les remarques des utilisateurs pour développer une nouvelle version du DigiHaptic, plus ergonomique.

# Conclusion

Notre travail visait à développer un nouveau périphérique haptique pour l'interaction en environnement 3D.

Nous sommes partis d'un *état de l'art* des périphériques d'entrée. Nous nous sommes pour cela concentrés sur la classification des périphériques existants, du point de vue de leurs propriétés physiques. Nous avons ensuite étudié les relations entre les propriétés physiques des périphériques et les grandeurs informatiques qu'ils contrôlent.

Puis nous avons étudié le *retour haptique* qui exploite le sens du toucher pour transformer les périphériques d'entrée en périphérique d'entrée-sortie. Nous avons pour cela étudié le sens du toucher, ce qui nous a permis de définir le retour d'effort et le retour tactile. Nous nous sommes alors concentrés sur notre sujet d'étude, le retour d'effort, pour analyser d'une part les moyens techniques qui permettent de le rendre et d'autre part les problèmes d'intégration que cela représente.

Notre volonté de concevoir un périphérique innovant et la constatation que la majeure partie des périphériques actuels sont à degrés de liberté intégrés nous ont conduits à l'idée originale de la *séparation des degrés de liberté*. Nous avons été confortés dans cette voie par la possibilité de conserver les *isomorphismes d'orientation et de direction* pour chaque degré de liberté et par la simplicité mécanique induite par la séparation des degrés de liberté. Nous avons montré comment mettre en œuvre cette idée de séparation des degrés de liberté à travers la résolution des problèmes de conception mécanique et électronique.

Les solutions adoptées font du DigiHaptic un périphérique robuste et fiable, facile à intégrer dans n'importe quelle application.

Lors de l'identification des paramètres du périphérique, nous avons constaté que le coefficient de frottement visqueux des manettes est trop faible pour simuler des murs virtuels de raideur importante. Les méthodes existantes, qui utilisent le frottement visqueux de l'interface pour calculer les paramètres du mur virtuel, n'ont par conséquent pas pu être appliquées. Nous avons alors développé une *méthode originale de détermination des paramètres optimaux d'un mur virtuel*. Cette méthode donne de très bons résultats pour le DigiHaptic.

Nous avons ensuite exploré les possibilités d'utilisation du DigiHaptic pour une tâche de manipulation. Nous avons vu que le DigiHaptic en mode isotonique ne permet que de manipuler les objets en translation dans un faible volume de travail. Pour augmenter l'espace de travail, le périphérique peut être utilisé en mode élastique. Il est alors possible

d'utiliser les degrés de liberté tour à tour pour les translations ou les rotations d'objets. Nous avons vu qu'il est cependant difficile de passer au cours d'une activité du mode isotonique au mode élastique.

Pour garder le contrôle habituel de la manipulation des objets en mode isotonique tout en dépassant les limites matérielles imposées par le périphérique, nous avons proposé différentes *métaphores originales d'interaction utilisant un contrôle hybride isotonique élastique*, valables pour tout périphérique à retour d'effort. Elles consistent à définir un volume de travail dans l'environnement virtuel sous forme de cube ou de sphère. L'utilisateur travaille en mode isotonique à l'intérieur du volume et peut déplacer celui-ci en mode élastique lorsqu'il en atteint les limites.

Nous avons ensuite étudié l'utilisation du DigiHaptic pour la *navigation en environnements 3D*. Nous avons proposé une métaphore de contrôle de la caméra proche de celle du contrôle de véhicule. En créant la métaphore, nous avons une nouvelle fois cherché à respecter les isomorphismes d'orientation et de direction. Nous utilisons ici le DigiHaptic en mode élastique pour réaliser un contrôle en vitesse de la caméra. Nous avons mis en application la simulation de murs en mode élastique dans cette application de façon à pouvoir ressentir les forces de collision de la caméra avec l'environnement virtuel. Nous introduisons ainsi la *navigation en environnement 3D avec retour de force*.

Nous nous sommes également intéressés au *retour de force en mode élastique*. Les périphériques à retour de force existants fonctionnent en mode isotonique et le retour de force est fait dans ce mode. Nous avons, dans un premier temps, traité la question de la *simulation de murs en mode élastique*. Nous avons vu que les objets doivent être contrôlés en position lorsqu'ils entrent en collision. Nous avons également étudié comment déterminer les paramètres optimaux d'un mur élastique. Nous avons ensuite proposé deux méthodes pour rendre des forces en mode élastique correspondant à des *phénomènes produisant des forces de normes faibles à modérées correspondant à la simulation de collisions entre objets déformables*. La première consiste à modifier la raideur du ressort simulée en fonction de la force calculée dans l'environnement virtuel et la seconde consiste à ajouter un terme proportionnel à la force générée dans l'environnement virtuel. La seconde méthode donne de meilleurs résultats qualitatifs.

Nous avons enfin évalué le DigiHaptic dans une *tâche de suivi de trajectoire* et de *navigation en environnement 3D avec retour d'effort*. Dans les deux expérimentations, nous avons utilisé le DigiHaptic en mode élastique avec contrôle en vitesse et nous avons comparé le DigiHaptic à la SpaceMouse afin d'évaluer l'effet de la séparation des degrés de liberté.

Pour la tâche de suivi de trajectoire, nous avons montré que le DigiHaptic est moins rapide mais plus précis que la SpaceMouse. La tâche de navigation a, quant à elle, montré des performances en rapidité équivalentes entre les deux périphériques et une précision nettement améliorée pour le DigiHaptic. Le retour d'effort en navigation, même s'il est préféré des utilisateurs, n'améliore pas significativement leurs performances. Dans les deux tâches, le DigiHaptic est considéré moins fatigant que la SpaceMouse et il est généralement préféré.

Le DigiHaptic, périphérique à degrés de liberté séparés, semble donc mieux adapté aux tâches séparables même s'il permet d'améliorer la précision des tâches intégrales. Il reste

---

à évaluer ses performances dans des tâches complètement séparables.

Lors des deux expérimentations, nous avons demandé aux utilisateurs de critiquer l'ergonomie du périphérique. Leurs remarques nous ont permis de développer une deuxième version du périphérique plus ergonomique.

Dans l'optique de développer un prototype industriel reproductible, la principale difficulté est de remplacer la solution électronique existante par une solution moins encombrante, plus économique et facilement reproductible tout en conservant ou en améliorant les performances de la solution actuelle. Cela passe par la mise en œuvre de solutions autour d'un DSP ou d'un PIC avec une liaison USB 2.0 ou Firewire avec l'ordinateur gérant l'environnement virtuel.

Pour l'instant, nous n'avons pas trouvé d'application industrielle dans laquelle le DigiHaptic pourrait être utilisé. Le DigiHaptic reste néanmoins un outil de laboratoire intéressant qui nous a permis d'évaluer l'effet de la séparation des degrés de liberté. Nous envisageons de l'utiliser pour d'autres pistes de recherche intéressantes.

## Perspectives

A l'issue de ces travaux, les pistes de recherche sont nombreuses et concernent principalement l'évaluation du DigiHaptic.

Des expérimentations menées, il semble que le *temps d'apprentissage* pour le DigiHaptic est plus important que celui de la SpaceMouse. Même si la maîtrise s'acquiert rapidement lors de séances d'entraînement répétées, il serait intéressant d'analyser plus finement le temps d'apprentissage du DigiHaptic en fonction de la tâche. Les tâches séparables nécessitent-elles moins de temps d'apprentissage que les tâches intégrales, comme semblent l'indiquer les expérimentations menées? Après apprentissage, les performances du DigiHaptic dépassent-elles celles d'autres périphériques comme la SpaceMouse ou le PHANToM à degrés de liberté assemblés? Dans quelles tâches?

Nous n'avons par ailleurs pas répondu à la question du *sens de la projection du vecteur force sur chaque manette, en terme de perception pour l'utilisateur*. Même si nous avons remarqué que ces forces ont du sens pour l'utilisateur parce qu'il y a une correspondance entre l'information visuelle et l'information ressentie, grâce aux isomorphismes d'orientation et de direction, nous aimerions comprendre plus en détail quelle est la portée de la séparation des forces. Plus généralement, nous cherchons à répondre à la question suivante : « L'homme est-il capable de reconstruire intellectuellement un vecteur force à partir de ses trois composantes ressenties sur des doigts différents? »

Pour répondre plus précisément à notre question, nous proposons d'évaluer la discrimination l'orientation de plans suivant deux et trois degrés de liberté. Dans cette expérimentation, nous comparerions les résultats du DigiHaptic à ceux d'un périphérique haptique à degrés de liberté assemblés comme le PHANToM. Des expérimentations informelles nous ont montré que l'utilisateur est capable de reconnaître l'inclinaison d'un plan en aveugle.

Mais plus précisément, quel JND<sup>34</sup> d'angle un utilisateur est-il capable de discriminer en aveugle ?

Nous avons également ouvert un champ d'expérimentations important sur *le retour de forces en mode élastique*. Nous avons proposé deux méthodes pour rendre des forces en mode élastique et nous avons proposé une méthode pour simuler des murs dans ce même mode. Cependant, il reste beaucoup de questions ouvertes :

- Comment ces forces sont-elles perçues par l'utilisateur ?
- Dans quelles mesures l'utilisateur est-il capable de faire la différence entre la force de rappel du ressort et la force supplémentaire envoyée ?
- Quel est le degré de réalisme des forces simulées en comparaison avec le retour de force en mode isotonique ?
- Quels sont les apports du retour de force en mode élastique ?

Enfin, le DigiHaptic est par conception un périphérique avec un volume de travail limité. Nous avons proposé les *métaphores avec contrôle hybride isotonique/élastique* pour permettre son utilisation dans de grands environnements virtuels. Il nous reste à déterminer si l'utilisateur peut utiliser facilement ces métaphores et les points à améliorer. Les résultats obtenus seront, en outre, valables pour tous les périphériques isotoniques à retour d'effort où le volume de travail est limité.

---

<sup>34</sup>Just Noticeable Difference

# Annexe A

## Quaternions

### A.1 Définition

Un quaternion est une expression de la forme  $q = x \mathbf{i} + y \mathbf{j} + z \mathbf{k} + w$  où  $x, y, z, w$  sont des nombres réels et  $\mathbf{i}, \mathbf{j}, \mathbf{k}$  vérifient les relations suivantes :

$$\begin{aligned} \mathbf{i}^2 &= \mathbf{j}^2 = \mathbf{k}^2 = -1 \\ \mathbf{i} \cdot \mathbf{j} &= -\mathbf{j} \cdot \mathbf{i} = \mathbf{k} \\ \mathbf{j} \cdot \mathbf{k} &= -\mathbf{k} \cdot \mathbf{j} = \mathbf{i} \\ \mathbf{k} \cdot \mathbf{i} &= -\mathbf{i} \cdot \mathbf{k} = \mathbf{j} \end{aligned}$$

On appelle  $x \mathbf{i} + y \mathbf{j} + z \mathbf{k}$  la partie imaginaire du quaternion et  $w$  sa partie réelle. Par la suite, on notera le quaternion  $q$  sous la forme  $|x \ y \ z \ w|$ .

Les quaternions sont une extension des nombres complexes et permettent de caractériser une rotation d'axe et d'angle quelconques dans l'espace.

### A.2 Conversion d'une matrice homogène de rotation en quaternion

Une matrice homogène de rotation A.1 peut être convertie en quaternion  $|x \ y \ z \ w|$  en utilisant l'algorithme suivant :

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & 0 \\ a_{21} & a_{22} & a_{23} & 0 \\ a_{31} & a_{32} & a_{33} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{A.1}$$

- Calcul de la trace de la matrice  $t = a_{11} + a_{22} + a_{33} + 1$
- Si la trace est supérieure à zéro, alors calculer le quaternion :

$$s = 0.5/\sqrt{t}$$

$$\begin{aligned}
 x &= (a_{32} - a_{23}) * s \\
 y &= (a_{13} - a_{31}) * s \\
 z &= (a_{21} - a_{12}) * s \\
 w &= 0.25/s
 \end{aligned}$$

- Si la trace de la matrice est inférieure ou égale à zéro, identifier l'élément de la diagonale qui a la plus grande valeur, puis calculer en fonction de cette valeur :  
Colonne 1 :

$$\begin{aligned}
 s &= 2 * \sqrt{1 + a_{11} - a_{22} - a_{33}} \\
 x &= 0.5/s \\
 y &= (a_{21} + a_{12})/s \\
 z &= (a_{13} + a_{31})/s \\
 w &= (a_{32} + a_{23})/s
 \end{aligned}$$

Colonne 2 :

$$\begin{aligned}
 s &= 2 * \sqrt{1 + a_{22} - a_{11} - a_{33}} \\
 x &= (a_{21} + a_{12})/s \\
 y &= 0.5/s \\
 z &= (a_{32} + a_{23})/s \\
 w &= (a_{13} + a_{31})/s
 \end{aligned}$$

Colonne 3 :

$$\begin{aligned}
 s &= 2 * \sqrt{1 + a_{33} - a_{11} - a_{22}} \\
 x &= (a_{13} + a_{31})/s \\
 y &= (a_{32} + a_{23})/s \\
 z &= 0.5/s \\
 w &= (a_{21} + a_{12})/s
 \end{aligned}$$

### A.3 Conversion d'un quaternion en axe et angle de rotation

Un quaternion  $|x y z w|$  peut être converti en axe  $(u_x, u_y, u_z)$  et angle de rotation  $(\alpha)$  en utilisant l'algorithme suivant :

$$\begin{aligned}
 \cos(\alpha/2) &= w \\
 \alpha &= 2 * \arccos(w) \\
 \sin(\alpha/2) &= \sqrt{1 - w^2}
 \end{aligned}$$



Si  $|\sin(\alpha)| < \varepsilon$  avec  $\varepsilon = 0.0005$  alors  $sa = 1$  sinon  $sa = \sin(\alpha)$

$$\begin{aligned}u_x &= x/sa \\u_y &= y/sa \\u_z &= z/sa\end{aligned}$$

## A.4 Conversion d'un axe et angle de rotation en quaternion

Etant donné un axe de rotation unitaire  $(u_x, u_y, u_z)$  et un angle de rotation  $(\alpha)$ , il est possible de calculer le quaternion correspondant  $|x y z w|$  de la façon suivante :

$$\begin{aligned}x &= u_x * \sin(\alpha/2) \\y &= u_y * \sin(\alpha/2) \\z &= u_z * \sin(\alpha/2) \\w &= \cos(\alpha/2)\end{aligned}$$

Pour combiner plusieurs rotations, il suffit alors de multiplier les quaternions.

## A.5 Multiplication de quaternions

Soient deux quaternions  $q_1 = |x_1 y_1 z_1 w_1|$  et  $q_2 = |x_2 y_2 z_2 w_2|$ , on obtient  $q_3 = q_1 * q_2$  de la manière suivante :

$$\begin{aligned}x_3 &= w_1 * x_2 + x_1 * w_2 + y_1 * z_2 - z_1 * y_2 \\y_3 &= w_1 * y_2 - x_1 * z_2 + y_1 * w_2 + z_1 * x_2 \\z_3 &= w_1 * z_2 + x_1 * y_2 - y_1 * x_2 + z_1 * w_2 \\w_3 &= w_1 * w_2 - x_1 * x_2 - y_1 * y_2 - z_1 * z_2\end{aligned}$$

$q_3$  définit une rotation instantanée correspondant à la composition des rotations autour de l'axe défini par  $q_2$  puis celui défini par  $q_1$ .

Attention, la multiplication de quaternions tout comme la multiplication de matrices de rotations n'est pas commutative.



# Annexe B

## Les arbres BSP

Nous présentons d'abord dans cette annexe le principe des arbres BSP et leur création. Nous détaillons ensuite le rendu de la scène et le principe de calcul des collisions.

### B.1 Principe des arbres BSP

La représentation informatique d'une scène 3D est un problème classique aux solutions multiples. Alors qu'un publiciste recherche l'esthétisme des images de synthèse, qu'un médecin privilégie leur réalisme, nombreux sont ceux pour qui la fluidité des animations prime. Cet objectif est ambitieux car tous les processus de visualisation connus sont très chers en calcul.

Des recherches dans des domaines aussi variés que l'aéronautique (pour les simulateurs de vol) ou les jeux vidéos ont abouti en 1980 à une solution adaptée : l'algorithme des arbres BSP (Binary Space Partitioning) introduite par Fuchs, Kedem et Naylor [FKN80]. Cet algorithme est utilisé par ID Softwares dans le jeu Quake par exemple. Il prend en charge l'élimination des parties cachées qui permet de passer du tracé du modèle fil de fer de l'objet à une représentation plus habituelle où n'apparaissent que ses faces superficielles.

La compagnie ID Software est la première à avoir utilisé ce type d'algorithme, particulièrement bien adapté à la description de mondes statiques, pour un rendu des environnements tridimensionnels.

### B.2 Création des arbres BSP

Les arbres BSP sont un cas particulier des arbres binaires et ils représentent une région dans l'espace. L'arbre BSP (fig. B.1) est composé de noeuds (un noeud possède exactement deux enfants) et de feuilles (noeuds terminaux qui ne possèdent aucun enfant). Un noeud représente une partition de l'espace qui crée deux nouveaux espaces, un devant le noeud (visible) et un derrière (caché).

Dans les jeux, l'ensemble du monde est représenté par un arbre BSP. Dans le cas d'un monde 2D, le noeud principal définit la partition du monde en deux espaces selon une ligne, l'une des pièces est devant la ligne, l'autre pièce est derrière la ligne. Chacune des deux pièces est représentée par un sous arbre, lui aussi de type BSP. Chaque noeud

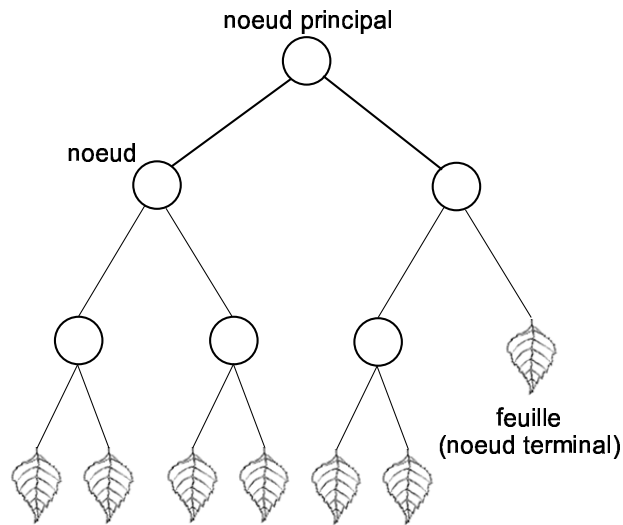


FIG. B.1 – Représentation simplifiée d'un arbre BSP.

contient un segment de ligne lui aussi utilisé dans la partition. Ce segment de ligne avec d'autres informations telles que la hauteur et l'identifiant de la texture deviendront un mur dans le monde. Les sous-arbres deviennent des feuilles si et seulement si l'espace qu'ils représentent ne contient pas d'autres murs à l'intérieur. Si ce n'est pas le cas, le mur sera utilisé pour partitionner de nouveau l'espace. Dans les applications 3D les choses sont à peu près semblables, l'espace est partitionné avec des plans.

Il existe deux principaux types d'arbres BSP. Dans la première méthode (appelée arbres BSP basés sur les nœuds ou *node-based BSP trees*), les nœuds contiennent l'ensemble des polygones et des plans utilisés pour la partition. Les feuilles sont vides. Dans l'autre méthode (appelée arbres BSP basés sur les feuilles ou *leafy BSP trees*), les nœuds contiennent uniquement les plans de partition. Les feuilles contiennent tous les polygones qui composent le monde complexe 2D ou 3D. Nous allons parler uniquement de la première méthode concernant les BSP basés sur les nœuds, mais les BSP basés sur les feuilles sont couramment utilisés. Les arbres BSP sont surtout employés quand les polygones utilisés pour la construction d'un arbre correspondent à une représentation limitée d'un objet. L'objet possède conceptuellement un intérieur solide, un extérieur entouré de vide et les polygones représentent la frontière entre ces deux milieux. Quand l'arbre est complet, chaque feuille représente soit un espace solide, soit un espace vide.

Nous allons illustrer la création d'un arbre BSP pour un environnement 2D. Le monde que nous allons décrire est composé d'un espace dans lequel est défini une région solide délimitée par quatre segments appelés respectivement A, B, C et D (fig. B.2). Chacun des segments possède une normale de visibilité qui est dirigée en dehors de la région solide.

Pour créer le noeud principal, nous allons utiliser le segment A (pris arbitrairement). Les segments B, C et D sont tous derrière la droite portant le segment A et sont donc cachés par rapport à celui-ci. Ainsi nous allons placer ces trois segments dans la liste des segments arrières. La liste des segments de devant est vide puisqu'il n'y a aucun segment devant le segment A. La liste vide est représentée par la feuille d'espace vide (signe moins).

Nous allons maintenant partitionner l'espace de derrière en utilisant le segment B

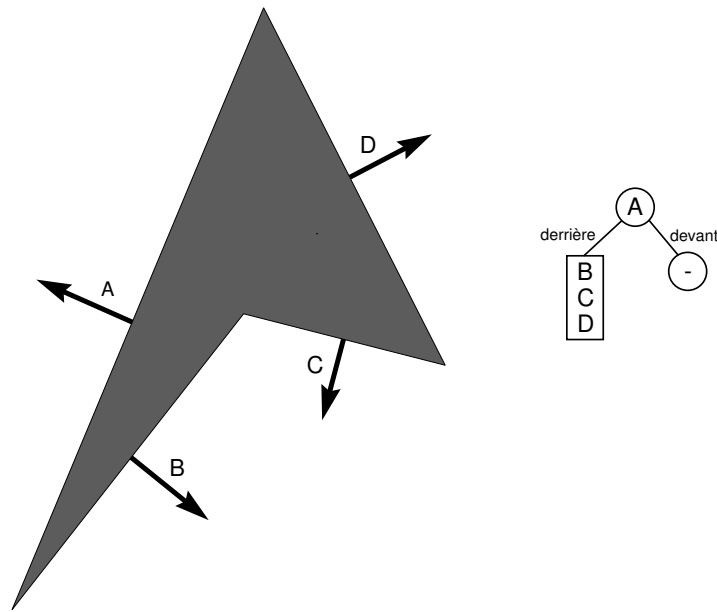


FIG. B.2 – Première étape de création de l'arbre BSP.

(choix arbitraire). Nous traçons alors la droite support du segment B qui intersecte le segment D. Le segment C est devant le segment B ainsi qu'une partie du segment D (appelée  $D_V$ ). L'autre partie est derrière (appelée  $D_C$ ). Partitionnons maintenant le devant du nœud (liste C et  $D_V$ ) en utilisant  $D_V$  comme segment de partition. C est l'unique segment à classifier et il se trouve derrière le segment  $D_V$ . La liste de devant est vide et de ce fait nous créons une feuille représentant un espace vide (voir figure B.3, schéma du haut pour illustration).

A ce stade, il ne reste plus que deux nœuds à traiter : C et  $D_C$ . Tous deux n'ont aucun segment à classifier et se voient attribuer deux feuilles : une pour la partie solide derrière le segment (signe +) et une pour l'espace vide devant le segment (signe -). On obtient alors l'arbre BSP final qui est représenté en bas de la figure B.3.

Le plus gros problème qui se pose pour un logiciel de partition de BSP est de choisir les segments pour partitionner l'espace du monde en sous-espaces en causant le moins d'éclatements de segments possible.

## B.3 Rendu de la scène

Il y a deux façons de rendre une scène, la première en affichant de l'arrière vers l'avant et la seconde de l'avant vers l'arrière. Dans la méthode arrière vers avant, les segments les plus éloignés sont d'abord affichés. Pour rester simple, le pseudo-code décrit figure B.4 affiche chaque segment dans sa totalité.

Dans notre exemple (fig. B.5), nous voulons savoir si le point de vue se trouve devant ou derrière le nœud racine (nœud A) afin d'afficher tout ce qui se trouve derrière le nœud avant d'afficher le nœud lui-même. Le point de vue est en fait derrière le segment A. Nous

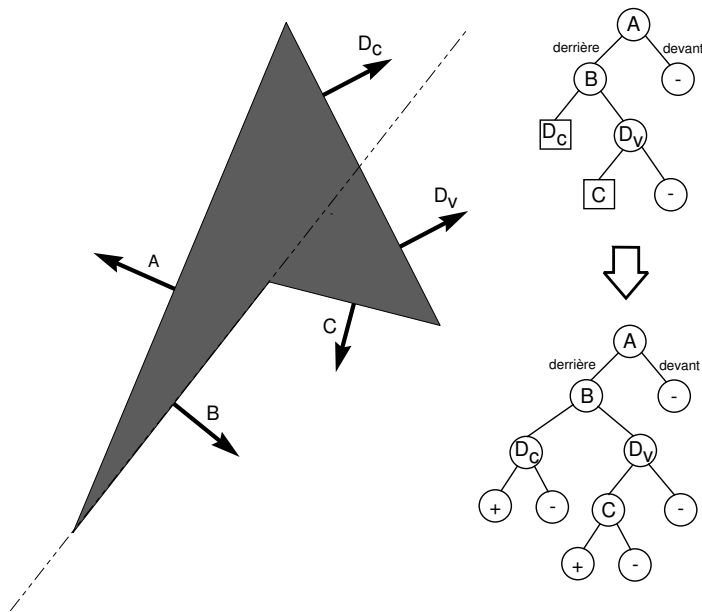


FIG. B.3 – Deuxième étape de création de l'arbre BSP.

```

procédure ArrièreVersAvant (noeud, point de vue) {
  Si (le noeud est une feuille) Alors {afficher noeud}
  Sinon
  {
    déterminer de quel côté se trouve la caméra
    Si (le point de vue est derrière le noeud) Alors
    {
      ArrièreVersAvant(noeud du sous arbre droit)
      ArrièreVersAvant(noeud du sous arbre gauche)
    }
    Sinon
    {
      ArrièreVersAvant(noeud du sous arbre gauche)
      ArrièreVersAvant(noeud du sous arbre droit)
    }
  }
}

```

FIG. B.4 – Algorithme de rendu arrière vers avant

savons maintenant que nous devons dessiner tous les segments devant  $A^{35}$ . Comme il n'y a rien à afficher devant A, nous affichons A. Nous cherchons maintenant à afficher B. B se trouve devant le point de vue, nous affichons alors  $D_C$  puis nous travaillons sur  $D_V$  qui se trouve derrière le point de vue. Nous affichons donc  $D_V$  puis C et enfin B.

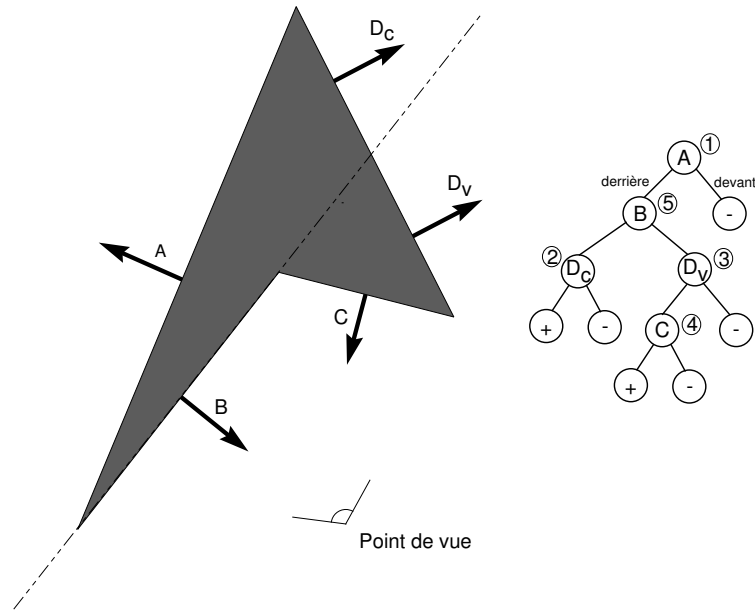


FIG. B.5 – Affichage de l'arbre BSP selon la position et l'orientation du point de vue.

Le nombre de polyèdres à représenter pour une carte de jeu comme Quake est de l'ordre de 10000. La création de l'arbre BSP est très longue mais l'arbre est pré-calculé une fois pour toute avant l'animation. Lors du rendu de la scène, le temps de parcours de l'arbre est négligeable devant le temps de tracé des faces.

## B.4 Gestion des collisions avec les arbres BSP

Pour détecter les collisions, l'idée est d'envoyer six rayons autour de la caméra (devant, derrière, en haut, en bas et sur les cotés) et de regarder si les distances entre la position de la caméra et les points d'intersection avec les polygones sont négatives. Dans ce cas, la caméra est replacée à une certaine distance de la surface du polygone en collision.

Le calcul du retour d'effort se fait en utilisant une méthode par pénalités. La distance de pénétration est calculée puis utilisée pour calculer la force correspondant à la collision entre la caméra et le mur modélisé par un ressort et un amortisseur.

L'algorithme de détection de collision est similaire à celui du rendu de la scène. Si nous prenons un des six segments, défini par son point de départ (PtDpt) et son point d'arrivée (PtArv), nous utilisons la procédure Collision (fig. B.6) afin de parcourir l'arbre et déterminer l'éventuelle distance de pénétration dans le mur.

<sup>35</sup>Pour savoir si le point de vue de l'utilisateur est devant ou derrière un segment, il suffit de faire un produit scalaire entre le vecteur visée du point de vue de l'utilisateur et la normale du segment considéré.

```
procédure Collision (noeud, PtDpt, PtArv) {
  Si (le noeud est une feuille) Alors {renvoyer distance entre caméra
  et plan de collision}
  Sinon
  {
    calculer distance entre PtDpt et plan de partitionnement du
    noeud (d1) et distance entre PtArv et plan de partitionnement
    du noeud (d2)
    Si (d1 > 0 et d2 > 0) Alors
    {
      //Il n'y a pas intersection et la caméra est devant le plan
      chercher Collision(noeud du sous arbre droit, PtDpt, PtArv)
    }
    Si (d1 < 0 et d2 < 0) Alors
    {
      //Il n'y a pas intersection et la caméra est derrière le plan
      chercher Collision(noeud du sous arbre gauche, PtDpt, PtArv)
    }
    Sinon
    {
      //Il y a intersection
      calcul point d'interrection (PtInter)
      chercher collision(noeud du sous arbre droit, PtDpt, PtInter)
      chercher collision(noeud du sous arbre gauche, PtInter, ptArv)
    }
  }
}
```

FIG. B.6 – Algorithme de gestion des collisions



# Index

- élastique, 8, 69
- admittance, 24
- affordances, 35
- bras maîtres, 20
- classification, 4
- continu, 4
- control display, 13
- détection de collision, 24
- degré de liberté, 4
- discret, 4
- Fitts, 88
- fonction de transfert, 9
- gants, 19
- god object, 25
- haptique, 14
- humunculus, 15
- impédance, 24
- interface intégrale, 10
- interface séparable, 10
- isométrique, 8
- isomorphisme, 12
- isotonique, 6
- JND, 16
- manipulation, 64
- mur virtuel, 26
- navigation, 64
- périphérique direct, 11
- périphérique indirect, 11
- périphériques liés, 19
- périphériques non-liés, 19
- passivité, 26
- proxy, 25
- quaternions, 119
- résolution, 4, 5
- retour de force, 25
- retour kinesthésique, 16
- retour tactile, 22
- stylos à retour de force, 20
- systèmes à câbles, 22
- taxonomie, 4
- temps réel, 27, 28
- transparence, 28



# Bibliographie

- [3dC] <http://www.3dconnexion.com> 3dConnexion.
- [AG90] L. Y. Arnaut and J. S. Greenstein. Is display/control gain a useful metric for optimizing an interface? *Human factors*, 32(6) :651–663, 1990.
- [AH99] Richard J. Adams and Blake Hannaford. Stable haptic interaction with virtual environments. *IEEE Transactions on Robotics and Automation*, 15(3) :465–474, 1999.
- [AZ97] Jonny Accot and Shumin Zhai. Beyond Fitts' Law : Models for Trajectory-Based HCI Tasks. In *CHI'97*, pages 295–302. ACM Press, 1997.
- [AZ99] Jonny Accot and Shumin Zhai. Performance Evaluation of Input Devices in Trajectory-based Tasks : An Application of The Steering Law. In *CHI'99*, pages 466–472. ACM Press, 1999.
- [AZ01] J. Accot and S. Zhai. Scale Effects in Steering Law Tasks. In *Proc of CHI'01*, pages 1–8, 2001.
- [BBPB02] M. Bouzit, G. Burdea, G. Popescu, and R. Boian. The Rutgers Masters II - New Design Force-Feedback Glove. *IEEE/ASME Transaction On Mechatronics*, vol. 7(no. 2) :pp. 256–263, 2002.
- [BC02] Joan De Boeck and Karin Coninx. Haptic Camera Manipulation : Extending the "Camera in Hand Metaphor". In *Eurohaptics*, 2002.
- [BH97] P. Berkelman and R. Hollis. Dynamic performance of a magnetic levitation haptic device. In *SPIE*, 1997.
- [BKH98] D. Bowman, D. Koller, and L. Hodges. A Methodology for the Evaluation of Travel Techniques for Immersive Virtual. *Virtual Reality : Research, Development and Applications*, 3(2) :120–131, 1998.
- [BM97] Ravin Balakrishnan and I. Scott MacKenzie. Performance Differences in the Fingers, Wrist, and Forearm in Computer Input Control. In *CHI'97*, pages 303–310. ACM Press, 1997.
- [BMH98] David C. Brogan, Ronald A. Metoyer, and Jessica K. Hodgins. Dynamically simulated characters in virtual environments. *IEEE Comput. Graph. Appl.*, 18(5) :58–69, 1998.
- [BS90] Teresa W. Bleser and John Sibert. Toto : a tool for selecting interaction techniques. In *Proceedings of the 3rd annual ACM SIGGRAPH symposium on User interface software and technology*, pages 135–142. ACM Press, 1990.

- [BSRO95] R. C. Barrett, E. J. Selker, J. D. Rutledge, and R. S. Olyha. Negative inertia : a dynamic pointing function. In *Conference companion on Human factors in computing systems*, pages 316–317. ACM Press, 1995.
- [BSSR02] A. Bicchi, E.P. Scilingo, N. Sgambelluri, and D. De Rossi. Haptic Interfaces Based on Magnetorheological Fluids. In *Eurohaptics*, 2002.
- [Bur96] G. Burdea. *Force and Touch Feedback for Virtual Reality*. New York : John Wiley & Sons, 1996.
- [Bux83] William Buxton. Lexical and pragmatic considerations of input structures. *Comput. Graph.*, 17(1) :31–37, 1983.
- [Car] <http://www.carlstahl.fr> CarlStahl.
- [Cat] <http://www.catia.com> Catia.
- [CC] <http://www.immersion.com> Cybergrasp CyberForce.
- [CC03] G. Casiez and C. Chaillou. Dispositif et procédé de commande d’un retour de force à appliquer à au moins une manette d’une interface motorisée. French Patent application number 03 06181, Filing date : May 22, 2003.
- [CCSP03] G. Casiez, C. Chaillou, B. Semail, and P. Plénacoste. Interface haptique de type ground-based et comportant au moins deux effecteurs digitaux rotatifs découplés. European Patent application number 03 370002.2, Filing date : January 07, 2003.
- [CEB78] Stuart K. Card, William K. English, and Betty J. Burr. Evaluation of Mouse, Rate-Controlled Isometric Joystick, Step Keys, and Text Keys for Text Selection on a CRT. *Ergonomics*, 21(8) :601–613, 1978.
- [CGSS93] J.E. Colgate, P.E. Grafing, M.C. Stanley, and G. Schenkel. Implementation of stiff virtual walls in force-reflecting interfaces. In *IEEE Virtual Reality Annual Int. Symposium*, pages 202–208, 1993.
- [Cla] <http://www.claw.com.au> Claw.
- [CMR90] Stuart K. Card, Jock D. Mackinlay, and George G. Robertson. The design space of input devices. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 117–124. ACM Press, 1990.
- [CMR91] Stuart K. Card, Jock D. Mackinlay, and George G. Robertson. A morphological analysis of the design space of input devices. *ACM Trans. Inf. Syst.*, 9(2) :99–122, 1991.
- [Cos] <http://www.ca.com/cosmo/> Cosmoplayer.
- [CP03] G. Casiez and P. Plénacoste. Le DigiHaptic, un nouveau périphérique haptique à trois degrés de liberté découplés pour la manipulation d’objets virtuels. In *Réalité Virtuelle et Sciences du Comportement, Ecole Thématique Interdisciplinaire du CNRS*, 19-23 mai 2003.
- [CPC04] G. Casiez, P. Plénacoste, and C. Chaillou. Does DOF Separation on Elastic Devices Improve User 3D Steering Task Performance? In LNCS 3101 Springer-Verlag Berlin Heidelberg, editor, *Asia-Pacific Conference on Computer-Human Interaction (APCHI’04)*, pages 70 – 80, June 29 - July 02 2004.

- 
- [CPCS03a] G. Casiez, P. Plénacoste, C. Chaillou, and B. Semail. Elastic Force Feedback with a New Multi-finger haptic device : The DigiHaptic. In *Proceedings of Eurohaptics*, pages 121–134, July 2003.
- [CPCS03b] G. Casiez, P. Plénacoste, C. Chaillou, and B. Semail. The DigiHaptic, a New Three Degrees of Freedom Multi-finger Haptic Device. In *Proceedings of Virtual Reality International Conference*, pages 35–39, May 2003.
- [CSB95] J.E. Colgate, M.C. Stanley, and J.M. Brown. Issues in the haptic display of tool use. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 140–145, 1995.
- [DDM] <http://www.discreet.com> Discreet 3DS Max.
- [DKM99] S. A. Douglas, A. E. Kirkpatrick, and I. S. MacKenzie. Testing pointing device performance and user assessment with iso 9241, part 9 standard. In *CHI'99*, pages 215–222. ACM Press, 1999.
- [dLLFR02] R.Q. Van der Linde, P. Lammertse, E. Frederiksen, and B. Ruiters. The HapticMaster, a new high-performance haptic interface. In *Proceedings of Eurohaptics*, pages 1–5, July 2002.
- [DMC02] J. Davanne, P. Meseure, and C. Chaillou. Stable haptic interaction in a dynamic virtual environment. In *IROS*, 2002.
- [DS93] Rudy P. Darken and John L. Sibert. A toolset for navigation in virtual environments. In *Proceedings of the 6th annual ACM symposium on User interface software and technology*, pages 157–165. ACM Press, 1993.
- [DS96] Rudolph P. Darken and John L. Sibert. Wayfinding strategies and behaviors in large virtual worlds. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 142–149. ACM Press, 1996.
- [eT] <http://www.etch3d.org> e Touch.
- [FCvdL04] E. C. Fritz, G. A.V. Christiansson, and R. Q. van der Linde. Haptic Gripper with Adjustable Inherent Passive Properties. In *Eurohaptics*, pages 324 – 329, 2004.
- [FD] <http://www.forcedimension.com> Force Dimension.
- [Fit54] P.M. Fitts. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology*, 47 :381–391, 1954.
- [FKN80] Henry Fuchs, Zvi M. Kedem, and Bruce F. Naylor. On visible surface generation by a priori tree structures. *Computer Graphics*, 14(3) :124–133, June 1980.
- [FM03a] P. Fuchs and G. Moreau. *Le Traité de la Réalité Virtuelle*, volume volume 1 : Fondements et interfaces comportementales. Ecole des Mines de Paris, 2 edition, 2003.
- [FM03b] Philippe Fuchs and Guillaume Moreau. *Le traité de la réalité virtuelle*, volume 1. Presses de l’Ecole des Mines, 2 edition, 2003.

- [FWC84] James D. Foley, Victor L. Wallace, and Peggy Chan. The human factors of computer graphics interaction techniques. *IEEE Comput. Graph. Appl.*, 4(11) :13–48, 1984.
- [Gar74] W.R. Garner. *The Processing of Information and Structure*. Lawrence Erlbaum, Potomac, 1974.
- [GCR<sup>+</sup>01] S. Grange, F. Conti, P. Rouiller, P. Helmer, and C. Baur. The Delta Haptic Device. In *Mecatronics*, 2001.
- [Gib62] C. B. Gibbs. Controller design : Interactions of controlling limbs, time-lags, and gains in positional and velocity systems. *Ergonomics*, (5) :385–402, 1962.
- [Gib79] J. Gibson. *The ecological approach to visual perception*. Lawrence Erlbaum Associates, 1979.
- [GLS04] F. Giraud and B. Lemaire-Semail. Position control of a small travelling wave ultrasonic motor. In *ACTUATOR*, 2004.
- [GLSH01] F. Giraud, B. Lemaire-Semail, and J.-P. Hautier. Model and control of a travelling wave ultrasonic motor. In *EPE*, August 2001.
- [Gol99] E.B. Goldstein. *Sensation and Perception*. Pacific Grove, 1999.
- [Gri99] Maarten W. Gribnau. *Two-handed interaction in computer supported 3D conceptual modeling*. PhD thesis, Netherland, 1999.
- [Hac03] Martin Hachet. *Interaction avec des environnements virtuels affichés au moyen d'interfaces de visualisation collective*. PhD thesis, Université Bordeaux I, 2003.
- [Hapa] <http://www.fcs-cs.com> HapticMaster.
- [Hapb] <http://www.haption.com> Haption.
- [HC97] J.P. Hautier and J.P. Caron. *Système automatiques Tome 2 Commande des processus*. Ellipses, 1997.
- [HH97] A. Z. Hajian and R. D. Howe. Identification of the mechanical impedance at the human finger tip. *J. Biomech. Eng.*, 119 :109 – 114, 1997.
- [HPGK94] Ken Hinckley, Randy Pausch, John C. Goble, and Neal F. Kassell. A survey of design issues in spatial input. In *Proceedings of the 7th annual ACM symposium on User interface software and technology*, pages 213–222. ACM Press, 1994.
- [HS93] R. Hollis and S.E. Salcudean. Lorentz Levitation Technology : a New Approach to Fine Motion Robotics, Teleoperation, Haptic Interfaces, and Vibration Isolation. In *Proc. Int'l Symposium for Robotics Research*, 1993.
- [IBM] <http://www.almaden.ibm.com/cs/user/tp/tp.html> IBM.
- [IM95] W. Hagood IV and A-J. McFarland. Modeling of a piezoelectric rotary ultrasonic motor. *IEEE Transactions on ultrasonics, ferroelectrics and frequency control*, 42(2) :210–224, mars 1995.
- [Imm] <http://www.immersion.com> Immersion.

- 
- [JC90] H.D. Jellinek and S. K. Card. Powermice and user performance. In *CHI'90*, pages 213–220. ACM Press, 1990.
- [JL99] A.J. Johansson and J. Linde. Using Simple Force Feedback Mechanisms as Haptic Visualization Tools. In *Proc. IEEE Instrumentation and Measurement Technology Conference*, 1999.
- [JS92] R.J.K. Jacob and L.E. Silbert. The perceptual structure of multidimensional input device selection. In *ACM CHI'92 Human Factors in Computing Systems Conference*, pages 211–218. ACM Press, 1992.
- [JSMM94] R.J.K. Jacob, L.E. Silbert, C. Mcfarlane, and M.P. Mullen. Integrality and Separability of Input Devices. *ACM Transactions on Computer-Human Interaction*, 1(1) :3–26, March 1994.
- [KHKS02] S. Kim, S. Hasegawa, Y. Koike, and M. Sato. Tension based 7-dof force feedback device : Spidar-g. In *Proceedings of the IEEE Virtual Reality Conference 2002*, page 283. IEEE Computer Society, 2002.
- [Léc01] Anatole Lécuyer. *Contribution à l'étude des retours haptique et pseudo-haptique et de leur impact sur les simulations d'opérations de montage/démontage en aéronautique*. PhD thesis, Université Paris XI, 2001.
- [LCK<sup>+</sup>00] A. Lecuyer, S. Coquillart, A. Kheddar, P. Richard, and P. Coiffet. Pseudo-Haptic Feedback : Can Isometric Input Devices Simulate Force Feedback ? In *Proc. of IEEE Int. Conf. on Virtual Reality*, pages 83–90, 2000.
- [Log] <http://www.logitech.com> Logitech.
- [MAc89] I. S. MAcKenzie. A note on the information-theoretic basis for Fitts' law. *Journal of Motor Behavior*, 21 :323–330, 1989.
- [Mas93] Thomas Harold Massie. *Design of a Three Degree of Freedom Force-Reflecting Haptic Interface*. PhD thesis, Massachusetts Institute of Technology, 1993.
- [Mas96] Thomas Massie. Taking the mush out of haptics with infinitely stiff walls. In *The First PHANToM Users Group workshop*. J.K. Salisbury and M.A. Srinivisan, 1996.
- [Max] <http://www.maxonmotor.com> Maxon.
- [MB92] I. S. MacKenzie and W. Buxton. Extending fitts' law to two-dimensional tasks. In *ACM Conference on Human Factors in Computing Systems - CHI'92*, pages 219–226. ACM Press, 1992.
- [MCR90] Jock D. Mackinlay, Stuart K. Card, and George G. Robertson. Rapid controlled movement through a virtual 3d workspace. In *Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, pages 171–176. ACM Press, 1990.
- [MF98] T. E. Milner and D. W. Franklin. Characterization of Multijoint Finger Stiffness : Dependence on Finger Posture and Force Direction. *IEEE Transactions on Biomedical engineering*, 45(11) :1363 – 1375, 1998.
- [MHS01] M. L. McLaughlin, J. P. Hespanha, and G. S. Sukhatme. *Touch in Virtual Environments : haptics and the design of interactive systems*. IMSC Press Multimedia Series, 2001.

- [Mica] <http://www.microchip.com> Microchip.
- [Micb] <http://www.microsoft.com> Microsoft.
- [MKS01] I. Scott MacKenzie, Tatu Kauppinen, and Miika Silfverberg. Accuracy Measures for Evaluating Computer Pointing Devices. In *CHI*, pages 9–16. ACM Press, 2001.
- [MM00] Maurice R. Masliah and Paul Milgram. Measuring the allocation of control in a 6 degree-of-freedom docking experiment. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 25–32. ACM Press, 2000.
- [MPC04] F. Martinot, P. Plénacoste, and C. Chaillou. The digitracker, a three degrees of freedom pointing device. In ACM, editor, *Eurographics Symposium on Virtual Environments (EGVE'04)*, June 2004.
- [MS94] T.H. Massie and J.K. Salisbury. The PHANToM haptic interface : A device for probing virtual objects. In *Proc. ASME Winter Annu. Meeting Symp. Haptic Interfaces for Virtual Environment and Teleoperator Systems*, volume vol. DSC-55-1, pages 295–300, 1994.
- [MSR89] M.J. Massimino, T.B. Sheridan, and J.B. Roseborough. One hand tracking in six degree of freedom. In *IEEE International Conference on Systems, Man, and Cybernetics*, pages 498–503, 1989.
- [MyMS<sup>+</sup>90] Margaret Minsky, Ouh young Ming, Oliver Steele, Frederick P. Brooks, Jr., and Max Behensky. Feeling and seeing : issues in force display. *SIGGRAPH Comput. Graph.*, 24(2) :235–241, 1990.
- [NKK04] S. Naito, Y. Kitamura, and F. Kishino. Steering Law in an Environment of Spatially Coupled Style with Matters of Pointer Size and Trajectory Width. In *APCHI*, pages 305 – 316. LNCS 3101, 2004.
- [OP] <http://www.sgi.com/products/software/performer/> OpenGL Performer.
- [Opea] <http://www.opengl.org> OpenGL.
- [Opeb] <http://www.opensg.org/> OpenSG.
- [Pal04] Alexis Paljic. *Interaction en environnements immersifs et retours d'effort passifs*. PhD thesis, Université Paris 6, 2004.
- [PBBW95] Randy Pausch, Tommy Burnette, Dan Brockway, and Michael E. Weiblen. Navigation and locomotion in virtual worlds via flight into hand-held miniatures. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 399–400. ACM Press, 1995.
- [PH03] J. Pasquero and V. Hayward. STRESS : A Practical Tactile Display System with One Millimeter Spatial Resolution and 700 Hz Refresh Rate. In *Eurohaptics*, pages 94 – 110, 2003.
- [PLS96] Niall R. Parker, Peter D. Lawrence, and Septimiu E. Salcudean. Velocity Controller with Force Feedback Stiffness control. United States Patent number 5,513,100, Apr. 30, 1996.



- 
- [Pou74] E.C. Poulton. *Tracking skill and manual control*. New York : Academic Press, 1974.
- [PW94] Mark A. Paton and Colin Ware. Passive force feedback for velocity control. In *Conference companion on Human factors in computing systems*, pages 255–256. ACM Press, 1994.
- [QNX] <http://www.qnx.com> QNX.
- [RKC00] S. Redon, A. Kheddar, and S. Coquillart. An Algebraic Solution to the Problem of Collision Detection for Rigid Polyhedral Objects. In *IEEE International Conference on Robotics and Automation*, pages 24 – 28, 2000.
- [RKK97] D.C. Ruspini, K. Kolarov, and O. Khatib. The Haptic Display of Complex Graphical Environments. In *ACM SIGGRAPH*, pages 345–352, 1997.
- [ROC97] Christopher Richard, Allison Okamura, and Mark R. Cutkosky. Feeling is believing : Using a force-feedback joystick to teach dynamic systems. In *ASME IMECE Annual Symposium on Haptic Interfaces*, 1997.
- [RS90] Joseph D. Rutledge and Ted Selker. Force-to-motion functions for pointing. In *Proceedings of the IFIP TC13 Third International Conference on Human-Computer Interaction*, pages 701–706. North-Holland, 1990.
- [RTA] <http://www.reachin.se> Reachin Technologies AB.
- [Sag77] G.H. Sage. *Introduction to Motor Behavior, a Neuropsychological Approach*. Addison-Wesley Publishing Company, 2 edition, 1977.
- [SB97] M.A. Srinivasan and C. Basdogan. Haptics in Virtual Environments : Taxonomy, Research Status, and Challenges. *Computer and Graphics*, 31(4) :393 – 404, 1997.
- [Sen] <http://www.sensable.com> Sensable.
- [Ser] <http://www.servotogo.com> Servotogo.
- [SGF99] Thomas A. Stoffregen, Kathleen M. Gorday, and Steven B. Flynn. Perceiving affordances for another person’s actions. In *Journal of Experimental Psychology Human Perception and Performance*, pages 120–136, 1999.
- [SMO04] C. Schneider, T. Mustufa, and A. M. Okamura. A Magnetically-Actuated Friction Feedback Mouse. In *Eurohaptics*, pages 330–337, June 2004.
- [Spa] <http://www.3dconnexion.com> Spaceball.
- [SV97] S.E. Salcudean and T.D. Vlaar. On the emulation of stiff walls and static friction with a magnetically levitated input/output device. *Trans. ASME, Journal of Dynamic Systems, Measurement, and Control*, 119(1) :127–132, 1997.
- [TSEC94] H. Tan, M. Srinivasan, B. Eberman, and B. Chang. Human factors for the design of force-reflecting haptic interfaces. In *Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems, Chicago, IL, November 1994. Proceedings of the ASME Winter Annual Meeting.*, pages 353–359, 1994.

- [Wel68] A. T. Welford. *The fundamentals of skill*. London : Methuen, 1968.
- [WF02] G. Welch and E. Foxlin. Motion tracking : No silver bullet, but a respectable arsenal. *IEEE Computer Graphics and Applications, special issue on Traking*, 22(6) :24 – 38, 2002.
- [Wic92] C.D. Wickens. *Engineering Psychology and Human Performance*. Harper Collins Publishers, 1992.
- [WJ88] Colin Ware and Danny R. Jessome. Using the bat : A six-dimensional mouse for object placement. *IEEE Computer Graphics and Applications, special issue on Traking*, 8(6) :65 – 70, 1988.
- [WKA92] Y.T. Wang, V. Kumar, and J. Abel. Dynamics of rigid bodies undergoing multiple friction contacts. In *IEEE International Conference on Robotics and Automation*, pages 2764–2769, 1992.
- [WKS99] Somsak WALAIRACHT, Yasuharu KOIKE, and Makoto SATO. A New Haptic Display for Both-Hands-Operation :SPIDAR-8. In *Proc. of ISPACS'99*, pages 569–572, 1999.
- [WO90] C. Ware and S. Osborne. Exploration and virtual camera control in virtual three dimensional environments. In *Proc. of ACM Symposium on Interactive 3D Graphics*, pages 175–183, 1990.
- [Zha95] Shumin Zhai. *Human Performance in Six Degree of Freedom Input Control*. PhD thesis, University of Toronto, 1995.
- [ZKSS99] S. Zhai, E. Kandogan, B. Smith, and T. Selker. In Search of the "Magic Carpet", Design and Experimentation of a 3D Navigation Interface. *Journal of Visual Languages and Computing*, 10(1) :3 – 17, 1999.
- [ZM93] S. Zhai and P. Milgram. Human performance evaluation of manipulation schemes in virtual environments. In *IEEE Virtual Reality Annual Symposium (VRAIS'93)*, pages 155–161, 1993.
- [ZM98] S. Zhai and P. Milgram. Quantifying Coordination in Multiple DOF Movement and Its Application to Evaluating 6 DOF Input Devices. In *Proc. of CHI'98*, pages 320–327, 1998.
- [ZMB96] Shumin Zhai, Paul Milgram, and William Buxton. The Influence of Muscle Groups on Performance of Multiple Degree-of-Freedom Input. In *Proc. of CHI96 : ACM Conference on Human Factors in Computing Systems*, pages 308–315, April 13-18 1996.
- [ZS95] C.B. Zilles and J.K. Salisbury. A Constraint-based God-Object Method For Haptic Display. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 146–151, 1995.

## Résumé

La caractéristique commune de la plupart des périphériques 3D existants, qu'ils soient isotoniques comme le PHANToM ou élastiques comme la SpaceMouse, est l'assemblage sur un unique effecteur de tous les degrés de liberté. Ces interfaces ont certes l'avantage d'être intuitives mais elles ont aussi l'inconvénient de limiter le contrôle individuel des degrés de liberté, en particulier pour les périphériques élastiques. Les périphériques à degrés de liberté séparés possèdent, quant à eux, des propriétés intéressantes encore peu exploitées. Leur simplicité mécanique a notamment l'avantage de simplifier le retour d'effort.

Le DigiHaptic est une proposition de périphérique haptique à trois degrés de liberté séparés, pouvant être utilisé en modes isotonique et élastique. Après les étapes de conception et de réalisation, nous exposons les nouveaux outils développés pour la commande du DigiHaptic.

Une analyse des différents modes d'interaction pour des tâches de manipulation et de navigation est ensuite réalisée. Nous proposons ainsi des métaphores d'utilisation du dispositif, en mode hybride isotonique-élastique, pour la manipulation en environnements 3D de grandes dimensions. L'intégration du retour haptique en mode élastique est également discutée et illustrée dans une application de navigation en environnement 3D avec retour d'effort.

Le périphérique a enfin été évalué dans des tâches de suivi de trajectoires 3D et de navigation en environnements 3D, dans le but de comparer la séparation des degrés de liberté du DigiHaptic en mode élastique aux degrés de liberté assemblés de la SpaceMouse. Ces études ont montré que les utilisateurs coordonnent efficacement les degrés de liberté. Et si le DigiHaptic est parfois moins rapide, il est toujours plus précis. Par ailleurs, il semble plus adapté à la navigation que la SpaceMouse.

Ces travaux, qui soulignent les capacités du DigiHaptic, laissent présager une nouvelle classe de périphériques haptiques à degrés de liberté séparés très prometteurs.

**Mots-clés:** Haptique, périphérique 3D, interaction homme-machine, retour d'effort, impédance, navigation, métaphores, évaluation.

## Abstract

The common characteristic of most 3D input devices, which are either isotonic like the PHANToM or elastic like the SpaceMouse, is that all the degrees of freedom are gathered on one end effector. These devices have the advantage to be intuitive but the drawback to limit the individual control of each degree of freedom, which is particularly true for elastic devices. Devices with separated DOF have interesting properties that have not been exploited yet. Their mechanical simplicity has in particular the advantage to simplify the force feedback.

The DigiHaptic is a proposition of haptic device with three separated degrees of freedom, that can be used in isotonic or elastic modes. After the stages of conception and realization, we expose the new tools developed to command the DigiHaptic.

An analysis of the different interaction modes for manipulation and navigation tasks is then realized. Thus we propose metaphors to use the device in hybrid isotonic elastic mode, for the manipulation in large scale 3D environments. The haptic feedback in elastic mode is also discussed and illustrated in a 3D navigation application with force feedback.

Finally, the device was evaluated in a 3D steering task experiment and a 3D navigation task, in order to compare the influence of the separated DOF of the DigiHaptic to the integrated DOF of the SpaceMouse. The experiments showed that users coordinate efficiently separated degrees of freedom. And if the DigiHaptic is sometimes slower than the SpaceMouse, it is always more accurate. Furthermore, the DigiHaptic is better for the navigation task than the SpaceMouse.

**Keywords:** Haptic, 3D device, human-computer interaction, force feedback, impedance, navigation, metaphors, evaluation.