

UNIVERSITÉ D'ANGERS

Année 2006

N° Ordre : 784

**CONTRIBUTION À L'ANALYSE DE L'INTERACTION DYNAMIQUE
EN ENVIRONNEMENT VIRTUEL :
PRIMITIVES ET AIDES LOGICIELLES**

THÈSE DE DOCTORAT

Spécialité : Automatique et Informatique Appliquée

ÉCOLE DOCTORALE D'ANGERS

Présentée et soutenue publiquement

le : 19 Octobre 2006

à : Angers

par : François-Xavier INGLESE

Devant le jury ci-dessous :

Mr Patrick MILLOT,	Professeur à l'Université de Valenciennes et du Hainaut-Cambrésis	Rapporteur
Mr Bruno ARNALDI,	Professeur à l'INSA de Rennes	Rapporteur
Mr Jean-Louis FERRIER,	Professeur à l'Université d'Angers	Directeur de Thèse
Mr Paul RICHARD,	Maître de Conférences à l'Université d'Angers	Co-encadrant
Mr Benoît BARDY,	Professeur à l'Université de Montpellier 1	Examineur
Mr Gilles MONTAGNE,	Maître de Conférences à l'Université de la Méditerranée	Examineur

Laboratoire d'Ingénierie des Systèmes Automatisés EA 4014

Institut des Sciences et Techniques de l'Ingénieur d'Angers
62, Avenue Notre Dame du Lac 49000 ANGERS

Table des matières

Liste des figures	9
Liste des tableaux	13
Remerciements	15
Introduction	17
1 Réalité Virtuelle et Environnement Dynamique	21
1.1 Introduction	21
1.2 Réalité Virtuelle	21
1.2.1 Définition	21
1.2.2 Historique	22
1.2.3 Structure d'un système de RV	23
1.2.4 Interfaces Motrices	24
1.2.4.1 Les capteurs de localisation	24
1.2.4.2 Les gants de données	26
1.2.5 Interfaces sensorielles et sensori-motrices	27
1.2.5.1 Les interfaces visuelles	28
1.2.5.2 Les interfaces haptiques	30
1.2.5.3 Le SPIDAR	31
1.2.6 La Plate-forme PREWISE	34
1.2.7 Techniques d'Interaction	34
1.2.7.1 Contrôle de l'Application	34
1.2.7.2 Sélection - Manipulation	35
1.2.7.3 Navigation	36
1.2.7.4 Interaction Dynamique	38
1.3 Travaux sur les Objets en Mouvement	39
1.3.1 Inconvénients et Avantages de la RV	40

1.3.1.1	Inconvénients	40
1.3.1.2	Avantages	41
1.4	Aides Logicielles	42
1.4.1	Aides Logicielles Comportementales	42
1.4.2	Fixtures Virtuelles	43
1.4.3	Aides classiques des Jeux Videos	44
1.4.4	Aides pour l'Interaction Dynamique	47
1.5	Conclusion	48
2	Analyse de l'interaction dynamique	49
2.1	Introduction	49
2.2	Primitives d'Interaction Dynamique	49
2.2.1	Modélisation des Primitives d'Interaction Dynamique	51
2.2.1.1	Objets Dynamiques	51
2.2.1.2	Lancer	56
2.2.1.3	Attraper	59
2.2.1.4	Frapper	63
2.2.1.5	Général	67
2.2.2	Les exigences de l'interaction dynamique	68
2.2.2.1	Problématique	68
2.2.2.2	Émetteur	68
2.2.2.3	Sujet	68
2.2.2.4	Geste	68
2.2.2.5	Cible	68
2.2.2.6	Conclusion	69
2.3	Conception d'Aides Logicielles pour l'Interaction Dynamique	69
2.3.1	Objectifs et moyens	69
2.3.2	Structure d'une ALID	69
2.3.2.1	Types de valeurs	72
2.3.2.2	Paramètres	73
2.3.2.3	Exemple d'ALID	77
2.4	Conclusion	79
3	PID : Lancer et Attraper	81
3.1	Introduction	81
3.2	Justification	81
3.3	Conception et développement du système	82

3.3.1	Analyse et modélisation de l'interaction	82
3.3.1.1	Informations rétiniennes et extra-rétiniennes	83
3.3.1.2	Nos Lancer/Attraper et la Modélisation des Interactions Dynamiques	83
3.3.2	Périphériques et technique d'interaction	83
3.3.3	Conception et développement logiciels	85
3.3.3.1	Machine à États	85
3.3.3.2	Ouverture/Fermeture de la Main Virtuelle	88
3.3.3.3	Gestion du Lancer	91
3.3.3.4	Modèles 3D	91
3.4	Expérimentations	93
3.4.1	Lancer de balle sur une cible fixe	93
3.4.1.1	Objectif de l'expérience	93
3.4.1.2	Description de la tâche	93
3.4.1.3	Environnement virtuel	94
3.4.1.4	ALID n°1 : TrajectoireParabolique	94
3.4.1.5	ALID n°2 : TrajectoireSimultanée	95
3.4.1.6	Protocole expérimental	96
3.4.1.7	Analyse des résultats	97
3.4.2	Attraper d'objets en approche frontale	101
3.4.2.1	Objectif de l'expérience	101
3.4.2.2	Description de la tâche	101
3.4.2.3	Environnement virtuel	102
3.4.2.4	Protocole expérimental	102
3.4.2.5	Analyse des résultats	104
3.4.3	Discussion	106
3.4.4	Attraper de balle en approche oblique	109
3.4.4.1	Objectif de l'expérience	109
3.4.4.2	Description de la tâche	110
3.4.4.3	Monde virtuel	110
3.4.4.4	ALID n°1 : DisqueVertPermanent	111
3.4.4.5	ALID n°2 : DisqueVertTemporaire	112
3.4.4.6	Protocole expérimental	114
3.4.4.7	Analyse des résultats et discussion	116
3.5	Conclusion	123

4	Application au Lancer Discret : Le Tir à l'Arc	125
4.1	Introduction	125
4.2	Justification	125
4.3	Conception et développement du système	126
4.3.1	Analyse et modélisation de l'interaction	126
4.3.2	Périphériques et technique d'interaction	129
4.3.3	Conception et développement logiciels	131
4.3.3.1	Machine à États	132
4.3.3.2	Monde Virtuel	137
4.3.3.3	Modèles 3D	139
4.3.3.4	Arrière-plan : Skybox	140
4.4	Expérimentations	141
4.4.1	Influence du coefficient de recul	141
4.4.1.1	Objectif de l'expérience	143
4.4.1.2	Description de la tâche	144
4.4.1.3	Monde virtuel	145
4.4.1.4	Protocole expérimental	145
4.4.1.5	Analyse et discussion des résultats	145
4.4.2	Influence des ALID visuelles	149
4.4.2.1	Objectif de l'expérience	149
4.4.2.2	Description de la tâche	149
4.4.2.3	Monde virtuel	149
4.4.2.4	ALID n°1 : PointRouge	150
4.4.2.5	ALID n°2 : ScorePrédictif	152
4.4.2.6	ALID n°3 : CaméraEmbarquée	152
4.4.2.7	Protocole expérimental	154
4.4.2.8	Analyse des résultats et discussion	154
4.5	Conclusion	158
5	Application au Lancer Continu : Le Bowling	159
5.1	Introduction	159
5.2	Justification	159
5.3	Conception et développement du système	160
5.3.1	Analyse et modélisation de l'interaction	160
5.3.1.1	Angle d'Attaque	162
5.3.1.2	Considérations énergétiques	163
5.3.1.3	Direction de la Force Initiale	164

5.3.1.4	Friction, Crochet et Angle d'Impact	165
5.3.1.5	Pin Action	165
5.3.1.6	Le Bowling et la Modélisation des Interactions Dynamiques .	168
5.3.2	Périphériques et technique d'interaction	168
5.3.3	Conception et développement logiciels	170
5.3.3.1	Machine à États	170
5.3.3.2	Modèles 3D	176
5.3.3.3	Modélisation Physique	177
5.3.3.4	Gestion du Lancer	178
5.3.3.5	Modes de Jeu	182
5.4	Expérimentation	182
5.4.1	Objectif de l'expérience	182
5.4.2	Description de la tâche	182
5.4.3	Monde virtuel	183
5.4.4	ALID : TrajectoireRectiligne	184
5.4.5	Protocole expérimental	185
5.4.6	Analyse des résultats et discussion	185
5.4.6.1	Scores Moyens	186
5.4.6.2	Nombre de Strikes	186
5.5	Conclusion	187
Conclusion et perspectives		189
Bibliographie		192
Annexes		199
A La Méthode Statistique ANOVA		201
A.1	Description de la méthode ANOVA	201
A.1.1	La variable "F"	201
A.1.2	La variable "p"	201
A.1.3	Les valeurs MS	202
A.1.4	Les valeurs df	202
A.2	Les effets croisés	203
A.2.1	Le t-test	203
B Application "PID" : Fichiers de Configuration		205
B.1	Fichiers de Configuration	205

Table des figures

1.1	Schéma d'un système de RV [Coi95]	23
1.2	Le système Polhemus Patriot	25
1.3	Le gant de données 5DT Data Glove Ultra	27
1.4	Un Visiocasque : le 5DT HMD 800 de Fifth Dimension Technologies	28
1.5	Un Workbench : le TAN Holobench	29
1.6	Un gant haptique : le Cybergrasp de Virtual Technologies	30
1.7	Un bras articulé à retour d'effort : le Phantom Omni de Sensable	31
1.8	Illustration du SPIDAR	32
1.9	L'utilisateur dans le cube du SPIDAR (a) ; L'utilisation des fils pour la position et le retour d'effort (b)	32
1.10	Enveloppes de travail du SPIDAR	33
1.11	Capture d'écran du système "TULIP" [BW01]	35
1.12	Capture d'écran du VRMAT [PWBI97] illustrant le concept de "main virtuelle"	37
1.13	Exemple de Fixture [Mor02]	44
1.14	Capture d'écran du jeu "Leaderboard" sur Atari ST	46
1.15	Capture d'écran du jeu "Side Pocket" sur NES	46
1.16	Capture d'écran du jeu "Pro Tennis WTA Tour" sur Gameboy Advance	47
2.1	Les différents États de la balle lors d'un Lancer.	52
2.2	Modélisation du Lancer	57
2.3	Modélisation de l'Attraper	61
2.4	Modélisation du Frapper	64
2.5	Modélisation des Interactions Dynamiques : le D-BiCube	67
2.6	Capture d'écran du jeu "Mario Golf : Toadstool Tour"	74
2.7	Positions des anneaux le long d'une trajectoire parabolique	77
3.1	Notre lancer de balle situé sur le Cube de la Modélisation du Lancer	84
3.2	Notre Attraper de balle situé sur le Cube de la Modélisation de l'Attraper	84
3.3	Diagramme d'États de l'Application	89

3.4	Diagramme d'États de la Main Virtuelle (Lancer-Attraper))	90
3.5	Diagramme de la Gestion du Projectile	92
3.6	Les 10 modèles de main	93
3.7	Capture d'écran de la simulation de Lancer	94
3.8	Dispositif de la balle fixée au gant de données	97
3.9	Distance moyenne par Condition	99
3.10	Distance Latérale par Condition	99
3.11	Distance en Profondeur moyenne par Condition	100
3.12	Temps de Visée moyen par Condition	101
3.13	Capture d'écran de la simulation d'Attraper dans les Conditions 1 à 4	103
3.14	Points de passage du projectile dans le plan vertical de la Main	104
3.15	Taux de Balles Attrapées moyen par Condition	105
3.16	Taux de Cylindres Attrapés moyen par Condition	105
3.17	Distance Main Fermée - Balle moyenne par Condition	106
3.18	Distance Main Fermée - Cylindre moyenne par Condition	107
3.19	Photo de la bague du SPIDAR	110
3.20	Capture d'écran de la simulation d'attraper de balle en approche oblique	111
3.21	Points de passage de la balle dans le plan horizontal de la Main	115
3.22	Balles Attrapées par Groupe	117
3.23	Moyenne de Balles Attrapées par Groupe pour la Session 1	117
3.24	Moyenne de Balles Attrapées par Groupe pour la Session 2	118
3.25	Moyenne des sommes des distances Main-Balle, par Groupe, pour la Session 1	120
3.26	Moyenne des sommes des distances Main-Balle, par Groupe, pour la Session 2	120
3.27	Numérotation des Points de passage de la balle dans le plan de la main	121
3.28	Moyenne de balles attrapées par Points de passage	122
4.1	Automate de l'Archer	128
4.2	Le Tir à l'Arc situé sur le cube de la Modélisation du Lancer	128
4.3	Position idéale des capteurs de position sur l'arc, au repos à gauche, en action à droite	130
4.4	Position des capteurs pour notre simulation et Offset vertical d_y généré	131
4.5	Utilisateur de notre simulation de tir à l'arc	132
4.6	Diagramme d'États de la Simulation	133
4.7	Schéma du Calcul du Score	134
4.8	Diagramme d'États des Doigts Virtuels	135
4.9	Diagramme d'États de la Main Virtuelle (Tir à l'Arc)	136
4.10	Diagramme d'États de l'Arc	136

4.11	Diagramme d'États de la Flèche	137
4.12	Capture d'écran de la simulation de tir à l'arc	138
4.13	Détection de Collision Flèche-Cible : nécessité des "Positions intermédiaires" .	139
4.14	Schéma de la Procédure d'Intégration des Objets 3D	140
4.15	Théorie de la Skybox	141
4.16	(a) Textures de la Skybox ; (b) Capture d'écran orientée vers une arête du cube	142
4.17	Méthode de Calibration	143
4.18	Force appliquée par l'utilisateur en fonction de l'allonge	144
4.19	Score moyen par Condition	146
4.20	Force moyenne appliquée par Condition	147
4.21	Temps de visée moyen par Condition	147
4.22	Point d'impact et Score prédictifs	150
4.23	Moyenne de Score en Session d'Entraînement (Conditions 1 à 4)	155
4.24	Moyenne de Score en Session de Test (Conditions 1b à 4b)	156
4.25	Moyenne de Score en Session d'Entraînement (Conditions 1 et 5) et de Test (Conditions 1b et 5b)	157
5.1	Positionnement relatif des quilles	161
5.2	Trajectoire en crochet	161
5.3	Split 7-10	162
5.4	Lanceur en situation réelle durant la course	164
5.5		165
5.6	Renversement des quilles	166
5.7	Automate du joueur de bowling	167
5.8	Bowling réel et virtuel situés sur le Cube de la Modélisation du Lancer	168
5.9	Diagramme d'États de la Simulation du bowling	173
5.10	Diagramme d'États de la Main Virtuelle (Bowling)	174
5.11	Diagramme d'États de la Boule virtuelle	174
5.12	Diagramme d'États d'une Quille virtuelle	176
5.13	Schéma de la Gestion des Objets 3D	177
5.14	Capture d'écran d'une simple collision Boule-Quille	178
5.15	Capture d'écran d'un "pin action"	179
5.16	Trajectoire de la main	179
5.17	Échantillons de la trajectoire de la main	180
5.18	Vecteur Vitesse calculé à partir des échantillons de la trajectoire de la main . .	180
5.19	Trajectoire de la main avec un temps d'arrêt "t" au point "p"	180
5.20	Échantillons de la Trajectoire comportant un temps d'arrêt	181

5.21	Vecteur Vitesse calculé à partir des échantillons de la trajectoire comportant un temps d'arrêt	181
5.22	Vecteur Vitesse adapté à la trajectoire de la main	181
5.23	Monde Virtuel de la simulation de Bowling avec affichage de la Trajectoire Rectiligne	183

Liste des tableaux

1.1	Structure d'une fixture virtuelle, d'après Otmane, Colle, Mallem et Hoppenot. .	45
2.1	Structure d'une Aide Logicielle pour l'Interaction Dynamique.	72
2.2	Tableau des Sous-Types.	73
2.3	Tableau des composantes d'une Modification.	76
2.4	Structure de l'ALID "GuideAnneaux".	78
2.5	Composantes de ModificationVert.	79
2.6	Composantes de ModificationRouge.	79
3.1	Structure de l'ALID "TrajectoireParabolique".	95
3.2	Structure de l'ALID "TrajectoireSimultanée".	96
3.3	Ordre des Conditions par Sujet (Lancer).	98
3.4	Ordre des Conditions par Sujet (Attraper).	102
3.5	Suggestion de Protocole pour l'évaluation de nos ALID.	108
3.6	Suggestion de Protocole pour le Lancer.	109
3.7	Structure de l'ALID "DisqueVertPermanent".	112
3.8	Structure de l'ALID "DisqueVertTemporaire".	113
3.9	Conditions pour chaque série de 50 balles, par groupe.	114
3.10	Balles attrapées par les sujets, par groupe et par session	116
3.11	Distances Main-Balle totales, par groupe et par session	119
4.1	Structure de l'ALID "PointRouge".	151
4.2	Composantes de PointVert.	151
4.3	Structure de l'ALID "ScorePrédictif".	152
4.4	Composantes de ScoreVert.	153
4.5	Structure de l'ALID "CaméraEmbarquée".	153
4.6	Conditions expérimentales par Groupe.	154
5.1	Structure de l'ALID "TrajectoireRectiligne".	184
5.2	Conditions pour chaque série de 20 boules, par groupe.	185

5.3	Nombre moyen de Quilles renversées par Sujet, par Groupe et par Session . . .	186
5.4	Nombre moyen de Strikes réalisés par Sujet, par Groupe et par Session . . .	186

Remerciements

Le travail présenté dans ce mémoire a été réalisé au Laboratoire d'Ingénierie des Systèmes Automatisés de l'Institut des Sciences et Techniques de l'Ingénieur d'Angers. J'adresse donc mes plus vifs remerciements à Jean-Louis Ferrier, responsable de ce Laboratoire et Professeur à l'Université d'Angers, pour avoir bien voulu m'accueillir.

Je remercie aussi Paul Richard, Maître de Conférences à l'Université d'Angers pour avoir accepté d'encadrer mon travail de thèse.

Je tiens également à exprimer ma profonde gratitude au Professeur Patrick Millot et à Monsieur Bruno Arnaldi pour m'avoir fait l'honneur d'accepter d'examiner ce travail en qualité de rapporteurs.

Mes remerciements s'adressent aussi au Professeur Benoît Bardy et à Monsieur Gilles Montagne pour avoir accepté de faire partie du jury et d'examiner mon travail.

Je voudrais par ailleurs remercier tous les sujets qui ont bien voulu participer aux différentes expériences que j'ai pu mener, en restant relativement confiants et de bonne humeur malgré des conditions parfois étranges. Sans eux rien n'aurait été possible.

Enfin, j'adresse mes plus vifs remerciements et ma sympathie à tous mes collègues, collaborateurs et amis, en particulier Solenna, Angèle et Mickaël, ainsi qu'à Sébastien pour ses conseils avisés en matière de rédaction.

Introduction

La Réalité Virtuelle (RV) et les Environnements Virtuels (EV) apparaissent comme une technologie pouvant couvrir un grand champ d'applications telles que l'imagerie bio-médicale, la planification d'interventions chirurgicales, la visualisation de données complexes, les jeux vidéo, l'entraînement sportif, etc.

Les techniques d'interaction qui ont été développées impliquent dans la plupart des cas, la navigation dans les environnements synthétiques, ou la sélection/manipulation d'objets virtuels. Dans ce contexte, il a été proposé par Philippe Fuchs [FMP03] une analyse permettant de concevoir des Aides Logicielles Comportementales (ALC) qui ont pour but de faciliter l'interaction avec les environnements ou les objets virtuels.

Différentes applications et simulateurs reposant sur les techniques de réalité virtuelle impliquent une interaction avec des objets virtuels en mouvement [BFM⁺03] [PXH⁺03] [CBR⁺06] [ZM03] [IKD⁺05] [SMWB03]. Dans certains cas, les simulateurs ont été développés dans un contexte ludique et n'ont jamais été évalués, dans d'autres cas, ils ont servi de support à des activités de recherche.

La contribution du travail présenté dans ce mémoire réside dans l'analyse de l'interaction dynamique en environnement virtuel, et plus particulièrement lors de tâches impliquant le lancer, le frapper et l'interception d'objets en mouvement. Une modélisation originale est proposée. Cette modélisation permet de proposer différentes aides logicielles dans le contexte de l'interaction dynamique (c'est-à-dire impliquant des objets en mouvement ou que l'humain doit mettre en mouvement). Les applications potentielles de ce travail se situent dans le domaine sportif, les jeux immersifs, etc.

Le **premier chapitre** de ce mémoire a pour but d'analyser la place des environnements dynamiques en Réalité Virtuelle. Nous débutons cette partie par un historique de la Réalité Virtuelle puis nous présentons la structure générale d'un système de RV. Cette dernière met en exergue l'importance des *interfaces* utilisées, aussi décrivons-nous les interfaces intéressantes

pour l'interaction dynamique. Nous faisons alors un point sur les principales *interactions* reconnues en réalité virtuelle et nous tâchons de repérer la place de l'interaction dynamique parmi celles-ci. Par la suite, nous traitons des travaux de RV comportant des objets en mouvement, et nous faisons une critique des *avantages et inconvénients* de la RV. Enfin, nous nous intéressons aux *Aides Logicielles* couramment utilisées et à leur intérêt pour l'Interaction Dynamique.

Le **second chapitre** constitue une contribution à l'analyse de l'interaction dynamique en environnement virtuel. Nous commençons par décrire les trois *Primitives d'Interaction Dynamique* (PID), c'est-à-dire les trois interactions de base possibles avec un objet en mouvement. Ces PID sont constituées du *Lancer*, de l'*Attraper* et du *Frapper*, et pour chacune d'elles nous proposons une modélisation. Nous présentons également une modélisation les regroupant toutes : le *D-BiCube*. Enfin, nous proposons une structure d'aides logicielles spécifiquement destinées à faciliter la réalisation et/ou l'apprentissage de tâches d'interaction dynamique en environnement virtuel (les *ALID*).

Avec le **troisième chapitre**, nous démarrons une série d'expérimentations ayant pour but l'évaluation et la validation des *ALID* proposées précédemment, en observant leurs effets sur la *performance* (capacités et apprentissage) de sujets humains lors de tâches impliquant une interaction dynamique.

Dans un premier temps, nous justifions le choix des tâches simples de lancer de balle et d'interception d'objets en tant que cadre à l'évaluation de nos Aides Logicielles pour l'Interaction Dynamique. Nous décrivons dans un deuxième temps les choix de conception et le développement logiciel de l'application qui a été créée, avant de détailler deux expérimentations que nous avons menées : l'une portant sur le lancer de balle, l'autre sur l'interception d'objets en approche frontale. Après une analyse des résultats obtenus, nous faisons une critique de la simulation et du protocole utilisé. Cette discussion aboutit à la proposition d'un nouveau *protocole expérimental* destiné à l'évaluation des *ALID*. Nous nous y conformons dans notre seconde étude sur l'interception de balle, en approche oblique cette fois-ci, qui nous permet de valider l'intérêt des Aides Logicielles pour l'Interaction Dynamique pour la *réalisation* d'une telle tâche.

Dans le **quatrième chapitre**, le *tir à l'arc* nous sert de base à l'évaluation et la validation de nos *ALID*. Nous débutons cette section par une justification de l'activité du tir à l'arc comme cadre d'étude de l'interaction dynamique. Le processus de conception et de développement de l'application de tir à l'arc qui a été créée est ensuite décrit. Enfin, nous présentons les deux expérimentations conduites avec cette simulation. La première porte sur l'importance du *recul visuel* de la flèche tandis que la seconde confirme l'hypothèse que les *ALID* peuvent être utiles

pour l'*apprentissage* d'une tâche autant que pour sa réalisation.

Enfin, le **dernier chapitre** nous amène à porter notre attention sur le *bowling*. Nous entamons cette partie par les raisons qui nous ont fait juger ce sport pertinent pour notre étude de l'interaction dynamique. Nous expliquons ensuite les choix matériels et les orientations logicielles qui ont abouti à la conception de notre simulation de bowling. Enfin, pour clore cet ultime chapitre, nous décrivons l'expérimentation qui a été conduite sur cette application.

Chapitre 1

Réalité Virtuelle et Environnement Dynamique

1.1 Introduction

Ce chapitre constitue une analyse de l'intérêt porté aux environnements dynamiques en Réalité Virtuelle (RV). Dans un premier temps, nous commençons par préciser ce qu'est la Réalité Virtuelle et par en faire un bref historique. A travers la structure d'un système de RV, nous faisons ressortir l'importance des *périphériques d'entrée et de sortie*, et nous présentons ceux qui peuvent être utilisés pour l'interaction dynamique. Puis, dans un deuxième temps, nous décrivons les principales interactions reconnues en réalité virtuelle et faisons état de la place de l'interaction dynamique dans celles-ci. Après avoir fait le point sur les travaux traitant d'activités comportant des objets en mouvement, nous nous intéressons aux Aides Logicielles couramment utilisées, et critiquons leur intérêt pour l'Interaction Dynamique.

1.2 Réalité Virtuelle

1.2.1 Définition

La Réalité Virtuelle est un ensemble de techniques et de technologies dont le but est l'immersion d'un utilisateur humain dans un Environnement Virtuel (EV) dans lequel il peut interagir en temps réel de manière naturelle ou pseudo-naturelle.

Cette immersion et cette interaction nécessitent l'emploi d'interfaces spécifiques permettant de mesurer les mouvements de l'utilisateur et de stimuler ses récepteurs sensoriels.

1.2.2 Historique

Les concepts de *Réalité Virtuelle* (RV) ou d'*Environnements Virtuels* (EV) ne sont pas très récents. Ainsi, le système "Sensorama" qui permettait à un utilisateur de se promener virtuellement à moto dans les rues de New York a été développé par Heilig en 1960 [Hei60]. Cinq années plus tard, en 1965, Ivan Sutherland publiait "The Ultimate Display" qui approfondissait l'idée d'immersion dans l'image au moyen de casques de visualisation [Sut65].

Néanmoins, ce n'est qu'au début des années 80, et consécutivement à l'apparition des ordinateurs graphiques à haute performance, que différents laboratoires se mirent à développer des projets utilisant la RV. Le MIT a ainsi mis au point en 1983 une application où, dans un EV assez sommaire, un utilisateur pouvait manipuler un objet graphique avec la main (et non pas avec un dispositif de pointage classique) [Sch83]. De leur côté, les ingénieurs de la NASA démarrèrent entre 1981 et 1988 les projets "VIVED" (Virtual Visual Environment Display) et "VIEW" (Virtual Interactive Environment Workstation). Ces projets avaient pour objectif la mise au point d'interfaces naturelles pour des applications de téléopération d'une part, et de surveillance et de maintien à distance de vastes systèmes d'informations d'autre part.

Toutefois, et malgré l'enthousiasme des chercheurs du monde entier pour ces technologies, il aura fallu attendre le 6 juin 1989 pour qu'Autodesk et VPL ne rendent publiques les premiers systèmes de RV. C'est aussi cette année là que les périphériques de RV font leur entrée dans le monde des jeux video, par l'intermédiaire du PowerGlove, fabriqué par Mattel jusqu'en 1991 pour la console Nintendo Entertainment System (NES).

A cette époque, l'engouement du grand public pour la RV s'accroît, mais il sera déçu par les piètres performances des dispositifs mis à sa disposition, et par le prix relativement prohibitif des interfaces performantes. La Réalité Virtuelle est alors réservée aux laboratoires de recherche et aux grandes entreprises ayant les moyens de soutenir des projets aux besoins financiers très élevés dans l'optique de rester concurrentiel. C'est ainsi que de nombreux travaux ont été réalisés dans des domaines d'application tels que la robotique, la médecine, l'automobile, l'aéronautique et l'aérospatial, ou la représentation d'ensemble de données scientifiques.

Cependant, le développement depuis quelques années de systèmes de RV "low-cost" (à bas coûts) a remis les techniques de Réalité Virtuelle sur le devant de la scène, non plus en tant que "technologie du futur", mais plutôt comme une composante innovante de la vie de tous les jours. Cette intégration en douceur chez le particulier s'est faite là encore grâce aux jeux video, tout d'abord avec un jeu de danse (Dance Dance Revolution) développé par Konami en version arcade en 1998 et adapté sur Sony Playstation en 1999, puis par un système de maracas pour le

jeu Samba de Amigo (Arcade et Nintendo Gamecube par la Sonic Team de Sega, 2000). Mais c'est en 2002 avec l'Eyetoy de Sony, développé pour la Playstation 2, que la Réalité Virtuelle rejoindra définitivement le quotidien du grand public.

1.2.3 Structure d'un système de RV

La figure 1.1 illustre la structure d'un système de RV selon Coiffet [Coi95]. Selon sa théorie, tout système de RV peut être décomposé en deux grands sous-ensembles : l'*Homme* d'une part et le *Système* d'autre part.

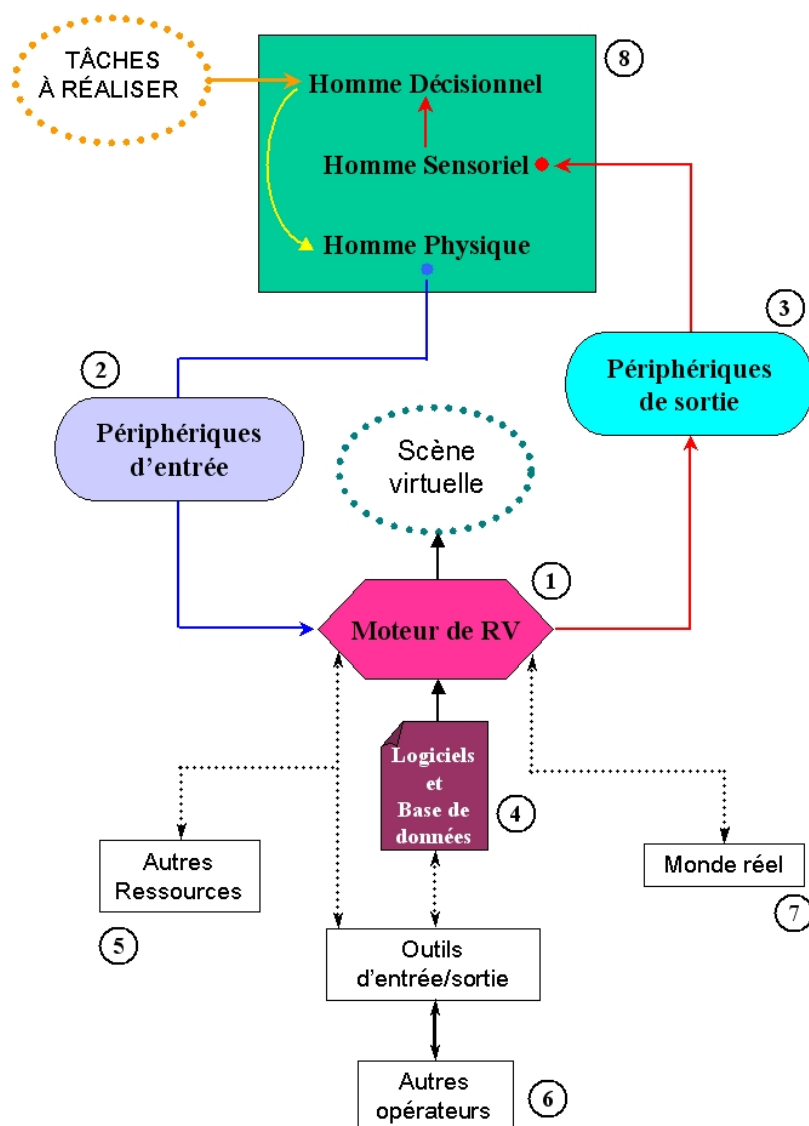


FIG. 1.1 – Schéma d'un système de RV [Coi95]

L'Homme (8) est constitué de 3 parties distinctes. Tout d'abord, la tâche à réaliser va être analysée par l'*homme décisionnel*. Une fois cette analyse effectuée, des commandes sont envoyées à l'*homme physique* qui agit sur le moteur de RV (1) via des périphériques d'entrée (2). Les nouvelles informations qui en découlent sont ressenties par l'*homme sensoriel* grâce aux périphériques de sortie (3) avant d'être transmises pour être converties en une nouvelle décision.

Le moteur de RV a pour charge l'affichage de la scène virtuelle générée grâce aux logiciels et aux bases de données. Il doit surtout faire en sorte que les informations sur la scène virtuelle transmises par les périphériques de sorties soient en accord avec les informations provenant des périphériques d'entrée.

Les parties (1), (2), (3), (4) et (8) constituent le système fermé général. Il peut aussi, selon les cas, être connecté au monde réel, à d'autres opérateurs ou à d'autres ressources.

On remarque également sur ce schéma qu'un système de RV n'est pas qu'un programme informatique. Le choix de périphériques de sorties adaptés aux organes sensoriels de l'homme, et le choix de périphériques d'entrée permettant une action naturelle et aisée sur le monde virtuel, sont également primordiaux.

1.2.4 Interfaces Motrices

Comme nous venons de le voir, l'homme interagit avec la simulation par l'entremise de périphériques spéciaux. Les plus connus d'entre eux sont les *capteurs de localisation* et les *gants de données*.

1.2.4.1 Les capteurs de localisation

Les capteurs de localisation, aussi appelés *traqueurs*, permettent de mesurer en temps réel des positions et des orientations dans l'espace. Utilisés en RV, ils servent généralement à localiser l'utilisateur dans l'environnement virtuel.

En positionnant un capteur de localisation sur la tête de l'utilisateur, on peut calculer l'orientation de son regard. Placé sur sa main, et utilisé conjointement avec un gant de données (ou *data-glove*) pour la mesure de la flexion des doigts, il permet la saisie et la manipulation d'objets de manière naturelle.

Il existe plusieurs types de capteurs de localisation, basés sur des propriétés physiques différentes. Les technologies exploitées pour concevoir ces capteurs sont principalement :

- Les liaisons mécaniques
- L'électromagnétisme [Div87] [Tec93].
- L'acoustique [Log91].
- L'optique [WM93].

Parmi les caractéristiques des capteurs de localisation, les plus intéressantes pour juger de leur performance sont :

- la résolution définie par le mouvement (translation ou rotation) de plus grande amplitude qui n'est pas détecté par le système.
- la précision,
- la fréquence d'échantillonnage : fréquence à laquelle sont effectuées les mesures,
- le temps de latence : temps mis par le système pour détecter un mouvement,
- l'enveloppe de travail : espace dans lequel les performances restent acceptables.

Pour nos expériences, nous avons utilisé les traqueurs électromagnétiques Polhemus Patriot.

Le système Patriot (Polhemus) Le système de capture de localisation Polhemus Patriot (figure 1.2) est constitué de trois éléments principaux qui sont le *boîtier de contrôle* (System Electronics Unit, SEU), la *source* et le *récepteur* (sensor), ainsi que d'une alimentation électrique.



FIG. 1.2 – Le système Polhemus Patriot

- Le *boîtier de contrôle* : Contient le matériel et le logiciel nécessaires pour produire et détecter les champs magnétiques, calculer la position et l'orientation des récepteurs, et gère l'interface avec la station de travail via un port série RS-232 ou USB 1.1.

- La *source* : La source est composée de plusieurs bobines magnétiques qui émettent un champ magnétique lorsqu'elles sont traversées par un courant électrique. La source est le référentiel du système pour les mesures du récepteur.
- Le *récepteur* : Le récepteur est également composé de bobines magnétiques, mais elles servent ici à détecter le champ magnétique émis par la source. La position et l'orientation du récepteur sont alors mesurées par rapport à la source. Le détecteur est complètement passif, aucune tension active n'y étant appliquée.

Il est possible d'ajouter un second récepteur sur le même boîtier. De plus, pour nos expérimentations, nous avons remplacé la source TX2 de base par une source "grand champ" TX4, nous permettant d'accroître de 40% la taille de l'espace de travail, ce qui nous donnait les performances suivantes :

- Résolution : 0.0015 pouce par pouce séparant la source du récepteur (1 pouce = 2.54 cm) ; 0.1° pour l'orientation,
- Précision : 0.1 pouce RMS pour la position en X, Y, ou Z ; 0.75° RMS pour l'orientation du récepteur,
- Fréquence d'échantillonnage : 60 Hz par récepteur,
- Temps de latence : 17 millisecondes (sans filtre logiciel),
- Enveloppe de travail : 7 pieds, soit environ 2.13m.

1.2.4.2 Les gants de données

Sturman a montré que la main pouvait être utilisée comme un dispositif d'entrée informatique sophistiqué, et permettait la réalisation en temps réel de tâches complexes nécessitant plusieurs degrés de liberté [Stu92]. Cette technique d'interaction exige l'utilisation de gants de données ou d'exosquelettes pour mesurer les mouvements de la main, et en particulier ceux des doigts.

Pour nos expérimentations nécessitant la mesure en temps réel des valeurs d'angle articulaires des doigts, nous avons utilisé le *5DT Data Glove 5TM* puis le *5DT Data Glove 5 UltraTM*.

Les 5DT Data Glove : Les 5DT Data Glove conçus par la société Fifth Dimension Technologie sont probablement les gants de données les plus utilisés dans les applications de Réalité Virtuelle [Tec04], et utilisent le même type de technologie que le *DataGlove* imaginé par Zimmerman [ZLB⁺87].

Ce sont des gants en nylon ou en lycra (voir la figure 1.3) dans lesquels sont incorporées, sur le dessus des doigts, des fibres optiques. Chaque flexion d'une articulation provoque une



FIG. 1.3 – Le gant de données 5DT Data Glove Ultra

pliure de la fibre optique du doigt correspondant. A une extrémité de la fibre est installée une diode émettrice tandis qu'à la seconde est monté un phototransistor qui mesure l'intensité de la lumière y parvenant. La flexion d'un doigt, provoquant une déformation de la fibre optique et atténuant l'intensité de lumière reçue par le phototransistor, permet de déduire une estimation de l'angle de la flexion.

Dans sa version de base, celle que nous utilisons, on mesure une valeur de flexion par doigt. La version plus évoluée permet d'avoir deux valeurs de flexion par doigt (mesurées sur la première et la deuxième phalange du doigt), ainsi que l'abduction, c'est à dire l'écartement entre chaque doigt. L'avantage de ce type de gant réside surtout dans sa légèreté, de l'ordre de quelques dizaines de grammes (300g pour la version sans fil, avec les piles).

1.2.5 Interfaces sensorielles et sensori-motrices

Les interfaces sensorielles ont pour but de stimuler les sens de l'utilisateur afin de lui faire ressentir une sensation de présence dans l'environnement virtuel. Le premier et principal sens utilisé pour permettre l'immersion de l'utilisateur est la vue, aussi allons nous détailler les *interfaces visuelles*. Bien que le son spatialisé soit important pour renforcer la sensation de présence [HB96], nous ne traiterons pas ici les différents dispositifs sonores. Nous ne parlerons pas non plus du peu d'interfaces gustatives existantes, et, bien que les recherches sur les interfaces olfactives et sur l'apport de l'odorat en environnement virtuel se développent [TRR06] [RTR06], elles ne font pas l'objet de ce travail. En revanche, nous ferons un point rapide sur les interfaces sensori-motrices qui nous intéressent : les *interfaces haptiques* et notamment le *SPIDAR*.

1.2.5.1 Les interfaces visuelles

Aujourd'hui, les interfaces visuelles se regroupent en deux grandes familles : les *visiocasques* et les *systèmes de projection sur grand écran*.

Les visiocasques : Les visiocasques, ou HMD (Head Mounted Display), développés d'après les travaux de Sutherland [Sut68], ont été très populaires à une époque et sont encore le symbole de la RV pour le grand public. Ce sont des dispositifs que l'utilisateur porte sur la tête et dont le rôle est l'affichage de la scène virtuelle sur des écrans situés devant les yeux (voir figure 1.4). Munis ou couplés à un capteur 3D, il devient possible d'adapter l'affichage de la scène virtuelle en fonction de la direction du regard de l'utilisateur.



FIG. 1.4 – Un Visiocasque : le 5DT HMD 800 de Fifth Dimension Technologies

Sans même parler du poids et de la gêne qu'ils occasionnent, les visiocasques souffrent de la faible résolution des écrans qu'ils emploient, ainsi que de l'étroitesse du champ de vision que leur utilisation entraîne. Cette réduction du champ visuel est très dommageable puisqu'elle modifie la représentation spatiale de l'environnement et affecte les capacités de navigation et de recherche visuelle.

Par ailleurs, la vision de l'utilisateur étant complètement recouverte par l'affichage de la scène virtuelle, celui-ci n'est plus en mesure de voir son corps, ce qui peut être très déconcertant et très gênant dans le cadre d'interactions manuelles.

Les systèmes de projection sur grand écran : Il existe 3 grands types de systèmes de projection sur grand écran : les *murs de projection*, les *Workbenches* et les *salles immersives*.

– **Les Murs de projection :**

Un "mur de projection" (Wall Display) est un simple écran, mono ou stéréoscopique, placé contre un mur. Il est généralement rectangulaire, mais il existe des versions cylindriques destinées à de larges audiences.

– **Les Workbenchs :**

Les Workbenchs ("plan de travail virtuel" ou "bureau immersif") s'apparentent à une table dont la face supérieure serait un écran. On trouve maintenant des workbenchs à deux écrans, le second étant placé verticalement et accolé au premier (voir figure 1.5. Ces dispositifs sont toutefois principalement utilisés pour le travail de maquettes virtuelles et pour la visualisation de données.

– **Les Salles immersives :**

Le concept de "Salle Immersive" s'applique aux dispositifs dans lesquels l'utilisateur est environné par des écrans. La salle immersive la plus connue est sans conteste le CAVE de Cruz-Neira [CNSD93]. Il s'agit d'un environnement cubique à 4 ou 6 écrans sur lesquels sont projetées les images de la scène virtuelle, parfois à l'aide de miroirs.

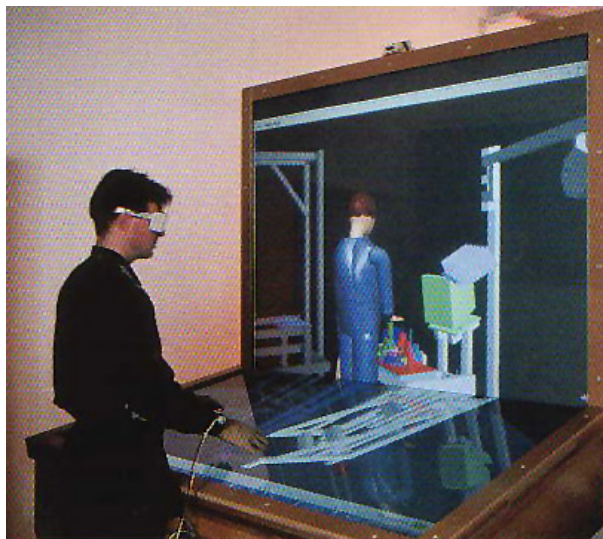


FIG. 1.5 – Un Workbench : le TAN Holobench

La stéréoscopie : La stéréoscopie est l'ensemble des techniques permettant de reproduire une perception de relief à partir de deux images planes. En RV, il existe deux modes de projection stéréoscopique :

– **La Stéréo Active :**

La stéréo active consiste à afficher successivement une image destinée à un œil particulier.

Des lunettes LCD, synchronisées avec le projecteur (généralement grâce à un système infrarouge) permettent de masquer l'œil qui n'est pas censé voir l'image.

– **La Stéréo Passive :**

L'idée de la stéréo passive est au contraire d'afficher en permanence sur un seul écran deux images (une pour chaque œil). Cela nécessite l'usage de deux projecteurs, chacun muni d'un filtre polarisant circulaire. Le port de lunettes munies de verres polarisants permet alors à chaque œil de ne voir que l'image d'un seul des projecteurs.

1.2.5.2 Les interfaces haptiques

Contrairement aux interfaces sensorielles qui ne font que stimuler un organe sensoriel de l'utilisateur, les interfaces sensori-motrices, dont font partie les interfaces haptiques, impliquent un échange d'énergie entre l'utilisateur et le système : elles mesurent une position et restituent un effort qui s'oppose au mouvement de l'utilisateur.

Ainsi, les *gants à retour d'effort* tels que le Cybergrasp (illustré sur la figure 1.6) permettent une manipulation réaliste d'objets en s'opposant à la fermeture des doigts. En revanche, la sensation de poids ne peut être fournie à l'utilisateur, ces systèmes ne pouvant s'opposer au déplacement de la main. Par ailleurs, ces systèmes sont souvent assez lourds ou encombrants, surtout combinés à un exosquelette.



FIG. 1.6 – Un gant haptique : le Cybergrasp de Virtual Technologies

L'autre grand type d'interface haptique est celle qui regroupe les *bras articulés à retour d'effort*. Il en existe de différentes versions, de différentes tailles, impliquant une utilisation desktop (comme le Phantom Omni de Sensable, figure 1.7) ou à grande échelle, et avec un nombre de degrés de libertés variables.



FIG. 1.7 – Un bras articulé à retour d'effort : le Phantom Omni de Sensable

Quoi qu'il en soit, ces périphériques ne permettent pas une interaction naturelle, à moins que la tâche simulée n'emploie, de base, un outil manipulé de manière similaire à l'interface. C'est notamment le cas en chirurgie [MNS98].

Les Props : Plutôt que d'utiliser des interfaces haptiques actives, coûteuses et encombrantes, une technique de restitution de retour haptique consiste à utiliser des "Props", c'est à dire des accessoires. On parle de retour "passif" puisqu'aucune force n'est calculée pour être renvoyée à l'utilisateur. Le Prop idéal est celui qui a la forme et les caractéristiques (poids, déformabilité, rugosité, etc.) de l'outil ou de l'objet manipulé. C'est cette approche qui est utilisée par Hinckley [HPGK94] pour son outil de visualisation de données médicales 3D.

1.2.5.3 Le SPIDAR

Le SPIDAR (SPace Interface Device for Artificial Reality) est un système haptique à base de fils développé par le Professeur Makoto Sato [IS93] [INS94] [IS94]. Il en existe plusieurs versions et plusieurs configurations (desktop, à 1 ou 2 mains, etc.), y compris pour les grands environnements de travail [BIS00].

Afin d'utiliser ce système pour nos travaux de recherche, nous avons mis en place, en collaboration avec l'Institut de Technologie de Tokyo, un SPIDAR à échelle humaine (SPIDAR-H). Cette interface, illustrée sur la figure 1.8, permet de mesurer en temps réel la position des deux mains de l'utilisateur et d'appliquer des forces sur celles-ci dans n'importe quelle direction, et ce dans un espace de travail beaucoup plus important que celui de la plupart des interfaces haptiques standard. Elle permet de restituer des informations relatives au poids et à l'inertie des objets ou de contraindre les mouvements de l'utilisateur (guide haptique) pour l'aider dans la réalisation

d'une tâche sensorimotrice.



FIG. 1.8 – Illustration du SPIDAR

Le SPIDAR-H utilise un contrôleur SH4 de la compagnie japonaise Cyverse. Le dispositif nécessite l'utilisation de 8 moteurs, chacun placé sur un des sommets d'un cube de 2,5 m de côté, au centre duquel évolue l'utilisateur (figure 1.9a). Les moteurs 1-3-6-8 permettent la localisation et l'application de forces sur la main gauche tandis que les moteurs 2-4-5-7 sont utilisés pour la main droite (figure 1.9b). En contrôlant la tension de chaque fil, le système peut générer des forces sur les deux mains.

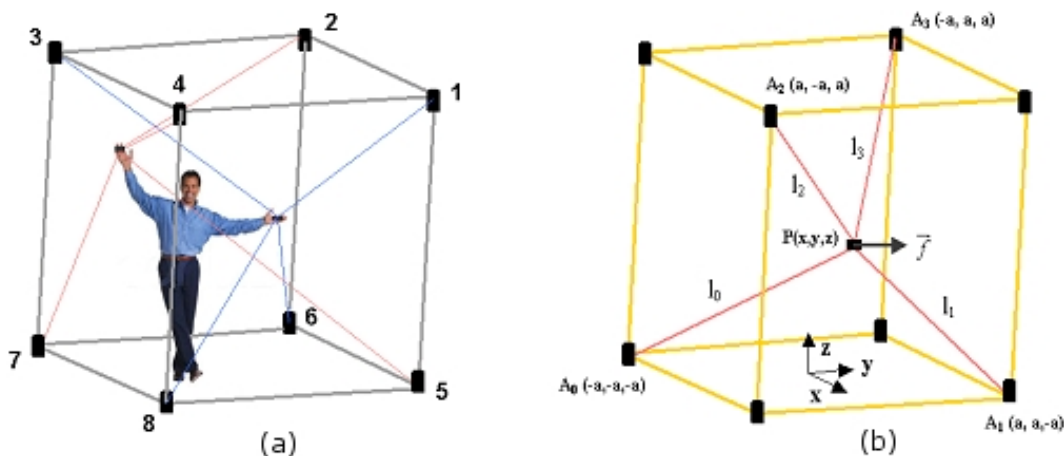


FIG. 1.9 – L'utilisateur dans le cube du SPIDAR (a) ; L'utilisation des fils pour la position et le retour d'effort (b)

Mesure de la position La position de chaque main de l'utilisateur est déterminée par la mesure de la longueur des fils. $P(x,y,z)$ représente la position de la main dans l'espace réel et l_i ($i=0, \dots, 3$) représente la longueur du i -ème fil. $P(x,y,z)$ doit satisfaire les équations (1-4) ci-dessous. On peut ainsi obtenir la position de chaque main réelle avec le système d'équations suivant (5) :

$l_0^2 = (x+a)^2 + (y+a)^2 + (z+a)^2 \quad (1)$ $l_1^2 = (x-a)^2 + (y-a)^2 + (z+a)^2 \quad (2)$ $l_2^2 = (x-a)^2 + (y+a)^2 + (z-a)^2 \quad (3)$ $l_3^2 = (x+a)^2 + (y-a)^2 + (z-a)^2 \quad (4)$	$(5) \quad \begin{cases} x = \frac{l_0^2 - l_1^2 - l_2^2 + l_3^2}{8a} \\ y = \frac{l_0^2 - l_1^2 + l_2^2 - l_3^2}{8a} \\ z = \frac{l_0^2 + l_1^2 - l_2^2 - l_3^2}{8a} \end{cases}$
---	--

L'origine des axes de coordonnées se trouve au centre de l'espace de travail. L'étendue des coordonnées se situe dans l'intervalle $[-1.25m; +1.25m]$. L'erreur de mesure de la position est inférieure à 1.5cm sur tout l'intervalle. La bande passante est d'environ 10KHz.

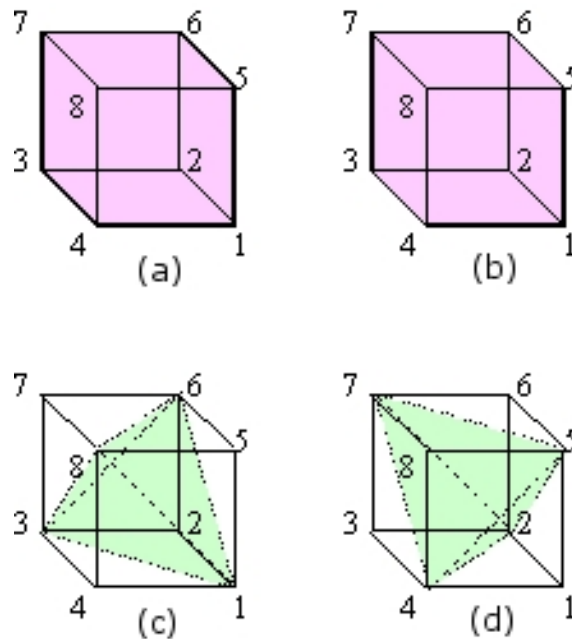


FIG. 1.10 – Enveloppes de travail du SPIDAR

Retour d'effort Le système utilise la force résultant de la tension des fils pour fournir une force. Comme la main est attachée à quatre fils, la force appliquée sur celle-ci est la résultante des forces appliquées par chacun des quatre moteurs. La force maximale pouvant être appliquée dans n'importe quelle direction est de 30N en $P(0,0,0)$. Toutefois, l'amplitude et la direction de

cette force dépend de la position de la main dans l'espace de travail. S'il est vrai que l'enveloppe de travail pour la position de chaque main est de la taille du cube (figure 1.10a pour la main droite, figure 1.10b pour la main gauche), l'espace haptique dans lequel le système peut appliquer un retour d'effort dans n'importe quelle direction se limite à un tétraèdre, visible sur la figure 1.10c pour la main droite, 1.10d pour la main gauche.

Dans l'équation ci-dessous, a_i représente le module de la force appliquée par le moteur correspondant au fil i .

$$\vec{f} = \sum_{i=0}^3 a_i \vec{u}_i \quad (a_i > 0)$$

1.2.6 La Plate-forme PREVISE

La Plate-forme de Réalité Virtuelle pour la Simulation et l'Expérimentation, située à l'Institut des Sciences et Techniques de l'Ingénieur d'Angers, est constituée d'un système de visualisation de 2 x 2,5m en rétro-projection stéréoscopique passive. Elle est pilotée par une station de travail dotée de 2 processeurs Xéon IV cadencés à 3 Ghz, et d'une mémoire vive de 1 Go. En ce qui concerne les périphériques d'action, elle inclut le système électromagnétique de capture de localisation à 6 degrés de liberté Patriot Polhemus, un ensemble de gants de données 5DT Glove de Fifth Dimension Technologies, et le SPIDAR-H que nous venons de décrire. De plus, un système de navigation basé sur le Walking Pad [BECH04] est en finalisation de réalisation. L'utilisation du SPIDAR dans la plate-forme PREVISE permet la conception d'applications diverses, qu'il s'agisse d'attraper de balles [ILRF05] ou de conception de produits [IJRS05].

1.2.7 Techniques d'Interaction

Les techniques d'interaction possibles en Réalité Virtuelle sont communément classifiées en trois types de tâches élémentaires : le Contrôle de l'Application, la Sélection et la Manipulation d'un élément de l'Environnement Virtuel, et la Navigation dans le-dit Environnement [Han97].

1.2.7.1 Contrôle de l'Application

Il s'agit ici d'effectuer l'ensemble des opérations de gestion de l'application, et principalement de régler des paramètres de la simulation.

En ce qui concerne les applications traditionnelles 2D (ordinateur avec écran et clavier-souris), l'interface standard est l'interface WIMP (pour Windows, Icons, Menus, and Pointing). C'est le système classique utilisant les fenêtres, les menus déroulants et les icônes sur lesquels

on clique par l'intermédiaire de la souris.

Dans une application 3D de Réalité Virtuelle, dans laquelle l'utilisateur est immergé et interagit de manière naturelle, l'utilisation du clavier (par l'utilisateur) est impossible, et celle d'un pointeur compliquée.

De nombreux auteurs tels que E. Kruijff [Kru00], J. LaViola [LaV01], G. McMillan et al. [Sal97] ou J. Grosjean et al. [GBCR02] ont tenté de résoudre ce problème en proposant des systèmes adaptés et en classifiant le type de commande employée. La figure 1.11 illustre le système de menus 3D "TULIP" de Bowman et Wingrave [BW01].

Le sujet de notre travail de recherche n'étant clairement pas le contrôle d'application, nous renvoyons donc le lecteur vers les travaux de ces chercheurs pour de plus amples détails.

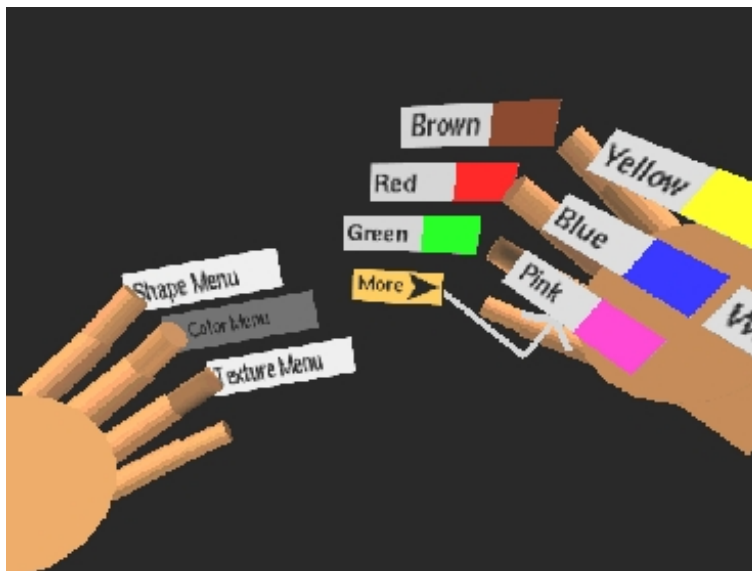


FIG. 1.11 – Capture d'écran du système "TULIP" [BW01]

1.2.7.2 Sélection - Manipulation

Dès son plus jeune âge, l'homme agit sur son environnement à travers la préhension et la manipulation des objets qui l'entourent. Prendre et manipuler un objet avec la main est une tâche particulièrement acquise par le cerveau, ce qui nous permet de l'effectuer sans y penser ou y prêter une attention soutenue. Il n'est donc pas étonnant qu'en RV la manipulation d'objets virtuels soit l'une des interactions principales [BH97].

Le terme de "manipulation" désigne un ensemble de sous-tâches différentes selon les auteurs. Ainsi, pour Foley et al. [FWC84] [FDFH92], cela peut consister à sélectionner une cible, positionner et orienter un objet dans l'espace, ou même saisir une chaîne de caractères. Buxton y inclut le tracé de lignes [Bux90] tandis que Mine propose d'y ajouter les tâches de Scaling, c'est à dire la réduction et l'agrandissement d'objet [Min95].

Toutefois, les deux interactions de manipulation les plus employées en RV restent la sélection d'un objet situé dans l'environnement virtuel, et son déplacement en position et/ou en orientation. Cette dernière tâche nécessite le contrôle de l'objet sur 6 degrés de liberté (3 pour la position sur les axes X, Y, Z ; 3 pour l'orientation : rotations sur les axes X, Y, Z de l'objet). L'utilisation d'une souris 3D permet la gestion simultanée de ces 6 degrés de liberté, mais outre le fait qu'elle ne permet pas une interaction naturelle, elle n'est en plus pas aisée sans une longue période d'apprentissage [Sta98]. Nous lui préférons largement la métaphore de la "Main virtuelle" dont Schmandt fut un des premiers pionniers [Sch83] et qui est employée dans les travaux de Poupyrev [PWBI97] [PWBI98] [PI99] dans son étude de la manipulation à une main dans le cadre de l'utilisation d'un visiocasque et d'un dispositif permettant de connaître la position de la main de l'utilisateur.

Cette métaphore consiste en l'affichage, dans l'environnement virtuel, d'un modèle 3D d'une main. Le déplacement de cette main virtuelle est assujéti au déplacement de la main de l'utilisateur dans le monde réel. Elle reproduit donc les mêmes mouvements que la main de l'utilisateur, de préférence à la même échelle.

De cette manière, la manipulation en environnement virtuel est quasi-identique à celle du monde réel, puisqu'elle est basée sur les schèmes moteurs de préhension et de manipulation d'objets qui nous sont naturels.

La métaphore de la main virtuelle est illustrée par figure 1.12, capture d'écran du Virtual Reality Manipulation Assessment Testbed (VRMAT) de Poupyrev [PWBI97].

1.2.7.3 Navigation

D'après Darken et Sibert [DS93], il existe 3 tâches de navigation :

- L'exploration libre de l'environnement virtuel, sans but ni cible définis,
- La recherche d'une cible dont la position peut être connue ou inconnue. On parle de "recherche amorcée" dans le premier cas et de "recherche naïve" dans le second,
- La manoeuvre en environnement virtuel nécessitant des mouvements courts et précis.

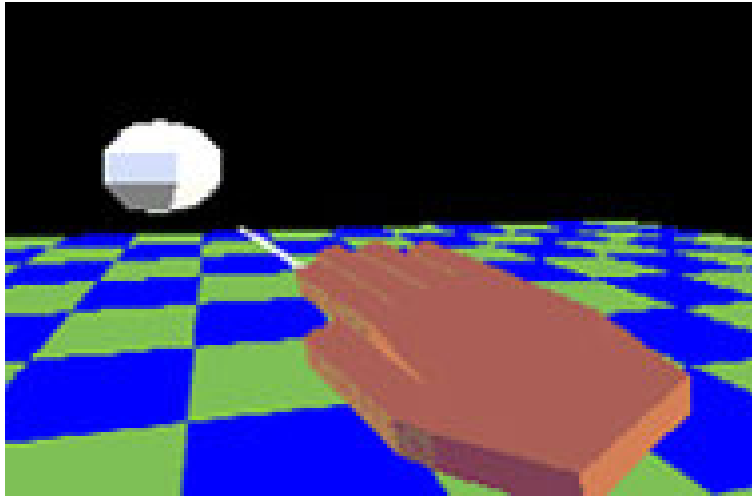


FIG. 1.12 – Capture d'écran du VRMAT [PWBI97] illustrant le concept de "main virtuelle"

De plus, on considère que la tâche de navigation peut être divisée en deux sous-tâches : d'une part la "locomotion" faisant appel aux capacités motrices du sujet, et d'autre part la "recherche d'itinéraire" tirant partie de ses capacités cognitives. Toutefois, pour le sujet navigant en environnement virtuel, ces deux sous-tâches sont traitées conjointement puisqu'il doit continuellement recueillir des informations sur son environnement, les traiter afin de planifier ses prochains mouvements, puis exécuter ceux-ci [BDHB99].

En ce qui concerne la partie "locomotion", on trouve deux grandes catégories de techniques d'interactions : le *déplacement par des moyens réels*, et le *déplacement par des moyens virtuels* [Gro03].

Déplacement par des moyens réels : Le gros avantage de ce type de déplacement est qu'il utilise des techniques de locomotion naturelles, principalement basées sur les mouvements physiques de l'utilisateur.

Ces techniques sont les suivantes :

- **La Marche :**

L'utilisateur se déplace naturellement, en marchant. Si cette technique permet à l'utilisateur de disposer des informations kinesthésiques et vestibulaires qu'elle engendre, elle nécessite surtout un espace de déplacement aussi vaste que l'environnement virtuel. Par ailleurs, il faut pouvoir repérer l'utilisateur dans un tel espace.

- **La Simulation de marche :**

Afin de circonvenir cette exigence spatiale, des techniques basées sur la marche mais restreignant le déplacement de l'utilisateur ont été développées. Citons notamment les

systèmes de Marche sur place [UAW⁺99] [BECH04], les tapis roulant mono ou omnidirectionnels, les tapis munis de capteurs comme le "Magic Carpet" [PAHR97] ou encore les Plate-formes pédestres programmables permettant de marcher sur un terrain infini, non-plat et dans toutes les directions [IYN01].

– **Le Poste de conduite :**

Il s'agit d'utiliser la métaphore d'une conduite de véhicule. Il peut s'agir d'une automobile (on utilise alors un volant, avec ou sans retour d'effort, couplé ou non à un pédalier), d'un avion (utilisation d'un manche à balai, ou joystick), ou même d'une bicyclette [BMH98].

Déplacement par des moyens virtuels : A l'inverse, les techniques de déplacement par des moyens virtuels nécessitent souvent peu de moyens matériels et utilisent des techniques d'interaction plus artificielles.

Ces principales techniques sont les suivantes :

– **Conduite :**

Pour cette technique, l'utilisateur contrôle la direction du mouvement de manière continue. Il peut s'agir d'un système d'analyse du regard ou de la tête [BDA99], ou bien d'un système de pointage, avec le doigt ou un dispositif spécial [OZ00].

– **Sélection de destination :**

Puisque le but de la navigation est de se rendre d'un point à un autre, cette technique propose de sélectionner, d'une manière ou d'une autre, la destination désirée afin d'y déplacer automatiquement le point de vue.

– **Manipulation de Caméra :**

Cette technique consiste à manipuler le point de vue, soit directement comme si l'utilisateur le tenait dans la main (métaphore "eyeball-in-hand" selon Ware [War90]), soit plutôt indirectement, par exemple en "agrippant l'air" ("grab the air") et en tirant le monde vers soi.

1.2.7.4 Interaction Dynamique

L'interaction dynamique, c'est à dire avec des objets en mouvements ou à mettre en mouvement, n'est clairement pas représentée dans les trois tâches élémentaires habituellement reconnues.

En effet, la tâche la plus proche, qui serait celle de "Sélection-Manipulation", s'intéresse plutôt à la préhension, puis au positionnement et à l'orientation d'un objet fixe de l'environnement

virtuel, tandis que le sujet qui nous intéresse traite des interactions possibles avec un objet en mouvement ou auquel un mouvement doit être transféré.

Le déplacement de l'utilisateur peut être nécessaire pour la réalisation de ces interactions, mais en aucun cas l'interaction dynamique ne saurait se résumer à une seule tâche de "Navigation". Toutefois, bien que l'interaction dynamique ne soit pas actuellement considérée comme l'une des tâches élémentaires de la réalité virtuelle, des travaux de recherche mettant en oeuvre des objets en mouvement ont été réalisés.

1.3 Travaux sur les Objets en Mouvement

Le développement de simulateurs de Réalité Virtuelle comportant une interaction avec des objets en mouvement ou à mettre en mouvement est généralement lié à l'un des trois types de recherche suivants :

- **Défi technique :**

Malheureusement, il s'agit là de la catégorie regroupant la plupart des simulateurs. En effet, beaucoup d'applications sont abandonnées une fois que leur développement est achevé. Aucune recherche utilisant le simulateur comme base n'est menée, si ce n'est pour prouver l'utilisabilité du système, et les articles le mentionnant ne traitent que de la partie technique et des solutions apportées aux difficultés technologiques. C'est par exemple le cas de l'environnement collaboratif en réseau conçu par Molet et al. pour simuler une partie de tennis [MAN⁺99]. Dans le même ordre d'idée, Pan a développé un simulateur de bowling basé sur l'analyse vidéo d'un lancer de boule réelle sur une piste raccourcie, mais sa recherche sur ce sujet s'arrête là [PXH⁺03]. On peut également mentionner le simulateur de lancer de balle de baseball Kyukon [IKD⁺05] adapté en lancer de boule de pétanque pour le Laval Virtual 2006. La pétanque avait d'ailleurs déjà été abordée lors d'un projet mené par des étudiants de l'École Polytechnique Fédérale de Lausanne [CS04]. Enfin, un simulateur de billard faisant usage d'un bras articulé de type Phantom a dernièrement été développé [TAT⁺06].

Bien qu'aucune recherche n'ait été conduite spécifiquement sur l'interaction dynamique, il faut néanmoins préciser que tous ces simulateurs fournissent des solutions technologiques très intéressantes pour la conception technique d'environnements virtuels dynamiques.

- **Recherches sur le réel :**

Dans ce type de recherches, la réalité virtuelle est utilisée comme outil permettant de résoudre une difficulté de mise en oeuvre rencontrée dans le réel. C'est notamment le cas pour le simulateur de football développé par Craig [CBR⁺06]. Inspirée par un coup-franc

magistralement tiré par Roberto Carlos pour le Brésil en 1997, Craig a montré, à l'aide de tirs reproduits en réalité virtuelle car compliqués à réaliser en réel, que même des joueurs professionnels étaient incapables de prédire l'effet qu'aura une rotation très élevée de la balle (plus de 600 tours par minute) sur sa trajectoire .

– **Recherches sur le virtuel :**

Dans cette dernière catégorie de recherches, les simulateurs complexes développés et validés sont ensuite ré-utilisés afin de conduire une série d'expérimentations permettant d'aboutir à des résultats sur le comportement humain en environnement dynamique. Nous pouvons ainsi souligner les travaux effectués à l'IRISA de Rennes par Bideau et al., qui ont tout d'abord conçu et évalué un simulateur de handball [BFM⁺03], puis qui se sont appuyés dessus pour analyser la performance d'un gardien de but réel face à des adversaires contrôlés et animés par le système de RV [BKM⁺03] [BMK⁺04].

Lorsque le simulateur développé est relativement simple, il ne fait pas l'objet d'une publication préalable. Il est plutôt rapidement décrit afin d'explicitier le protocole expérimental utilisé durant l'étude, comme par exemple pour les travaux de recherche de Zaal sur l'estimation de chute d'une balle dans un CAVE [ZM03] ou ceux de Siegler portant sur le contrôle du rebond en situation de jonglerie virtuelle [SMWB03].

1.3.1 Inconvénients et Avantages de la RV

L'utilisation des techniques de réalité virtuelle dans le cadre de simulation d'environnement dynamique n'est pas sans apporter quelques problèmes qui ne se retrouvent pas en conditions réelles. Elle amène néanmoins des points positifs, qu'il faut savoir employer et déployer pour contrebalancer ses inconvénients.

1.3.1.1 Inconvénients

Le principal problème est la perte, pour l'utilisateur, de beaucoup d'informations sensorielles normalement disponibles dans le réel. En effet, lorsque l'on tente de recréer en réalité virtuelle une situation réelle, on utilise un modèle de la réalité qui ne peut être qu'une approximation de celle-ci, ce qui aboutit pour l'utilisateur placé dans la simulation à une dégradation, voire une perte, des informations qu'il possède par rapport à la même situation en réel. Citons notamment :

– **Perception de la profondeur :**

Quel que soit le système de visualisation choisi à l'heure actuelle, l'accommodation des

yeux se fait presque toujours sur le plan d'affichage (écrans devant les yeux pour un visiocasque, grand écran pour les murs de projection) alors qu'en réel elle se fait sur l'objet et s'adapte selon le déplacement de ce dernier. Les mouvements oculaires mis en jeu lors de cette accommodation permettent une estimation de la distance de l'objet.

– **Décalages ("offset") spatio-temporels :**

Lors de l'utilisation de murs de projection comme système de visualisation, la main virtuelle de l'utilisateur, par exemple, est perçue par celui-ci sur le plan d'affichage alors que sa main réelle est proche de lui (offset spatial). Il peut également y avoir un délai entre le moment où l'utilisateur effectue un mouvement et celui où il est détecté par les interfaces. Il en résulte un décalage (ou "offset") temporel entre l'action de l'utilisateur et sa représentation dans l'environnement virtuel. Ceci rejoint les préoccupations sur l'étude des systèmes à retard.

– **Retours tactilo-proprio-kinesthésiques :**

Dans une simulation de RV, l'utilisateur perd généralement les informations relatives à l'objet : son poids, sa déformabilité, sa rugosité, sa température... Il perd également, lorsque des déplacements sont simulés, les informations vestibulaires issues de l'oreille interne qui lui sont normalement fournies lorsqu'il est en mouvement.

Un autre inconvénient important des techniques de réalité virtuelle est lié à ses interfaces. Elles sont généralement intrusives pour l'utilisateur : il doit mettre un casque, enfiler un gant, porter des capteurs, manipuler un bras articulé, etc. Un système de caméra et de reconnaissance d'image pourrait régler ce problème, mais à l'heure actuelle seules des interactions très basiques sont possibles sans le port de marqueurs spéciaux.

1.3.1.2 Avantages

L'un des intérêts de la RV est que les conditions de réalisation d'une tâche peuvent être définies et fixées, apportant un caractère répétable à celle-ci. C'est une chose généralement impossible dans le réel, surtout lorsqu'il s'agit de tâches en extérieur : on ne maîtrise pas les conditions météorologiques (ensoleillement, vitesse et direction du vent, agitation de la mer...). On peut ainsi, en Réalité Virtuelle, s'entraîner à une tâche dans des conditions particulières sans avoir à attendre qu'elles ne se produisent en réel. Dans le même souci de répétabilité de la tâche, la RV peut permettre de simuler des tâches trop coûteuses en réel, comme le tir au lance-missiles sur des chars en mouvement.

Un autre grand intérêt de simuler des tâches en RV plutôt que de les effectuer en réel est que cela anéantit les risques encourus en cas d'activité dangereuse, telle que la maintenance d'un

réacteur nucléaire. Par ailleurs, il y a également un avantage logistique puisque, si l'on reprend l'exemple précédent, la centrale n'a pas à être arrêtée le temps de la formation.

Enfin, si l'utilisation de la RV engendre une perte d'informations disponibles dans le réel, elle permet également de fournir des données que l'utilisateur ne peut percevoir normalement. Ainsi, on peut d'une part faire de la substitution sensorielle pour corriger les défauts inhérents à l'utilisation du système, en affichant par exemple des flèches proportionnelles à la pression exercée par les doigts de l'utilisateur si celui-ci n'a pas d'interface haptique à sa disposition (on remplace un sens par un autre, ici l'haptique par la vue), et on peut d'autre part faire de "l'augmentation sensorielle", en affichant ou en faisant ressentir des informations imperceptibles en réel, comme par exemple la trajectoire complète d'un objet qui vient d'être lancé. Ce dernier exemple se base sur la connaissance d'un résultat *a priori* que l'on peut prédire grâce au savoir que l'on a du comportement physique d'un objet en fonction de ses paramètres initiaux (position, vitesse, etc.). Cette possibilité offerte par la réalité virtuelle de connaître le résultat d'une tâche dynamique avant même qu'elle ne soit terminée permet, dans certains cas, de choisir une approche centrée sur le résultat du geste plutôt que sur le geste en lui-même (voir le cas du tir à l'arc [IRFB05] dans le chapitre 4). Cette approche passe par la création d'aides logicielles allant dans ce sens.

Mais avant cela, commençons par nous intéresser aux aides logicielles existantes.

1.4 Aides Logicielles

Nous avons recherché l'existence d'une structure d'aides logicielles qui serait adaptée à nos besoins, c'est à dire utilisable dans le cadre d'interactions avec un objet en mouvement. Nous avons trouvé dans la littérature de nombreuses références aux "Aides Logicielles Comportementales," et quelques unes concernant les "Fixtures Virtuelles". Nous avons également dirigé nos investigations vers le domaine du jeu vidéo.

1.4.1 Aides Logicielles Comportementales

Philippe Fuchs définit une Aide Logicielle Comportementale comme une aide programmée censée "*faciliter l'utilisation matérielle ou mentale des interfaces comportementales*". Une Interface Comportementale est définie comme un "*dispositif matériel qui exploite un comportement humain, naturel et qui n'implique pas une période d'apprentissage importante et complexe*

pour pouvoir être utilisé" [FMP03].

Il s'agit donc "d'astuces" logicielles devant permettre à l'utilisateur d'employer efficacement les périphériques mis à sa disposition. Fuchs précise que *"les ALC peuvent être déterminées en fonction des "affordances" des objets ou de l'environnement autour du sujet"*. Les affordances, selon Gibson [Gib77], sont une caractérisation des interactions possibles entre un organisme et un objet, en fonction des capacités sensori-motrices et du mode de vie du-dit organisme.

Deux exemples d'ALC mises en oeuvre par Fuchs permettant d'illustrer leur utilisation :

- **ALC de manipulation :**

Si, dans une simulation, des objets manipulables sont rapprochés par l'utilisateur d'une table (caractérisée par l'affordance "on peut y poser des objets"), alors ceux-ci sont automatiquement plaqués dessus et ne possèdent plus que 3 degrés de liberté.

- **ALC de motricité :**

Dans une simulation de maintenance de système de commandes d'aiguillage situés le long d'une voie ferrée, lorsque l'intention de l'utilisateur de se rapprocher du téléphone ou de la commande d'aiguillage a été détectée, le logiciel prend le contrôle du déplacement virtuel de l'utilisateur afin de l'amener automatiquement à l'endroit désiré.

1.4.2 Fixtures Virtuelles

On appelle "Fixtures Virtuelles" les entités graphiques et haptiques 3D développées dans le but d'aider à la réalisation de tâches de télé-opération de robot. Le concept de Fixtures Virtuelles a été défini dans les travaux de [Ros93] et [SP94]. Rosenberg illustre ce concept avec l'exemple suivant :

When asking one to draw a straight line segment in the real world, human performance can be greatly enhanced by using a simple tool such as a ruler. The use of a ruler reduces the amount of mental processing required to perform the task, and increases operator's performance and ability during the drawing process. Most of all this allows one to draw a correct line segment that if no ruler had been used.

que l'on peut traduire par :

Quand vous demandez à quelqu'un de tracer un segment de droite dans le monde réel, la réalisation de cette tâche peut être grandement améliorée par l'utilisation d'un outil simple tel qu'une règle. L'utilisation de la règle réduit la charge mentale nécessaire à l'effectuation de la tâche, et améliore la dextérité de l'opérateur durant le tracé du trait. Elle permet surtout de tracer un segment de droite plus

correct qu'il ne le serait sans utiliser de règle.

Une fixture sert donc à faciliter une coordination sensori-motrice difficile, par exemple en guidant un déplacement dans un espace virtuel 3D. Pour Otmane et al. [OCMH00], il s'agit d'objets graphiques élémentaires, comme un disque, une sphère, un cône, un plan, un cylindre, un cube ou un carré, pouvant être combinés entre eux, et ayant une capacité attractive ou répulsive envers l'opérateur. Morineau [Mor02] illustre l'exemple d'un cône servant à guider automatiquement le déplacement d'un agent vers l'espace disponible entre deux murs virtuels avec la figure 1.13.

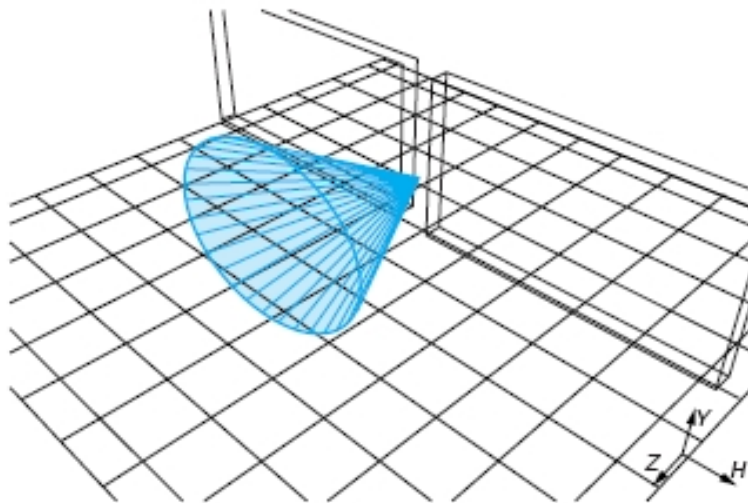


FIG. 1.13 – Exemple de Fixture [Mor02]

Otmane et al. proposent la définition formelle d'une fixture illustrée par le tableau 1.1.

1.4.3 Aides classiques des Jeux Videos

Les ordinateurs et consoles grand public des années 80 ne disposaient pas de performances très évoluées, si l'on compare avec ce qu'il se fait de nos jours. Les jeux de l'époque étaient principalement réalisés en 2D, ou en "fausse 3D", c'est à dire en 2D avec une vue isométrique donnant un effet de perspective, les machines étant incapables de gérer un environnement en "vraie 3D". Il était donc, encore plus qu'aujourd'hui, très difficile d'estimer les distances en profondeur.

Par ailleurs, le retour d'informations sensorielles dont disposait le joueur était limité à la

Nom	Identifie la fixture
Type	Simple ou complexe et active ou passive. Si complexe, elle contient les différents liens des fixtures combinées
Référentiel	Contient la position et l'orientation de la fixture sur les axes X, Y, Z
Attachement	Statique ou dynamique et contient les coordonnées du point virtuel ou de l'objet
Zone d'effet	Contient l'équation de la forme volumétrique, des surfaces ou toute autre forme géométrique pouvant être associée à la fixture virtuelle
Pré-condition	Contient la condition d'activation de la fixture
Fonction	Contient un ensemble d'actions à effectuer dans le guidage virtuel soit par le robot, l'opérateur humain ou les deux à la fois
Post-condition	Contient la condition d'inactivation de la fixture

TAB. 1.1 – Structure d'une fixture virtuelle, d'après Otmane, Colle, Mallem et Hoppenot.

vision et à l'ouïe, dans les deux cas avec une qualité assez faible. De plus, les techniques d'interaction du joueur se cantonnaient principalement à l'utilisation d'une croix directionnelle et d'un ou deux boutons d'action maximum (même si certains jeux sur ordinateur faisaient appel aux touches du clavier).

On a donc rapidement assisté à l'introduction d'aides devenues communes dans le domaine du jeu vidéo. Ainsi, le jeu de golf "Leaderboard - Pro Golf Simulator", développé sur Atari ST par Access Software Inc fournissait-il dès 1986 un ensemble d'informations numériques (la distance entre la balle et le trou, "YARDS 350" sur la capture d'écran) et vectorielle (la direction et l'amplitude du vent, la ligne bleue sous l'indication "WINDS") sur l'environnement virtuel (figure 1.14).

Pour sa part, le jeu de billard "Side Pocket" illustré par la figure 1.15 et développé par Data East pour la console NES de Nintendo, est l'un des tout premiers jeux du genre à proposer l'affichage de la trajectoire de la bille blanche, même s'il ne tenait pas compte des éventuelles collisions.

Enfin, il est devenu classique dans les jeux de tennis d'indiquer par une marque au sol la position à adopter sur le terrain pour réaliser un smash lorsque l'adversaire tente un lob. Cette aide au positionnement est par exemple utilisée dans "Pro Tennis WTA Tour" sur Gameboy Advance (Konami, 2002) dont une capture d'écran est représentée par la figure 1.16.



FIG. 1.14 – Capture d'écran du jeu "Leaderboard" sur Atari ST



FIG. 1.15 – Capture d'écran du jeu "Side Pocket" sur NES



FIG. 1.16 – Capture d'écran du jeu "Pro Tennis WTA Tour" sur Gameboy Advance

1.4.4 Aides pour l'Interaction Dynamique

A notre connaissance, il n'existe aucune structure d'aides logicielles spécifiques aux interactions utilisateur-objet en mouvement. Par ailleurs, les aides existantes ne sont pas à même de nous satisfaire, pour les raisons que nous allons détailler maintenant.

- **Aides Logicielles Comportementales :**

Les Aides Logicielles Comportementales ne correspondent pas au type d'interaction qui nous intéresse. Elles ont pour but de simplifier certaines tâches, en restreignant les mouvements de l'utilisateur, ou en les automatisant sitôt l'intention de ce dernier perçue. Dans une simulation d'attraper de balle, une telle ALC pourrait permettre de coller automatiquement la balle à la main dès que l'utilisateur amorce un geste dirigé vers celle-ci. Ce n'est pas du tout ce genre d'aide que nous recherchons car elles réduiraient le réalisme de la tâche d'interaction dynamique.

- **Fixtures Virtuelles :**

Les Fixtures Virtuelles peuvent être intéressantes pour certains cas précis, et à condition de disposer d'équipement à retour d'effort, mais leur concept reste néanmoins trop orienté vers la télé-opération de robot. L'idée d'un guide haptique reste toutefois à retenir.

- **Aides issues des Jeux-videos :**

Finalement, ce sont les aides utilisées dans les jeux videos qui sont pour nous les plus intéressantes, ce qui est assez normal puisqu'elles concernent des objets en mouvement (tennis) ou à mettre en mouvement (billard, golf). Elles consistent à afficher des indications et des repères visuels dans l'environnement virtuel, et l'utilisateur reste libre de ses

"mouvements".

Au regard des systèmes d'aides logicielles existants, on se rend compte qu'aucun n'est adéquat pour les interactions dynamiques. Nous pensons donc qu'il peut être intéressant de se pencher sur la conception d'aides logicielles spécifiques à l'interaction dynamique, notamment en adaptant certaines aides devenues classiques dans le domaine des jeux vidéo.

1.5 Conclusion

Dans ce premier chapitre, nous avons essayé de situer la place des environnements dynamiques dans le domaine de la Réalité Virtuelle. Nous avons remarqué que, s'il existait des interfaces relativement adaptées pour interagir avec un objet en mouvement ou à mettre en mouvement, les interactions reconnues en réalité virtuelle, que sont le Contrôle d'application, la Sélection-manipulation d'objets et la Navigation en environnement virtuel, ne recouvraient pas les possibilités offertes par l'interaction dynamique. Il existe néanmoins des applications de RV utilisant un environnement dynamique, mais pour la plupart, hormis pour le défi technique que représente leur conception, aucun travail de recherche n'est ensuite conduit dessus : elles restent à l'état de démonstrations technologiques ludiques. Dans le cas où des recherches sont effectivement menées, elles traitent d'un problème précis relatif à la simulation, et n'abordent pas le sujet de l'interaction dynamique dans son ensemble. Enfin, bien que les aides logicielles soient couramment utilisées en RV pour améliorer ou enrichir l'interaction, il n'en existe aucune structure adaptée à l'interaction dynamique.

Nous pensons donc qu'il est nécessaire de commencer à se pencher sur les interactions dynamiques possibles, de développer des outils d'aide à l'analyse de l'interaction dynamique en général, et de concevoir une structure d'aides logicielles spécifique à ce type d'interaction. C'est ce travail que nous proposons et décrivons dans le chapitre 2.

Chapitre 2

Analyse de l'interaction dynamique

2.1 Introduction

Par interaction dynamique, nous englobons ici les interactions entre l'Homme et un objet qui est en mouvement, ou que l'Homme doit mettre en mouvement ; ceci par opposition à une saisie/manipulation d'objets statiques, à la navigation dans un environnement plus ou moins figé, et au contrôle d'application via un menu ou une interface quelconque.

L'objet ne doit pas être d'un volume trop conséquent et ne doit pas posséder une vitesse trop importante, sans quoi l'unique interaction entre les deux entités se résume à une collision ou à une tentative d'évitement de la collision par l'Homme. On suppose donc qu'il s'agit d'un objet manipulable à une ou deux mains par l'Homme, ou que l'Homme possède l'outil approprié pour interagir avec l'objet en question.

Dans ce chapitre, nous commencerons par faire une analyse des interactions possibles avec un objet en mouvement, dont nous déduirons une modélisation pour chacune d'entre elles. Nous proposerons alors une structure d'aides logicielles spécifiques ayant pour but de faciliter l'apprentissage et/ou la réalisation de tâches d'interaction dynamique en environnement virtuel.

2.2 Primitives d'Interaction Dynamique

Naturellement, quand on nous lance un objet de petite taille à une vitesse raisonnable, le premier réflexe est de l'attraper. On peut alors à notre tour lancer l'objet vers quelqu'un, son propriétaire d'origine par exemple. Dans le cas d'activités sportives en revanche, il n'est pas rare de vouloir frapper directement l'objet en plein vol, plutôt que de le relancer après l'avoir

préalablement stoppé.

Ces actions d'"Attraper", de "Lancer" et de "Frapper" définissent selon nous ce que nous appellerons dorénavant des "Primitive d'Interaction Dynamique". Ce sont les interactions les plus élémentaires concernant un ou plusieurs objets en mouvement ou à mettre en mouvement. Ainsi, quelle que soit l'interaction dynamique envisagée avec un objet, il sera toujours possible de la réduire à une simple Primitive d'Interaction Dynamique (PID), ou de la décomposer en une suite de PID.

Voyons plus en détail ce que nous entendons par "Lancer", "Attraper" et "Frapper".

- **Lancer** : lancer un objet consiste à le propulser dans une direction donnée avec une vitesse non nulle. Cette action est en règle générale dirigée vers une cible à atteindre, l'objet devant toucher, traverser, ou s'arrêter sur la cible ou à proximité. La manière la plus naturelle pour l'homme de lancer un objet est de le tenir en main, d'effectuer un mouvement permettant de transférer une vitesse à l'objet, puis de le lâcher. C'est un mouvement continu durant lequel l'objet est assujéti à l'utilisateur. Toutefois, il est possible de propulser l'objet de manière discrète, comme dans le cas d'un tir avec une arme à feu.
- **Attraper** : généralement, un objet devant être attrapé ne se propulse pas de lui-même vers l'utilisateur, sauf si l'on considère un animal à attraper. Il était assujéti à une autre entité, et soit il s'en est simplement détaché auquel cas l'objet est simplement soumis aux forces extérieures (gravité, vent, etc.), soit l'entité a propulsé l'objet dans une direction, généralement proche de celle de l'utilisateur. Dans une simulation de Réalité Virtuelle, on peut toutefois s'affranchir de ce type de situations. Ainsi, il est possible d'imaginer un objet statique, "flottant" à une position donnée avec une vitesse nulle et non soumis aux forces extérieures, qui se trouverait soudainement propulsé dans la direction de l'utilisateur avec une vitesse initiale non nulle, et soumis aux forces extérieures. Cela revient à imaginer un "porteur" de l'objet invisible.

NB : La saisie d'Objets statiques n'est pas considérée comme une Interaction Dynamique. En effet, cette action se rapporte à une simple Manipulation telle qu'elle a déjà été étudiée et définie par Bowman [BH97].

- **Frapper** : frapper un objet en vol est une action qui se situe à la limite de l'*attraper*, puisqu'il y a un rendez-vous entre l'utilisateur et l'objet, et du *lancer*, car l'objet est généralement renvoyé dans la direction d'une cible afin qu'il la touche, la traverse ou s'arrête dessus, ou à proximité. Le geste permettant de renvoyer un objet vers une cible peut être

soit discret, c'est-à-dire que la surface entrant en collision avec l'objet en mouvement lors du rendez-vous est fixe, et simplement orientée dans la bonne direction, l'objet ne faisant alors que "rebondir" dessus ; soit il est continu et assez analogue au geste du lancer, sauf que c'est le contact au moment du rendez-vous qui donne à l'objet sa nouvelle direction. NB : Le "Frapper" tel que nous le définissons ici concerne un objet déjà en mouvement. Frapper un objet immobile, comme au golf par exemple, c'est mettre cet objet en mouvement dans une direction donnée, et c'est donc une action couverte par notre définition du "Lancer".

Dans toutes ces actions, il n'est pas exclu que l'Homme, la cible ou le porteur de l'objet puissent être en mouvement. Ces déplacements accroissent la difficulté de l'interaction car ils apportent des facteurs supplémentaires à prendre en compte lors de la réalisation de la tâche (vitesse et accélération de l'Homme, de la cible ou du porteur de l'objet).

Un autre facteur important à relever intervenant dans ces interactions est la relativement faible durée de la tâche. Hormis le temps que peut prendre une visée avec une arme à feu, on peut raisonnablement admettre que l'action ne prendra pas plus d'une dizaine de secondes à être effectuée. Du fait de cette durée d'action très réduite, l'interaction dynamique fait énormément appel au sens de l'anticipation de l'homme, à sa perception en un temps très court des distances et des vitesses, ainsi qu'à sa connaissance du comportement physique des objets en fonction des forces extérieures (gravité, vent).

2.2.1 Modélisation des Primitives d'Interaction Dynamique

Avant de modéliser les Primitives d'Interaction Dynamique, il convient de s'intéresser tout d'abord aux objets dynamiques qu'elles manipulent.

2.2.1.1 Objets Dynamiques

Nous considérons qu'un objet peut être déclaré dynamique s'il est en mouvement et qu'on peut interagir avec lui, ou qu'il est statique mais qu'il peut être mis en mouvement par une action physique de l'homme ou un événement logiciel. On se rend tout de suite compte que, puisque l'objet est ou sera en mouvement, la position de l'objet au début de la simulation ne sera probablement pas la même qu'à la fin. L'objet va donc passer par plusieurs phases, durant lesquelles il sera par exemple à l'arrêt à une certaine position de départ, puis en mouvement, et enfin à l'arrêt à une position finale dont il ne bougera plus.

On voit bien là qu'il y a au moins deux "états" très importants pour un objet dynamique : son État Initial et son État Final. Dans certains cas, ses États intermédiaires peuvent également avoir leur intérêt, si la trajectoire utilisée est importante par exemple (voir figure 2.1).

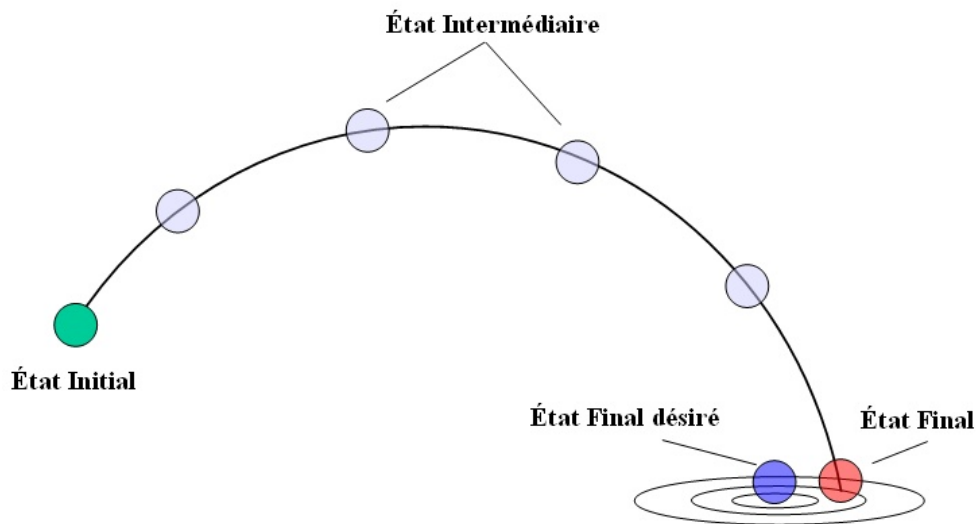


FIG. 2.1 – Les différents États de la balle lors d'un Lancer.

État d'un Objet : On définit l'État d'un objet dynamique par ses informations suivantes :

- **P** : Position de l'objet
- **O** : Orientation de l'objet

- $\dot{P}$ ou V_l : Vitesse linéaire de l'objet
- $\ddot{P}$ ou A_l : Accélération linéaire de l'objet

- $\dot{O}$ ou V_a : Vitesse angulaire de l'objet
- $\ddot{O}$ ou A_a : Accélération angulaire de l'objet

La Position, l'Orientation, la Vitesse et l'Accélération sont les données de base permettant de connaître la situation d'un objet à un instant t . Elles sont essentielles également pour calculer la situation de l'objet à un instant $t+1$, même si elles sont souvent insuffisantes.

Selon l'objet que l'on définit, d'autres données variables peuvent être incorporées. Ce sont des caractéristiques propres à l'objet qui n'interviennent pas forcément dans sa dynamique,

mais qui peuvent influencer sur sa perception ou qui changent fondamentalement sa situation (voir la modélisation de la Main Virtuelle ci-dessous).

L'Objet possède également un certain nombre de Caractéristiques, dont nous allons détailler celles qui nous apparaissent comme étant les plus intéressantes.

Caractéristiques Physiques : Les Caractéristiques physiques sont des données qui vont être prises en compte par la boucle physique de la simulation, afin de déterminer le prochain État de l'Objet (donc Position, Orientation, Vitesse, Accélération) en fonction des forces appliquées et des conditions environnementales.

On peut citer par exemple :

- Forme
- Position du centre de gravité
- Masse
- Amortissement linéaire
- Amortissement angulaire
- Coefficient de restitution

Si l'objet est physiquement déformable, les données liées à cette propriété seront à prendre en compte.

Caractéristiques Graphiques : Les Caractéristiques Graphiques concernent l'apparence visuelle de l'objet, et de ce fait n'interviennent pas dans sa dynamique (si l'on omet le fait que la forme physique de l'objet puisse être déduite de son maillage). En revanche, elles ont un rôle important sur sa perception par l'utilisateur.

Voici une liste de ces caractéristiques :

- Maillage (Mesh) : la forme de l'objet, définie généralement par un ensemble de faces triangulaires.
- Échelle (Scale) : échelle, taille de l'objet. Elle peut être fixe ou évolutive.
- Couleurs : la ou les couleurs de l'objet, définies selon les composantes Rouge, Vert Bleu et Alpha pour la transparence.
- Matériaux : les couleurs de diffusion, ambiante, d'émission, spéculaire, et la valeur de brillance, permettant de définir la couleur de l'objet en réponse à l'éclairage de la scène.
- Textures : images plaquées sur l'objet pour lui donner un aspect.
- Keyframes : les keyframes sont des "photographies" de la forme du Mesh. En interpolant

les positions des triangles composants le mesh d'une keyframe à un autre, on génère une animation. C'est le même principe que l'animation de dessins animés.

- Squelette (Bones) : le squelette du mesh. Lorsqu'un os du squelette subit une transformation (une rotation par rapport à une jointure), la partie du mesh qui est attachée à l'os subit également la transformation. Cela permet d'animer un mesh en effectuant de simples opérations sur les jointures du squelette.

Caractéristiques Sonores : Ces caractéristiques concernent le son ou les vibrations sonores qui peuvent être émises par l'objet. Elles n'ont aucune influence sur la dynamique de l'objet, mais elles peuvent aider l'utilisateur à situer l'objet dans l'espace et l'influencer dans la perception qu'il a de ce dernier. On retrouve ainsi comme caractéristiques : Son, Fréquence, Volume.

Caractéristiques Haptiques : Ces caractéristiques ne sont utiles que si l'environnement virtuel comporte une interface haptique. Compte tenu de la diversité des interfaces haptiques, on peut dire que ces caractéristiques serviront pour la gestion du retour de sensations haptiques à l'utilisateur, qu'il s'agisse d'informations tactiles ou kinesthésiques. Sans être exhaustif, citons tout de même :

- Poids
- Dureté
- Friction Statique
- Friction Dynamique
- Amortissement
- Température
- Rugosité

Caractéristiques Olfactives : A l'instar des caractéristiques haptiques, celles ayant trait à l'olfaction ne valent que si un système de diffusion d'odeur est inclus dans l'environnement virtuel. Les données qu'elles comportent seront donc utilisées pour gérer la diffusion de l'odeur de l'objet.

- Odeur
- Intensité de l'odeur (on peut mesurer l'intensité d'une odeur à l'aide d'un olfactomètre)
- Persistance de l'odeur de l'objet

Caractéristiques Gustatives : Bien que les interfaces gustatives ne soient pas encore très répandues actuellement, on peut déjà penser que les objets pourraient posséder des caractéristiques gustatives, qui serviraient à la gestion de ce retour sensoriel, par exemple dans le cas d'une application ludique où le but du jeu consisterait à attraper avec la bouche des bonbons lancés vers l'utilisateur.

Quelques exemples de caractéristiques envisageables : Goût, Acide, Amer, Salé, Sucré, Umami (savoureux en japonais), Frais, Fort, Épicé, Astringent, Persistance du goût.

Exemple de la Main Virtuelle : Comme on l'a vu précédemment, la principale technique utilisée est celle de la métaphore de la main virtuelle. La main virtuelle va servir à attraper un objet un mouvement ou à mettre en mouvement un objet tenu. Dans les deux cas, son état va influencer sur l'état de l'objet. La donnée importante est de connaître le degré d'ouverture de la main.

L'état de cette entité peut ainsi être modélisé comme suit :

- P : Position de la main
- O : Orientation de la main

- $\dot{P}$, ou VI : Vitesse linéaire de la main
- $\ddot{P}$ ou AI : Accélération linéaire la main

- $\dot{O}$ ou Va : Vitesse angulaire de la main
- $\ddot{O}$ ou Aa : Accélération angulaire la main

- **Ouverture de la main** : binaire (ouverte/fermée) ou non, en phase d'ouverture ou de fermeture, ou constante.

Par extension, nous appellerons *Main Virtuelle* le dispositif chargé de saisir un Objet en mouvement dans une simulation de réalité virtuelle, qu'il s'agisse effectivement d'une main, ou de tout autre chose (pince, filet à papillons, etc.).

Maintenant que nous avons défini l'état d'un objet et donné un exemple de modélisation de l'état de la Main virtuelle, nous pouvons décrire notre modélisation des Primitives d'Interaction Dynamique.

2.2.1.2 Lancer

Si l'on prend en compte la modélisation des objets dynamiques ci-dessus, l'objectif du lancer d'un objet revient finalement à faire correspondre l'état final de l'objet avec un "état final désiré" qui dépend de la tâche à réaliser et de l'environnement dans lequel évolue l'objet.

Dans le cas du tir à l'arc par exemple, la composante "Position" de l'"état final désiré" pour l'objet "Flèche" est celle du centre de la cible, l'"Orientation" est telle que la pointe de la flèche est dirigée vers la cible, et la "Vitesse" et l'"Accélération" sont nulles (après impact).

On peut donc dire que Lancer un objet, c'est lui **transmettre une Vitesse et/ou une Accélération à partir d'une Position initiale** telles que l'état final de l'objet correspondra à son état final désiré.

Du point de vue de l'Objet, le fait de Lancer est une action, ou plus exactement un *Geste*, qui part de l'*Utilisateur* et qui est dirigée vers le Monde en général, et plus précisément vers une *Cible*.

Si l'on met de côté les cas de lancer par pouvoirs télékinésiques ¹ que l'on pourrait simuler, la position de l'objet est liée à celle de l'Utilisateur jusqu'à ce que l'objet soit effectivement lancé. La position initiale de l'objet étant un paramètre important pour déterminer la trajectoire à lui faire suivre afin d'atteindre la cible visée, la position de l'Utilisateur devient de ce fait impossible à négliger. Si l'Utilisateur est Mobile (en mouvement) au moment de la réalisation du Geste, les données relatives à son déplacement (Position, Vitesse et Accélération) vont être transmises à l'Objet à lancer. Si en revanche l'Utilisateur est Fixe, la Position initiale de l'objet devient constante (on ne parle pas là du déplacement de l'objet inhérent au Geste), et donc beaucoup plus facile à gérer.

NB : dans le cas d'une télé-opération d'un robot lanceur d'objet par un humain, c'est évidemment le déplacement du robot qui sera à prendre en compte, et non pas celui de l'opérateur.

Le Geste peut être, comme on l'a vu précédemment, Continu ou Discret. Il est considéré Continu si l'action permettant de transmettre la Vitesse et/ou l'Accélération à l'Objet est relativement ample et continue. C'est le cas du lanceur au baseball par exemple. A l'inverse, il est Discret si cette action est très courte et discrète. C'est typiquement le cas au tir au pistolet.

¹ La télékinésie est la faculté de déplacer les objets à distance par la seule force de la pensée. Son étymologie est issue du grec, "télé" pour "à distance" et "kinesis" pour "le mouvement". De tels pouvoirs permettraient donc de lancer un objet sans qu'il soit tenu par l'utilisateur.

Le comportement de la Cible visée est également très important à prendre en compte. Si la cible est Fixe, comme le sont les buts de football par exemple, la tâche de l'utilisateur reste relativement simple. Elle se complique en revanche s'il doit en plus prendre en compte la Vitesse et/ou l'Accélération d'une cible Mobile dans l'ajustement de son geste. Cette situation se retrouve notamment dans beaucoup de sports collectifs (football, rugby, volley-ball, etc.) où la transmission d'un ballon à un partenaire en mouvement doit s'effectuer, le partenaire en question tenant le rôle de la Cible.

La conjugaison de ces trois composantes aboutit à la modélisation du Lancer illustrée par la figure 2.2. Le terme Utilisateur, plutôt utilisé dans le cadre d'une simulation, a été remplacé ici par le terme Sujet, plus général.

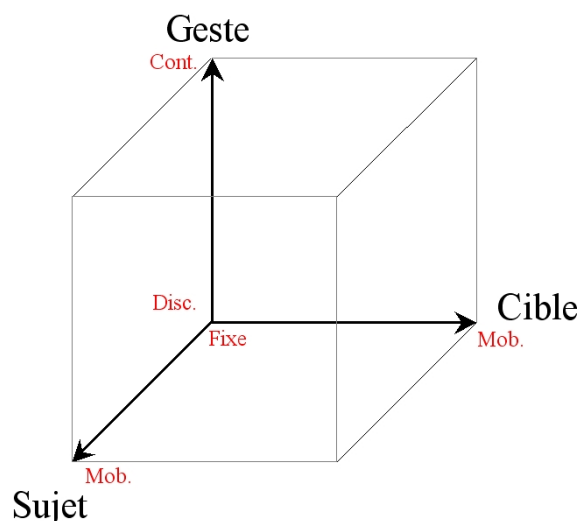


FIG. 2.2 – Modélisation du Lancer

Soit S le Sujet, G le Geste et C la Cible, le Sujet et la Cible pouvant être $_f$ Fixes ou $_m$ Mobiles, le Geste pouvant être $_d$ Discret ou $_c$ Continu. On obtient alors les sommets Sujet-Geste-Cible (SGC) du cube :

- $S_f G_d C_f$ (Sujet Fixe, Geste Discret, Cible Fixe) : il s'agit typiquement du cas du tir à l'arc sportif classique. Le tireur est positionné face à une cible fixe située à une certaine distance de lui. L'action permettant de mettre la flèche en mouvement, l'échappement, est une action très brève consistant principalement à ouvrir les doigts pour lâcher la flèche. Les séquences de tir durant une épreuve de biathlon remplissent ces mêmes critères.

- $S_f G_d C_m$ (Sujet Fixe, Geste Discret, Cible Mobile) : les conditions du lancer sont les mêmes que précédemment, mais cette fois-ci la cible est en mouvement. On trouve ce cas dans le tir à la carabine sportif sur cible mobile, à 10m et 50m.

A 10 mètres, la cible est fixée sur un rail et parcourt un trajet rectiligne de 2 mètres face au tireur, de droite à gauche et de gauche à droite, en deux vitesses de passage, lente (5 secondes) ou rapide (2,5 secondes). A 50m, la cible effectue un trajet de 10 mètres. La cible représentant un sanglier a donné à la discipline l'appellation de "Sanglier Courant". La chasse étant bien entendu une activité pouvant correspondre aux trois critères Sujet-Geste-Cible (SGC) indiqués.

- $S_f G_c C_f$ (Sujet Fixe, Geste Continu, Cible Fixe) : Cette fois-ci, le geste est continu, contrairement aux deux sommets du cube déjà étudiés. Il s'agit donc d'activités où l'action permettant de mettre l'objet en mouvement est continue et plutôt ample, avec un sujet qui ne se déplace pas, et une cible fixe. Le service au tennis entre dans cette description. Le joueur est positionné derrière la ligne de fond de court, il ne se déplace pas. Le mouvement de la raquette qui va venir frapper la balle est un mouvement Continu, et la cible visée est une petite zone de terrain située dans le carré de service adverse. Les services "tennis" ou à la cuillère au volley-ball font également partie de cette catégorie (mais pas le service smashé, voir plus loin).

Citons enfin le rôle du lanceur au baseball. On le considère Fixe car il n'est pas dans une optique de déplacement (il fait juste un pas en avant lors de son mouvement de lancer), son geste est Continu, et sa cible est un espace fixe appelé zone de prise.

- $S_f G_c C_m$ (Sujet Fixe, Geste Continu, Cible Mobile) : On pourrait trouver dans le football ou le rugby certaines situations correspondant à ces critères, mais l'exemple illustrant le mieux le cas d'un sujet fixe lançant un objet vers une cible en mouvement avec un geste continu reste celui du quarterback de football américain lançant une passe vers un receveur.
- $S_m G_d C_f$ (Sujet Mobile, Geste Discret, Cible Fixe) : Par rapport au premier sommet du cube que nous avons décrit, la différence concerne le Sujet qui est maintenant en mouvement. Pour illustrer cette combinaison de critères, on retrouve donc le tir à l'arc, mais dans sa version à cheval, notamment le Yabusame ² au Japon.

- $S_m G_d C_m$ (Sujet Mobile, Geste Discret, Cible Mobile) : On peut retrouver dans le paint-

²Le Yabusame est, avec le Kasagake et l'Inuoumono, l'un des trois types d'archerie montée traditionnelle au Japon.

ball, ce jeu qui consiste à tirer des boules de peinture sur ses adversaires à l'aide d'un marqueur en forme d'arme à feu, des situations remplissant ces critères. Il suffit qu'un joueur en mouvement tire sur un autre joueur en mouvement. Même s'il est préférable d'être fixe pour avoir une meilleure visée, cette situation peut survenir.

Les combats aériens de la Bataille d'Angleterre de 1940 entrent également dans cette description. Le pilote et son chasseur sont le Sujet et l'avion ennemi est la Cible. Aujourd'hui, les missiles dits "à tête chercheuse" viennent compliquer la problématique, mais l'idée reste la même.

- $S_m G_c C_f$ (Sujet Mobile, Geste Continu, Cible Fixe) : Il s'agit peut-être là d'une des plus simples combinaisons SGC à illustrer. En effet, le tir de penalty au football ou la transformation d'un essai au rugby en sont des exemples évidents. Dans ces deux cas, le sujet va être en mouvement à travers une course d'élan qui fait environ 3 pas (même si certains joueurs tirent sans élan). Le geste de la jambe est bien sûr Continu. Quant à la cible, il s'agit d'une zone plus ou moins grande située entre les poteaux, en dessous de la transversale pour le football, au dessus pour le rugby.

Citons enfin le cas du lanceur de pierre au curling.

- $S_m G_c C_m$ (Sujet Mobile, Geste Continu, Cible Mobile) : De nombreux sports de ballon offrent des situations de ce type. On retrouve en effet dans cette situation les cas de transmission de balle par un joueur en mouvement vers un autre joueur en mouvement. Football, rugby, handball, basket-ball sont autant de sports réunissant ces critères.

2.2.1.3 Attraper

Attraper un objet consiste à effectuer un rendez-vous spatio-temporel entre l'objet et la main, ou l'effecteur faisant office de main (pince, filet à papillons, etc.). L'utilisateur déplace sa main de manière à être en mesure de saisir l'objet au vol. Dans le cas d'une main réelle ou d'un effecteur tel qu'une pince, il faut refermer la main ou la pince pour que l'objet soit saisi. Nous appelons cette action de fermeture "l'activation" de la main ou de la pince.

Si la saisie s'effectue correctement, l'objet est lié à la main, et donc l'État de l'objet est assujéti à l'État de la main.

Attraper un objet, c'est donc simplement **déplacer, et éventuellement activer, la main virtuelle afin que l'état final de l'objet corresponde à l'état final de la main virtuelle.**

Du point de vue de l'Objet, l'action d'Attraper est l'inverse de celle du Lancer au sens où elle est orientée du Monde en général, et plus précisément d'un *Émetteur*, vers le *Sujet*, lequel doit effectuer un *Geste* afin de capter l'Objet.

L'objet que l'utilisateur doit attraper est généralement envoyé par une autre entité vivante, c'est pourquoi nous parlons ici d'émetteur. Si l'objet à attraper est une brique qui se détache d'un mur, on peut considérer que le mur est l'entité émettrice de la brique. En revanche, si l'objet à attraper est un être vivant capable de se mouvoir par lui-même, il est son propre émetteur. L'état de l'émetteur est intéressant à connaître car, tant que l'objet lui est encore assujéti, l'état de l'objet est le même que celui de l'émetteur. De plus, la vitesse de déplacement de l'émetteur sera transmise à l'objet, indépendamment de celle donnée par l'action qui mettra l'objet en mouvement et provoquera la désolidarisation de l'objet d'avec l'émetteur. Cette composante est ignorée si l'émetteur est fixe, et c'est pourquoi nous distinguons les émetteurs Fixes de ceux considérés comme étant Mobiles.

Dans le cas classique où le dispositif préhensile se situe à l'extrémité du bras de l'utilisateur, ou est tout au moins lié à celui-ci, l'état du sujet va influencer sur l'état de la main virtuelle. Il convient donc de différencier les cas où le sujet est Fixe de ceux dans lesquels il est Mobile (au moment de la réalisation du Geste), car sa vitesse de déplacement est transmise à la main virtuelle, indépendamment de la vitesse propre de cette dernière.

Nous considérons que le Geste est Continu si l'action permettant de capter l'objet nécessite un certain accompagnement du mouvement de l'objet, ou une continuation du mouvement pour assurer la saisie. De ce fait, les gestes Continus seront plutôt ceux consistant à ralentir progressivement l'objet, ou à l'amortir par un contact.

Dans le cas contraire, même si la main virtuelle se déplace un peu, le Geste sera défini comme Discret. C'est le cas le plus fréquemment rencontré, peu d'activités d'Attraper d'objet nécessitent un geste continu, même si l'on peut trouver quelques situations bien précises où cela est nécessaire.

En réunissant ces trois composantes, on obtient la modélisation de l'Attraper illustrée par la figure 2.3.

Soit E l'Émetteur, S le Sujet, et G le Geste, l'Émetteur et le Sujet pouvant être $_f$ Fixes ou $_m$ Mobiles, le Geste étant lui soit $_d$ Discret soit $_c$ Continu. On obtient les sommets ESG du cube suivants :

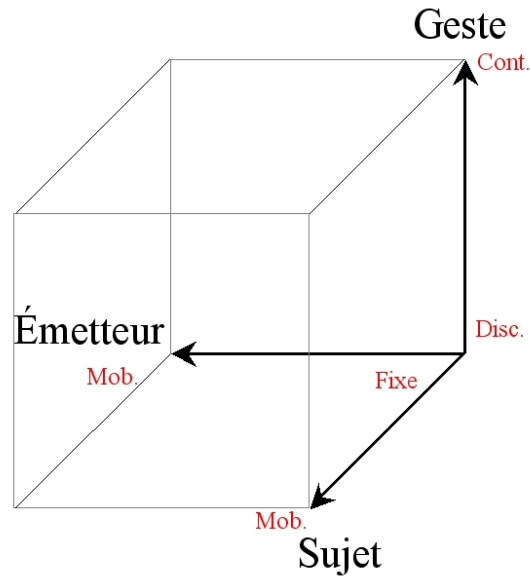


FIG. 2.3 – Modélisation de l'Attraper

- $E_f S_f G_d$ (Émetteur Fixe, Sujet Fixe, Geste Discret) : Au baseball, le rôle du receveur (catcher) consiste entre autres à attraper les balles manquées ou non tentées par le batteur. Il se tient derrière la plaque de but ou marbre (home plate) en position accroupie. L'émetteur de la balle est le lanceur (pitcher) qui a déjà été décrit précédemment. Le receveur positionne son gant sur la trajectoire de la balle et referme la main au moment du contact, c'est bien un geste plutôt Discret, car même si la main doit d'abord être déplacée au préalable, il n'y a pas de geste d'accompagnement.
- $E_f S_f G_c$ (Émetteur Fixe, Sujet Fixe, Geste Continu) : On peut trouver ce genre de situation au football, notamment sur une touche, où l'émetteur est normalement fixe puisqu'il doit garder les deux pieds au sol en dehors du terrain et lancer le ballon de derrière sa tête. Si le sujet est visé, il peut contrôler le ballon en effectuant un amorti de la poitrine. Ce geste sert bien à "attraper" la balle avec la poitrine, même si le ballon va ensuite retomber aux pieds du joueur par la force de gravité. De plus, le geste est continu, car autrement il y aurait une collision franche entre la balle et le torse du joueur qui ne permettrait pas de contrôler la balle.
- $E_f S_m G_d$ (Émetteur Fixe, Sujet Mobile, Geste Discret) : Cette combinaison ESG se retrouve dans le jeu de disque (frisbee) Ultimate. L'Ultimate est un sport sans contact qui se joue sur un terrain similaire à celui d'un terrain de football américain. L'équipe offen-

sive doit déplacer le disque à l'aide de passes jusqu'à ce qu'il soit attrapé dans la zone de but défensive de l'équipe adverse. Il est interdit de courir avec le disque ; un joueur doit s'arrêter lorsqu'il prend possession du disque, établir un point de pivot avec le pied et tenter une passe à un coéquipier.

Le receveur est généralement en mouvement et peut même être obligé de plonger pour attraper le disque. La saisie s'effectue de préférence à deux mains. La technique consiste à coincer le disque avec une main au-dessus du disque et l'autre au-dessous. C'est un geste brusque et rapide que nous considérons donc Discret.

- $E_f S_m G_c$ (Émetteur Fixe, Sujet Mobile, Geste Continu) : Une fois encore, c'est une phase de football qui illustre le mieux cette situation, ce qui est assez logique quand on connaît l'importance des contrôles de balle dans ce sport, lesquels se font principalement avec le pied, donc sans saisie possible, ce qui implique d'amortir le ballon.

Ici, si une remise en touche a été effectuée dans la course du sujet (émetteur fixe et sujet mobile donc), ce dernier peut vouloir contrôler le ballon du pied, en gardant la balle "collée" à la chaussure. Le mouvement nécessite d'accompagner le ballon dans sa course afin de l'amortir, sans le renvoyer, c'est typiquement un geste continu.

- $E_m S_f G_d$ (Émetteur Mobile, Sujet Fixe, Geste Discret) : Le gardien de but de hockey sur glace a une petite cage à défendre, ce qui fait qu'il est plutôt fixe. En revanche, les attaquants sont eux principalement en mouvement lors des lancers. Enfin, même si le palet va parfois être simplement détourné, le but est tout de même de l'attraper avec le gant, ce qui est un geste Discret. On retrouve bien là les trois composantes de cette situation.

- $E_m S_f G_c$ (Émetteur Mobile, Sujet Fixe, Geste Continu) : La pêche à l'épuisette est probablement l'activité répondant le mieux à ces critères. Le poisson étant à la fois l'objet à attraper et son propre émetteur puisqu'il se propulse naturellement. Le pêcheur doit généralement rester immobile sous peine de faire fuir sa proie. La technique consiste à faire passer doucement l'épuisette sous la cible, et à la remonter le plus vite possible. Il y a bien là un accompagnement à effectuer, le geste est Continu.

- $E_m S_m G_d$ (Émetteur Mobile, Sujet Mobile, Geste Discret) : On trouve cette situation dans certaines phases de jeu au rugby, notamment lorsqu'un joueur en possession du ballon est en mouvement et effectue une passe vers un coéquipier, lui-même en mouvement. Le joueur qui reçoit le ballon le capte généralement au niveau du ventre, avec un geste qui ne nécessite pas d'accompagnement spécial, donc Discret selon notre définition.

- $E_m S_m G_c$ (Émetteur Mobile, Sujet Mobile, Geste Continu) : Un bon exemple de sport réunissant ces trois composantes ainsi paramétrées est la pelote basque avec un chistéra. Le Grand Chistera, notamment, est une spécialité qui ne se joue qu'en France sur des frontons «place libre» de 80 mètres de long. La forme du gant (courbure accentuée et poche) facilite la réception, le blocage de la pelote, la décomposition du geste et la possibilité de prendre son élan pour le renvoi. Le geste nécessite d'accompagner la pelote afin d'amortir le choc avec le gant, et compléter la saisie.

2.2.1.4 Frapper

Comme vu précédemment, le Frapper est un geste conjuguant l'Attraper et le Lancer, mais qui garde des spécificités propres. Comme l'Attraper, il consiste en un rendez-vous spatio-temporel entre un objet en mouvement et, cette fois-ci, une **surface de frappe** (main, raquette, etc., devant entrer en collision avec l'objet en mouvement). En revanche, son geste s'effectue dans le but de renvoyer l'objet vers une cible, de manière un peu analogue à celle du Lancer, puisque la finalité du Frapper est également de faire correspondre l'état final de l'objet avec un "état final désiré" qui dépend de la tâche à réaliser, mais en rapport avec une Cible. En revanche, le geste du Frapper peut ne transmettre aucune Vitesse à l'objet, voire même diminuer la Vitesse propre de l'objet.

On peut donc dire que Frapper un objet, c'est **déplacer la surface de frappe de manière à provoquer une collision avec l'objet en mouvement telle que l'état final de l'objet correspondra à son état final désiré.**

Du point de vue de l'Objet, le Frapper est une action globale, qui part d'un *Émetteur* vers le *Sujet*, lequel doit effectuer un *Geste* afin de renvoyer l'objet vers une *Cible*.

L'émetteur est ici le même que dans le cas de l'Attraper, et son état est intéressant à connaître pour les mêmes raisons. Nous retrouvons donc la distinction entre un émetteur Fixe et un émetteur Mobile.

La surface de frappe est généralement liée à l'utilisateur, qu'il s'agisse d'une partie même de son corps (main, pied, tête, etc.), ou qu'il s'agisse d'un objet tenu en main (raquette). Il est donc logique de différencier les cas où le sujet est Fixe de ceux dans lesquels il est Mobile (au moment de la réalisation du Geste), puisque sa vitesse de déplacement est transmise à la surface

de frappe, indépendamment de la vitesse propre de cette dernière.

Contrairement au Lancer, l'Objet à Frapper possède une vitesse initiale non nulle avant que le Geste ne lui en transfère une. Le geste va donc consister à modifier l'amplitude et la direction de la vitesse de l'objet afin de le renvoyer vers un objectif précis. Toutefois, si la modification de la direction de la vitesse de l'objet est impérative, celle concernant son amplitude est facultative. De ce fait, nous distinguons les gestes Discrets consistant simplement à mettre la surface de frappe en opposition, orientée dans la bonne direction, des gestes Continus qui ont pour but supplémentaire d'accélérer ou de ralentir l'objet.

On note qu'ici les Caractéristiques Physiques de la surface de frappe et de l'Objet ont une importance accrue. Elles influent en effet grandement sur le changement de vitesse et de direction de l'objet suite à la collision entre celui-ci et la surface de frappe.

De la même manière que dans le cas du Lancer, le comportement de la Cible visée est également important à prendre en compte, et pour les mêmes raisons. Ainsi, une cible Fixe sera plus facile à appréhender et l'ajustement du geste en sera simplifié par rapport au cas d'une cible en mouvement.

En réunissant toutes ces composantes, on obtient la modélisation du Frapper en 4 dimensions illustrée par la figure 2.4.

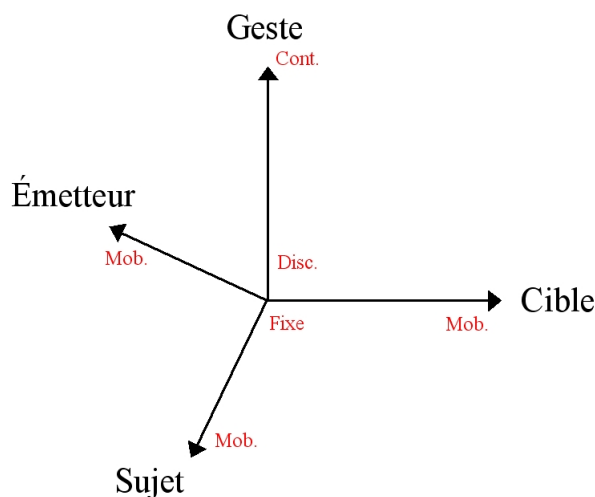


FIG. 2.4 – Modélisation du Frapper

Soit E l'Émetteur, S le Sujet, G le Geste et C la Cible. l'Émetteur, le Sujet et la Cible

peuvent être f Fixes ou m Mobiles, tandis que le Geste est ou d Discret, ou c Continu. On obtient un total de 16 sommets ESGC. Nous n'allons cette fois pas détailler l'intégralité des sommets. En effet, il n'existe pas, pour chacun des sommets, une activité dont l'interaction corresponde spécifiquement aux paramètres ESGC du sommet. En revanche, l'analyse de la tâche de réception au Volley-ball peut recouvrir une majorité des cas :

– **Émetteur :**

L'émetteur peut être Fixe ou Mobile, selon qu'il s'agisse par exemple d'un service "tennis" (les deux pieds au sol, sans élan) ou d'un service smashé (le serveur s'élance, saute, et frappe la balle durant le saut).

– **Sujet :**

Si le Sujet est correctement placé par rapport à la trajectoire de la balle, il sera Fixe. Si en revanche il est loin de la balle, il devra se déplacer et effectuer sa réception pendant son déplacement, il sera donc Mobile.

– **Geste :**

Un bon placement du Sujet facilite son Geste. En effet, si le réceptionneur est idéalement positionné, sous la balle et orienté vers son passeur, le Geste effectué avec la surface de frappe (les avant-bras du Sujet) sera Discret. Dans le cas contraire, le Sujet devra effectuer un geste Continu plus complexe, afin de renvoyer la balle dans la direction voulue.

– **Cible :**

La Cible du réceptionneur est généralement fixe, étant donné qu'il doit diriger la balle vers son passeur, accolé au filet, et normalement immobile au moment de la réception. Si par contre la réception se fait après une première touche de balle par un de ses coéquipiers, le Sujet devra diriger le ballon de manière à pouvoir être attaqué par un de ses partenaires. A haut niveau, l'attaquant est déjà en mouvement pour effectuer son smash durant cette phase de jeu, la Cible du Sujet est donc Mobile.

Voici néanmoins une illustration de quelques sommets ESGC usuels :

– $E_f S_f G_d C_f$ (Émetteur Fixe, Sujet Fixe, Geste Discret, Cible Fixe) : Il s'agit d'un cas particulier du baseball. Plutôt que de tenter le "Home run", c'est à dire le coup parfait faisant sortir la balle des limites du terrain et donc la mettant hors de portée de ses adversaires, le batteur peut choisir de simplement mettre sa batte en opposition. La balle va alors toucher le sol rapidement, ce qui empêchera l'arrêt "de volée" éliminatoire. Dans cette situation, le lanceur est Fixe, le Sujet également, le Geste l'est aussi afin de ne pas donner de vitesse à la balle, et la Cible est une petite surface au sol située devant le Sujet.

– $E_f S_f G_c C_f$ (Émetteur Fixe, Sujet Fixe, Geste Continu, Cible Fixe) : Cette configuration

est celle qui s'applique aux activités sportives mettant en oeuvre une tâche de frapper, sous réserve que le sujet soit correctement placé. On retrouve donc le batteur de baseball, mais aussi le joueur de tennis, de ping-pong, etc.

- $E_f S_m G_c C_f$ (Émetteur Fixe, Sujet Mobile, Geste Continu, Cible Fixe) : Ces paramètres modélisent typiquement le smash de volley-ball. L'émetteur est le passeur, normalement Fixe. Le Sujet quant à lui est en mouvement au moment de frapper la balle puisqu'il saute afin de la prendre le plus haut possible. Le Geste permettant de donner la vitesse et la direction de la balle est Continu, et la cible est une petite portion de terrain adverse.
- $E_f S_m G_c C_m$ (Émetteur Fixe, Sujet Mobile, Geste Continu, Cible Mobile) : Si l'on prend le cas d'un joueur de football en mouvement (Sujet Mobile) devant passer la balle (Geste Continu) dans la course d'un co-équipier (Cible Mobile) alors que le ballon vient juste de lui être transmis par un de ses partenaires effectuant une remise en touche (Émetteur Fixe), on a une interaction dynamique correspondant aux paramètres de ce sommet.
- $E_m S_f G_c C_m$ (Émetteur Mobile, Sujet Fixe, Geste Continu, Cible Mobile) : Ce sommet ESGC peut également être illustré par une phase de jeu de football. Le Sujet sert de relais (Fixe) à un co-équipier en mouvement (Émetteur Mobile) qui lui transmet le ballon. En une seule touche de balle (Geste Continu), le Sujet renvoie le ballon dans la course de son partenaire (Cible Mobile). Dans cette phase de "une-deux", on constate que l'Émetteur et la Cible sont une entité unique.
- $E_m S_m G_d C_f$ (Émetteur Mobile, Sujet Mobile, Geste Discret, Cible Fixe) : Lorsqu'un joueur effectue un contre au volley-ball, l'Émetteur est l'attaquant adverse, on a déjà indiqué qu'il était mobile. Le Sujet effectue lui-même un saut pour aller contrer le plus haut possible, il est donc également mobile. En revanche, son action consiste à mettre ses bras en opposition dans le camp adverse, c'est un Geste Discret. Comme à l'accoutumée, la Cible visée est une petite surface du terrain adverse, elle est Fixe.

On peut remarquer que les Cibles visées sont généralement Fixes puisqu'il s'agit d'une surface située dans le terrain de l'adversaire. Elles ne sont Mobiles que lorsqu'elles représentent un partenaire en mouvement.

2.2.1.5 Général

En observant les modélisations précédentes, on constate qu'il y a deux composantes communes aux trois Primitives d'Interaction Dynamique : le *Sujet* et le *Geste* qu'il effectue. Par ailleurs, les actions sont dirigées vers deux directions : de l'*Émetteur* (du Monde) vers le sujet ; et du sujet vers la *Cible* (vers le Monde).

On en déduit la modélisation globale des interactions dynamiques illustrée par la figure 2.5, consistant en deux cubes ayant une face commune. Nous l'appelons le D-BiCube (Dynamic BiCube).

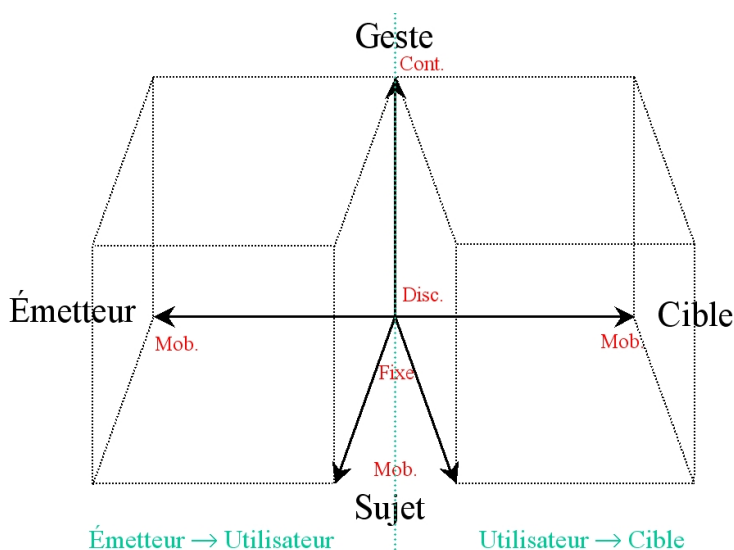


FIG. 2.5 – Modélisation des Interactions Dynamiques : le D-BiCube

La partie gauche, Émetteur-Sujet-Geste (du Monde vers le Sujet), est l'illustration de l'Attraper, tandis que la partie droite, Sujet-Geste-Cible (du Sujet vers le Monde), définit le Lancer. Prise dans son ensemble, Émetteur-Sujet-Geste-Cible (du Monde vers le Sujet vers le Monde), cette modélisation correspond à celle du Frapper.

Par ailleurs, certains sommets n'ont pas été illustrés par des activités dont la technique correspond spécifiquement aux critères ESGC indiqués. Il s'agit plutôt de situations spéciales, impliquant une action inhabituelle dont l'exécution correspond aux-dits critères.

On pourrait donc imaginer des sports ou activités virtuels, tirant partie de propriétés ou caractéristiques physiques (du monde ou des objets) impossibles à réaliser ou reproduire dans le réel : balle qui ne peut pas être ralentie (les contacts avec l'environnement ou la surface de frappe n'entraînent aucun amortissement), balle que l'émetteur peut encore contrôler à distance par

des gestes ou par le regard, accessoire en forme de disque sur lequel la balle reste collée au moindre contact, quelle que soit sa vitesse au moment de l'impact, etc.

2.2.2 Les exigences de l'interaction dynamique

2.2.2.1 Problématique

La problématique issue de l'interaction avec un objet en mouvement ou à mettre en mouvement est centrée autour de l'Utilisateur, du "Sujet" de notre modélisation. Ce dernier doit pouvoir percevoir, anticiper, et/ou générer l'état de l'objet avec lequel il doit interagir. Il y a donc des exigences à remplir concernant les 4 acteurs de l'interaction dynamique.

2.2.2.2 Émetteur

Le Sujet doit avoir une bonne perception de l'état de l'émetteur. C'est à dire qu'il doit pouvoir au moins estimer la position et la vitesse de l'émetteur.

2.2.2.3 Sujet

Le Sujet doit également pouvoir estimer et contrôler son propre état. Ainsi, il doit être capable de connaître sa propre position/orientation et sa propre vitesse, mais il doit également pouvoir modifier l'une et l'autre de manière intuitive et contrôlée.

2.2.2.4 Geste

Le geste de l'Utilisateur est effectué vers l'Objet ou avec l'Objet, mais dans tous les cas, le Sujet doit pouvoir estimer la position et la vitesse de l'Objet, et, pour le Lancer, le poids de l'Objet. Cela signifie qu'il doit également avoir conscience des forces exercées par l'environnement (gravité, vent, etc.).

Concernant son geste en lui-même, le Sujet doit avoir le contrôle de son mouvement et une bonne perception de celui-ci.

2.2.2.5 Cible

Enfin, le Sujet doit pouvoir estimer la distance le séparant de la Cible (donc sa position), la vitesse et la direction de la Cible si elle est mobile, ainsi que ses dimensions.

2.2.2.6 Conclusion

Beaucoup de paramètres vont influencer sur le confort et la performance de l'utilisateur durant une tâche d'interaction dynamique. Au niveau logiciel, le choix des détails graphiques de la scène virtuelle va influencer sur la perception des acteurs par le Sujet.

En ce qui concerne les périphériques de sortie, le dispositif choisi pour l'affichage aura également un grand rôle. Ainsi, les "Offset" (décalages) générés par l'utilisation de systèmes de projection sur grand écran par exemple sont susceptibles de gêner le Sujet tant dans la perception qu'il a de sa position que dans celle qu'il a de son geste. Mais si utiliser un visiocasque résoudra ce problème, nous avons vu dans le Chapitre 1 qu'il en soulèvera d'autres.

Enfin, il convient d'utiliser des périphériques d'entrée correctement calibrés, disposant d'une faible latence et d'une précision élevée, afin que le Sujet ait un contrôle optimal de son mouvement, c'est-à-dire que le mouvement retranscrit dans l'environnement virtuel corresponde le plus précisément possible (position et délai) au mouvement effectué par le Sujet dans le monde réel.

2.3 Conception d'Aides Logicielles pour l'Interaction Dynamique

2.3.1 Objectifs et moyens

Les Aides Logicielles que nous développons ont pour but de faciliter la réalisation d'une tâche dynamique, et/ou d'aider à l'apprentissage de cette tâche. Elles consistent en un retour sensoriel, principalement visuel mais aussi auditif ou haptique, d'informations dont l'utilisateur ne dispose pas dans le réel. Compte tenu du temps moyen de détection d'une odeur observée par [Tij06] (environ 7s, mais très variable selon les diffuseurs, les odeurs, les sujets) par rapport à la durée d'une tâche dynamique, il ne paraît pas aujourd'hui possible d'envisager un retour olfactif en tant qu'Aide Logicielle pour l'Interaction Dynamique (ci-après notée ALID).

2.3.2 Structure d'une ALID

La structure d'une Aide Logicielle pour l'Interaction Dynamique est composée des Propriétés suivantes :

- **Nom :**

Il s'agit simplement du nom de l'Aide Logicielle, permettant de l'identifier et d'y faire référence.

– **But :**

Comme dit précédemment, les ALID peuvent avoir pour but soit de faciliter la réalisation d'une tâche mettant en oeuvre une ou plusieurs interactions dynamiques, soit de permettre un meilleur apprentissage de la-dite tâche, soit éventuellement les deux. Il est donc intéressant de préciser l'objectif souhaité d'une ALID en développement, ou la finalité réelle d'une ALID validée.

Nous émettons l'hypothèse qu'une Aide dédiée à la "Réalisation" fournira généralement une information cruciale pour la tâche à effectuer, tandis qu'une Aide plutôt orientée vers l'"Apprentissage" aura tendance à simplement permettre une meilleure perception de l'Objet, de sa trajectoire, etc.

– **PID :**

Nous avons isolé et étudié 3 Primitives d'Interaction Dynamique (PID), ayant un fonctionnement et des exigences propres. Il convient donc d'indiquer pour laquelle ou lesquelles de ces PID l'Aide Logicielle est destinée.

En sus de cela, on peut plus précisément indiquer dans quel(s) cas précis de la modélisation appropriée (Sujet-Geste-Cible pour le Lancer ; Émetteur-Sujet-Geste pour l'Attraper ; Émetteur-Sujet-Geste-Cible pour le Frapper) l'ALID doit être utilisée.

– **Type :**

Nous l'avons précisé dans le paragraphe précédent, une ALID consiste à fournir un retour sensoriel. C'est ici que l'on indique si ce retour est Visuel, Auditif ou Haptique.

Plus en détails, on spécifiera si elle consiste, dans le cas d'une aide visuelle, en l'affichage d'un ou plusieurs Objets 2D ou 3D, de Texte ou de l'utilisation d'une Caméra spéciale. Pour une aide sonore, il peut s'agir d'un son ponctuel ou d'un son continu. Enfin, dans le cas d'une aide haptique, nous noterons si l'on utilise une stimulation ponctuelle ou continue, ou plutôt un guide haptique.

– **Description :**

Cette propriété permet de décrire l'ALID et son fonctionnement complet de manière rédigée. Elle doit permettre à la fois d'en déduire les valeurs de certaines autres propriétés, mais aussi de vérifier si celles qui ont été fixées sont cohérentes avec le "cahier des charges" de l'ALID donné ici.

– **Genre :**

Nous indiquons à cet endroit si l'Aide Logicielle est censée être Statique, c'est à dire fixe à un endroit de l'environnement virtuel, ou Dynamique. Dans ce dernier cas, cela signifie que l'aide (un objet graphique, une source sonore ou un guide haptique par exemple) peut se déplacer, d'une manière ou d'une autre.

– **Pré-condition :**

L'intérêt de cette propriété est de spécifier, s'il y a lieu, la condition d'activation de l'ALID. En d'autres termes, tant que la condition n'est pas remplie, l'Aide Logicielle n'est pas perceptible (l'Objet graphique n'est pas affiché, le son n'est pas émis, le retour haptique n'est pas présent).

– **Post-condition :**

Cette propriété a pour objectif de spécifier, si nécessaire, la condition d'inactivation de l'ALID. Il s'agit là de définir la condition qui, si elle est respectée, mettra fin à l'Aide Logicielle (l'Objet graphique ne sera plus affiché, le son cessera d'être émis, le retour haptique ne sera plus envoyé).

– **État :**

A l'instar des Objets dynamiques de la simulation, l'État d'une ALID est composé des données relatives à sa Position, son Orientation, sa Vitesse et son Accélération. Il est cependant évident que, selon le Type d'Aide Logicielle utilisé, certaines de ces données peuvent ne pas être applicables.

Ainsi, une aide de type "Son ponctuel" pourra être munie d'une Position, correspondant à la position de la source, visible ou non, de l'onde sonore, mais sa Vitesse Angulaire pourra très probablement être ignorée (sauf si la source possède une rotation sur elle-même, influant sur la direction de propagation de l'onde sonore, laquelle étant prise en compte dans la simulation).

Si l'aide est constituée de multiples objets, on pourra selon les situations indiquer l'État de tous les objets constituant l'aide, ou de seulement un seul d'entre eux (on en déduira alors l'État des autres objets s'ils sont liés entre eux par une relation).

Enfin, on ajoute une composante binaire "Active" qui indique si oui ou non l'Aide Logicielle est actuellement activée.

– **Fonctionnement :**

L'objectif de cette propriété est de décomposer de manière plus formelle le fonctionnement de l'ALID, et ce en indiquant la ou les Caractéristiques sujettes à modification, et de quelle manière. Le cas échéant, on pourra également indiquer les modifications des composantes de l'État de l'Aide.

Comme d'autres propriétés, celle-ci ne sera pas pertinente pour toutes les Aides Logicielles imaginables.

Les Paramètres du Fonctionnement seront détaillés plus tard.

Cette structure est bien entendu ouverte et pourra être complétée au fur et à mesure des avancées des travaux sur ce sujet.

Le tableau 2.1 synthétise la Structure d'une Aide Logicielle pour l'Interaction Dynamique. Dans ce tableau, une propriété est notée d'une astérisque (*) si elle possède des sous-propriétés (ou Paramètres), détaillées plus tard. Les valeurs notées entre guillemets (") et séparées par des point-virgules (;) sont des valeurs fixes que la propriété peut effectivement prendre. Les valeurs notées sans guillemet et *en italique* font référence à un type de valeur, détaillé plus bas.

Propriété	Description	Valeur
Nom	Identifie l'ALID	<i>Texte</i>
But	L'objectif de l'aide logicielle, si elle porte sur l'apprentissage de la tâche ou sur la réalisation de la tâche	"Apprentissage" ; "Réalisation"
PID*	La ou les Primitives d'Interaction Dynamique sur laquelle porte l'Aide Logicielle	"Lancer" ; "Attraper" ; "Frapper"
Type*	Le type d'ALID, s'il s'agit d'une aide visuelle, auditive ou haptique	"Visuel" ; "Auditif" ; "Haptique"
Description	Donne une description textuelle détaillée de l'ALID	<i>Texte</i>
Genre	Indique si l'aide est fixe dans l'espace ou peut se déplacer	"Statique" ; "Dynamique"
Pré-condition	Spécifie la condition d'activation de l'ALID	<i>Condition</i>
Post-condition	Spécifie la condition d'inactivation de l'ALID	<i>Condition</i>
État	Indique l'état de l'ALID	<i>État</i>
Fonctionnement*	La décomposition plus formelle du fonctionnement de l'ALID	<i>Modification</i>

TAB. 2.1 – Structure d'une Aide Logicielle pour l'Interaction Dynamique.

2.3.2.1 Types de valeurs

Les types de valeurs utilisés dans le Tableau 2.1 sont :

– **Texte :**

Il s'agit simplement d'un texte explicatif.

– **Condition :**

Elle peut être écrite de manière textuelle ("le projectile est à moins de 5 mètres de l'utilisateur") ou de manière plus formelle ($|Position_{projectile} - Position_{utilisateur}| < 5m$)

– **État :**

L'ensemble {Position, Orientation, Vitesse linéaire, Accélération linéaire, Vitesse angulaire, Accélération angulaire}. Selon le Type de l'ALID, certains éléments de cet ensemble peuvent être ignorés.

– **Modification :**

Une Modification est un ensemble de composantes permettant de décrire l'évolution de l'aide logicielle durant la simulation. Elle sera explicitée plus tard.

2.3.2.2 Paramètres

Nous avons indiqué précédemment que les Propriétés PID, Type et Fonctionnement possédaient un ou plusieurs Paramètres. Nous allons maintenant voir en quoi ils consistent.

PID → Modèle PID : Rappelons tout d'abord que la Propriété PID permet de spécifier sur quelle(s) Primitive(s) d'Interaction Dynamique va porter l'Aide Logicielle. Ces Primitives sont "Lancer", "Attraper" et "Frapper".

Le Paramètre Modèle PID permet d'indiquer, selon notre modélisation, pour quelle(s) situations(s) de la Primitive sélectionnée l'Aide Logicielle est adaptée. On le note sous la forme :

- **Sujet-Geste-Cible :** pour le Lancer.
- **Émetteur-Sujet-Geste :** pour l'Attraper.
- **Émetteur-Sujet-Geste-Cible :** pour le Frapper.

Chaque composante peut être "Fixe" / "Mobile" pour l'Émetteur, le Sujet et la Cible, ou bien "Discret" / "Continu" pour le Geste, conformément aux modélisations développées précédemment (Modélisation des Primitives d'Interaction Dynamique).

Type → Sous-Type : Le Type de l'ALID indique s'il s'agit d'une aide Visuelle, Auditive ou Haptique. Le Sous-Type permet d'entrer un peu plus en détail dans la composition de l'Aide Logicielle. Le Tableau 2.2 contient les valeurs possibles des Sous-Types selon le Type de base.

Type	Sous-Type
Visuel	"Objet 2D unique" ; "Objet 3D unique" ; "Multiples Objets 2D" ; "Multiples Objets 3D" ; "Caméra", "Texte"
Auditif	"Son ponctuel" ; "Son continu"
Haptique	"Stimulation ponctuelle" ; "Stimulation continue" ; "Guide"

TAB. 2.2 – Tableau des Sous-Types.

Les valeurs utilisées dans le Tableau 2.2 pour le Type "Visuel" sont :

– **Objet 2D unique :**

Un unique objet en deux dimensions, comme un cercle, un carré, un polygone, etc.

– **Objet 3D unique :**

Un unique objet en trois dimensions, qui peut être simple (une sphère, un cylindre) ou complexe (le modèle 3D d'un animal par exemple).

– **Multiples Objets 2D :**

Il s'agit d'un ensemble d'Objets 2D. On peut penser à un alignement d'objets 2D pouvant symboliser une trajectoire. Ce type d'aide fait partie de ceux employés depuis longtemps dans les jeux video et dont nous avons fait mention dans le chapitre 1 à l'aide du jeu de billard "Side Pocket".

– **Multiples Objets 3D :**

Il s'agit d'un ensemble d'Objets 3D. Imaginons plusieurs anneaux espacés, symbolisant autant de zones de passage de référence pour un objet. Cet exemple fait notamment référence à un mode de jeu de "Mario Golf : Toadstool Tour" illustré par la figure 2.6, développé pour la console GameCube de Nintendo.



FIG. 2.6 – Capture d'écran du jeu "Mario Golf : Toadstool Tour"

– **Caméra :**

L'idée est d'utiliser une autre caméra que celle utilisée pour représenter le point de vue de l'utilisateur dans une simulation. Cette caméra supplémentaire peut ainsi offrir un point de vue et/ou un angle de vue normalement inaccessibles à l'utilisateur.

– **Texte :**

Une simple information textuelle. On peut par exemple afficher la valeur numérique d'une vitesse (celle de l'objet en mouvement, du vent, etc.).

Pour le Type "Auditif", les valeurs sont :

– **Son ponctuel :**

Il s'agit là d'un son dont la durée est relativement courte (généralement moins d'une seconde) et qui est joué une seule fois.

– **Son continu :**

Un son continu peut être soit un enregistrement sonore de courte durée qui est joué en boucle, soit une note simple maintenue jusqu'à l'inactivation de l'Aide. Il est envisageable de faire varier certaines Caractéristiques du son telles que la fréquence ou l'amplitude pendant qu'il est joué (voir Fonctionnement)

Enfin, les valeurs utilisées pour le Type "Haptique" sont :

– **Stimulation ponctuelle :**

A l'instar du son ponctuel pour une aide auditive, une stimulation haptique ponctuelle est un retour haptique de courte durée envoyé à l'utilisateur.

– **Stimulation continue :**

De la même manière, il s'agit ici d'appliquer un retour haptique continu à l'utilisateur jusqu'à ce que l'Aide soit inactivée. Là encore, certaines Caractéristiques, comme l'amplitude ou la direction, peuvent être modifiées "à la volée".

– **Guide :**

On retrouve ici l'idée du Guide Haptique développée par Otmane et al. [OCMH00] et détaillée dans le Chapitre 1 (Fixtures Virtuelles). Il va donc s'agir d'un objet haptique possédant des capacités d'attraction/répulsion vis à vis de l'utilisateur.

Fonctionnement → Modification : Nous allons maintenant détailler les Paramètres du Fonctionnement, qui doit permettre de décomposer de manière formelle la façon dont l'Aide Logicielle évolue au cours de la simulation.

Ces paramètres consistent en une unique Modification, ou en un ensemble de Modifications. Chaque Modification est définie par ses composantes suivantes :

– **Nom :**

Il s'agit du nom de la Modification, ce qui permet de l'identifier et d'y faire référence.

– **Type :**

Le Type de la Modification indique si l'Aide est modifiée en permanence ou si elle ne l'est que lorsqu'une condition est préalablement remplie.

– **Caractéristique :**

Il s'agit de la Caractéristique de l'Aide qui est soumise à modification. On fait là référence aux Caractéristiques de l'aide logicielle, héritées de son Type ("Visuel", "Auditif" ou

"Haptique"). Il s'agit des mêmes Caractéristiques Graphiques, Auditives et Haptiques que celles listées précédemment lors de la modélisation des Objets.

Plutôt qu'une Caractéristique, la Modification peut également porter sur une composante de l'État de l'ALID (Position, Vitesse, etc.).

– **Effet :**

L'effet de la modification proprement dit.

– **Pré-condition :**

Cette composante n'est valable que si la Modification est de Type "Discret". Il s'agit de la Condition qui doit être vérifiée pour que l'Effet soit appliqué.

– **Post-Condition :**

Cette composante n'est valable que si la Modification est de Type "Discret". Si cette composante est renseignée, elle indique alors la Condition qui doit être vérifiée pour que l'Effet cesse d'être appliqué, la Caractéristique reprenant alors sa valeur d'origine.

Le tableau 2.3 contient les valeurs possibles des composantes d'une Modification. De la même manière que pour le tableau 2.1, les valeurs notées entre guillemets (") et séparées par des point-virgules (;) sont des valeurs fixes que la composante peut effectivement prendre. Les valeurs notées sans guillemet et *en italique* font référence à un type de valeur, qui soit a déjà été explicité précédemment, soit est détaillé plus bas.

Composante	Valeur
Nom	<i>Texte</i>
Type	"Discret" ; "Continu"
Caractéristique	<i>Caractéristique</i>
Effet	<i>Effet</i>
Pré-condition ⁺	<i>Condition</i>
Post-condition ⁺	<i>Condition</i>

TAB. 2.3 – Tableau des composantes d'une Modification.

Les composantes notées d'un "+" ne sont applicables que si le Type de la Modification est "Discret".

Les nouvelles valeurs utilisées dans le Tableau 2.3 sont :

– **Caractéristique :**

La Caractéristique Graphique, Auditive ou Haptique sur laquelle porte la Modification. Voir les paragraphes "Caractéristiques Graphiques", "Caractéristiques Auditives" et "Caractéristiques Haptiques" de la section "Modélisation des Primitives d'Interaction Dyna-

mique" pour plus de détails.

Éventuellement, il peut s'agir d'une composante de l'État de l'ALID.

– **Effet :**

L'Effet de la modification peut être écrit de manière textuelle ("La couleur de l'Aide devient rouge") ou de manière plus formelle ("Couleur = #FF0000")

2.3.2.3 Exemple d'ALID

Nous allons maintenant illustrer la Structure d'une Aide Logicielle pour l'Interaction Dynamique par un exemple concret. L'Aide décrite ici a été envisagée pour l'application décrite dans le chapitre 3, mais n'a pas été implémentée ni testée faute de temps.

On considère qu'il s'agit d'une application de lancer de balle dans laquelle l'utilisateur se tient debout et doit lancer la sphère sur une cible posée à plat sur le sol, à environ 5 mètres de lui. La trajectoire de la sphère sera une trajectoire parabolique, avec une portée ayant une altitude plus basse que la position de départ (au moment où la balle est lâchée). On calcule donc, en fixant la position et la vitesse initiale de la boule au moment du lancer, une trajectoire parabolique se terminant au centre de la cible. 5 anneaux seront alors disposés le long de cette trajectoire parabolique.

La Structure de cette Aide Logicielle est illustrée par le tableau 2.4.

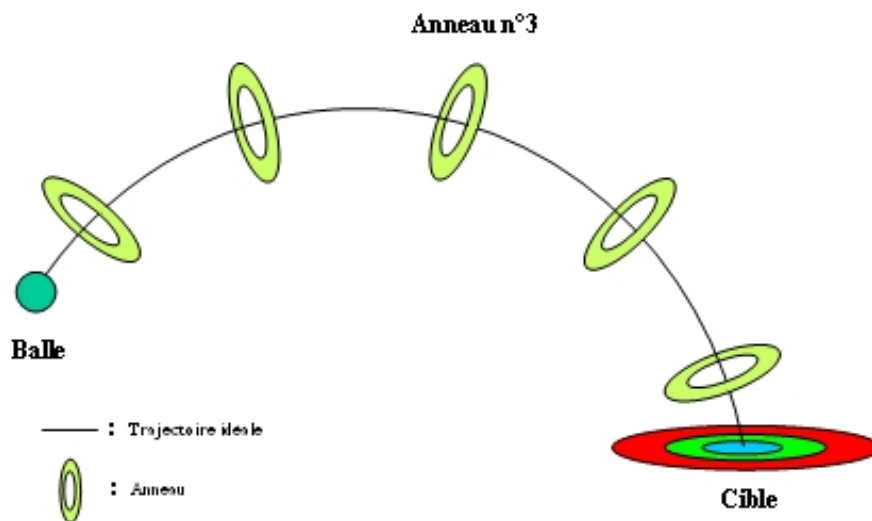


FIG. 2.7 – Positions des anneaux le long d'une trajectoire parabolique

L'ALID "GuideAnneaux" a pour Paramètres :

– **PID** → **Modèle PID** : $S_f G_c C_f$.

Propriété	Valeur
Nom	GuideAnneaux
But	"Apprentissage"
PID	"Lancer"
Type	"Visuel"
Description	Cette aide consiste en l'affichage de 5 anneaux de couleur dorée. Le centre de chacun de ces anneaux est situé le long d'une trajectoire parabolique précalculée avec des paramètres de tirs fixés (position et vitesse). Les anneaux sont disposés le long de la trajectoire parabolique, aux points de passage correspondant à 10, 30, 50, 70 et 90% de la durée totale du vol de la balle dans le cas de la trajectoire idéale (voir figure 2.7). Si la balle lancée par l'utilisateur passe à travers un anneau, ce dernier devient vert. Si en revanche elle traverse le plan de l'anneau sans passer à travers lui, il devient rouge. Les anneaux sont affichés en toutes circonstances. La position de l'anneau n°3 est utilisée comme position de référence de l'ALID.
Genre	"Statique"
Pré-condition	aucune
Post-condition	aucune
État	État_Anneau_#3
Fonctionnement	ModificationVert ; ModificationRouge

TAB. 2.4 – Structure de l'ALID "GuideAnneaux".

- **Type** → **Sous-Type** : Multiples Objets 3D.
- **État** → **État_Anneau_#3** : Soit $P_{BalleTI}(t)$ la position en fonction du temps d'une balle qui suit la trajectoire parabolique avec t_0 le moment du lâcher et t_{final} le moment où la balle touche la cible, alors on a :

$$P_{Anneau_3} = (P_{BalleTI}((t_{final} - t_0) * 0.5))$$

Par ailleurs, l'Orientation de l'Anneau se déduit des vecteurs normaux à la tangente au point P_{Anneau_3} de la trajectoire idéale.

De plus, les tableaux 2.5 et 2.6 illustrent respectivement le contenu de ModificationVert et ModificationRouge, soit les Paramètres du Fonctionnement de l'ALID.

Composante	Valeur
Nom	ModificationVert
Type	"Discret"
Caractéristique	Couleur
Effet	Couleur = Vert
Pré-condition	La balle a traversé l'Anneau
Post-condition	aucune

TAB. 2.5 – Composantes de ModificationVert.

Composante	Valeur
Nom	ModificationRouge
Type	"Discret"
Caractéristique	Couleur
Effet	Couleur = Rouge
Pré-condition	La balle est derrière l'Anneau et ne l'a pas traversé
Post-condition	aucune

TAB. 2.6 – Composantes de ModificationRouge.

2.4 Conclusion

Après analyse de l'interaction dynamique, nous avons isolés 3 PID (Primitives d'Interaction Dynamique). Ces 3 Primitives sont : le "Lancer, l'"Attraper" et le "Frapper". Nous les avons chacune analysées et modélisées, et nous en avons déduit le D-BiCube, modélisation bi-cubique des Interactions Dynamiques.

Nous avons ensuite présenté nos Aides Logicielles pour l'Interaction Dynamique, avec leur Structure, leurs Paramètres et les Valeurs de leurs Composantes. Nous avons enfin présenté un exemple d'une telle ALID.

Afin de mettre à l'épreuve et de valider notre concept d'Aide Logicielle pour l'Interaction Dynamique, nous avons dû concevoir et développer entièrement des applications spécifiques, de manière à pouvoir réaliser des expériences sur la pertinence de nos aides.

Ce sont ces Applications et ces Expérimentations que nous allons maintenant présenter, à commencer par celles concernant les PID de Lancer et d'Attraper.

Chapitre 3

PID : Lancer et Attraper

3.1 Introduction

Nous ouvrirons ce chapitre par une justification des tâches basiques de lancer de balle et d'interception d'objets qui ont servi de cadre à l'évaluation de nos Aides Logicielles pour l'Interaction Dynamique. Nous décrirons ensuite la conception et le développement de l'application que nous avons utilisée. Puis, nous détaillerons l'expérimentation de lancer de balle que nous avons menée ainsi qu'une première étude sur l'interception d'objets en mouvement (en approche frontale). A l'issue de cette dernière, nous discuterons des résultats obtenus et nous proposerons notamment un protocole expérimental destiné à l'évaluation des ALID. Celui-ci sera utilisé dans la seconde expérience d'interception de balle (en approche oblique) que nous décrirons alors.

3.2 Justification

Nous avons isolé dans le chapitre précédent 3 Primitives d'Interaction Dynamique, et nous avons suggéré que toute interaction, même complexe, avec un objet en mouvement ou à mettre en mouvement, pouvait se réduire à une PID ou se décomposer en une suite de PID.

Il était donc naturel de commencer l'évaluation de l'apport de nos Aides Logicielles pour l'Interaction Dynamique en les mettant en oeuvre sur les interactions dynamiques les plus basiques qui soient : d'une part le lancer de balle vers une cible fixe en étant immobile, et d'autre part l'interception manuelle, en étant immobile, d'un objet en déplacement vers soi ("Frapper" étant la PID la plus difficile, elle a tout de suite été écartée). Nous considérons que ces deux tâches sont les plus simples de leurs PID respectives puisque le Sujet et la Cible pour l'une, l'Émetteur et le Sujet pour l'autre, sont Fixes (et nous avons vu que s'ils étaient Mobiles, le Sujet aurait

davantage de paramètres à prendre en compte). Le Geste est Discret pour l'interception d'objets et il est Continu en ce qui concerne le Lancer, mais il s'agit du geste de lancer le plus naturel de l'homme. Par ailleurs, ces tâches ont un but très simple, par opposition à des activités ayant un but complexe, c'est-à-dire impliquant des données supplémentaires à intégrer ou à gérer, comme dans le cas du jeu de fête foraine qui consiste à faire tomber en 3 boules un ensemble de boîtes de conserve empilées de façon pyramidale par exemple.

L'intérêt de la balle pour ces deux tâches réside dans sa forme sphérique : il n'y a pas à proprement parler de sens pour la lancer ni pour s'en saisir, les frottements de l'air lorsqu'elle est en vol s'appliquent de manière uniforme quelle que soit son orientation et ses rebonds en cas de contact sont faciles à prévoir du fait de sa rotondité.

Enfin, si nous avons choisi de remplacer la sphère par un cylindre dans une de nos expérimentations sur l'interception d'objets, c'est justement parce qu'il y a un sens dans lequel il est plus aisé de s'en saisir. Toutefois, il s'agit d'une forme simple dont on se sert tous les jours, elle est donc aussi naturelle pour l'homme que la sphère.

3.3 Conception et développement du système

3.3.1 Analyse et modélisation de l'interaction

Le contrôle des déplacements des segments corporels est réalisé grâce à l'utilisation des capteurs tactilo-proprio-kinesthésique situés dans les muscles, les articulations, les tendons (capteurs dits proprioceptifs) et dans la peau, et à ceux situés au niveau des systèmes visuel et vestibulaire.

Ils fournissent au système nerveux central un flux d'informations sensorielles qui va permettre de déterminer les positions et les vitesses relatives des différents segments corporels, ainsi que les forces qui leur sont appliquées, et la nature des contacts de la surface du corps avec le milieu extérieur.

Le système visuel est probablement celui qui fournit les informations les plus indispensables au contrôle des mouvements, même si certaines tâches peuvent être faites les yeux fermés et derrière le dos. C'est pourquoi de nombreux travaux de recherche ont porté sur la nature des informations fournies par le système visuel pour le contrôle des mouvements des membres supérieurs [Woo99]. Ainsi, plusieurs chercheurs ont mis en évidence les sources d'informations visuelles pertinentes ([AMK90] ; [Car81] ; [EM87]), le temps mis pour traiter ces informations ([Car81] ; [ZHK83]), la coordination et la précision spatiale du mouvement des yeux et de la main [AMK90].

3.3.1.1 Informations rétinienne et extra-rétinienne

Les *informations rétinienne* et les *informations extra-rétinienne* sont les deux types d'informations visuelles pouvant être utilisées dans le contrôle des membres.

- **les informations rétinienne :**

Elles comprennent les informations obtenues à partir des formes de stimulation de la rétine et relatives à la position et au mouvement du membre et de la cible. Elles jouent plusieurs rôles lors du *guidage des mouvements* des membres.

Avant l'exécution du mouvement, les informations relatives à la position d'une cible périphérique sont uniquement des informations rétinienne. Au cours du mouvement, les informations rétinienne peuvent suffire à fournir le *retour visuel* nécessaire à contrôler l'état du mouvement. Ce retour visuel ne comprend pas seulement les informations relatives à la position du membre en mouvement [Car81] mais aussi à la position de la cible, et à leur position relative [PPG86].

- **les informations extra-rétinienne :**

Elles sont relatives à la position des yeux jouent aussi un rôle important lors du guidage des membres. Plusieurs chercheurs ont montré que l'on peut guider un membre avec précision en direction d'une cible en utilisant uniquement des informations extra-rétinienne [HS85]. Cependant les informations extra-rétinienne ne peuvent être utilisées que pour localiser une cible périphérique pour un mouvement dirigé [PEKJ79].

3.3.1.2 Nos Lancer/Attraper et la Modélisation des Interactions Dynamiques

Par rapport à notre Modélisation des Interactions Dynamiques (Chapitre 2), le lancer naturel que nous allons étudier correspond à la situation $S_f G_c C_f$ (Sujet Fixe, Geste Continu, Cible Fixe) du lancer sur le cube du Lancer (voir figure 3.1).

L'action naturelle consistant à attraper une balle sur laquelle nous allons nous pencher correspond à la situation $E_f S_f G_d$ (Émetteur Fixe, Sujet Fixe, Geste Discret) sur le cube de l'Attraper. La figure 3.2 illustre cela.

3.3.2 Périphériques et technique d'interaction

Pour nos expérimentations sur le lancer de balle et l'interception d'objets en mouvement, nous voulions que l'interaction soit la plus naturelle et réaliste possible. Cette contrainte a d'entrée éliminé les techniques d'interaction à base de clavier-souris. Il n'était pas non plus envisa-

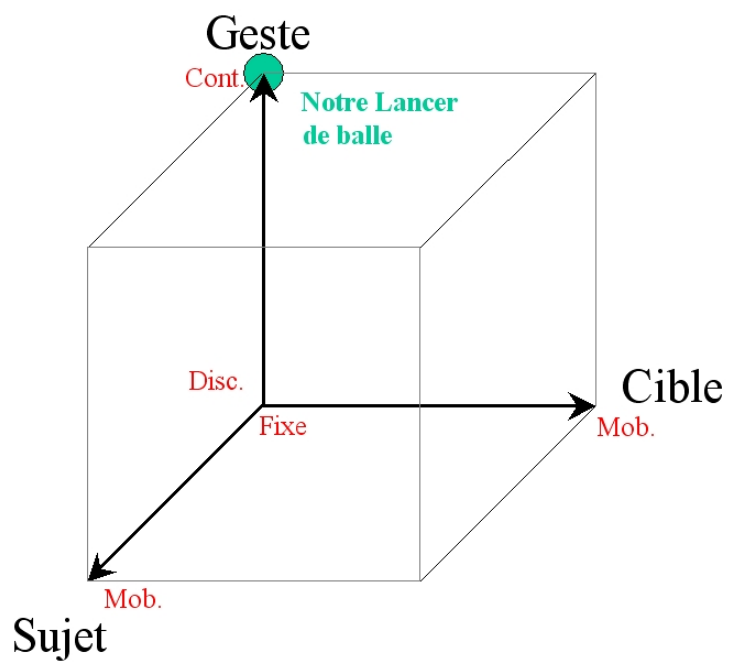


FIG. 3.1 – Notre lancer de balle situé sur le Cube de la Modélisation du Lancer

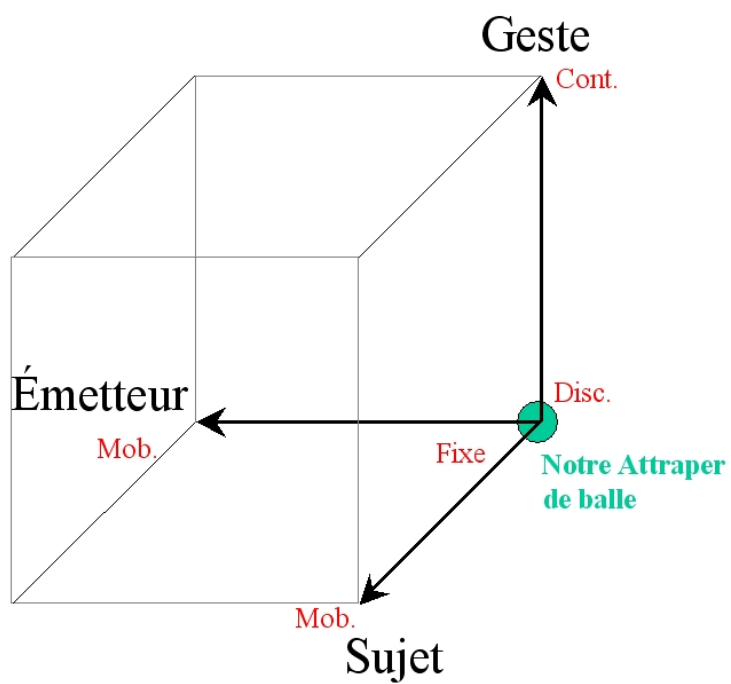


FIG. 3.2 – Notre Attraper de balle situé sur le Cube de la Modélisation de l'Attraper

geable d'utiliser un bras articulé de type Phantom. Il a donc été décidé d'employer un capteur de localisation électromagnétiques Polhemus pour traquer le déplacement de la main de l'utilisateur. Le système à fils SPIDAR était également envisagé, d'autant qu'il permettait de restituer un retour haptique, mais il n'était pas installé sur la plate-forme PREWISE au début de nos expérimentations.

Comme il fallait pouvoir détecter l'ouverture et la fermeture de la main, afin de connaître le moment où la balle était lâchée (dans le cas du Lancer) et calculer si l'objet était saisi (pour l'Attraper), un gant de données 5DT Glove a été utilisé afin de mesurer la valeur de flexion des doigts et nous permettre de calculer le degré d'ouverture de la main de l'utilisateur.

En ce qui concerne l'interface visuelle, nous avons opté pour un affichage sur un écran rétro-projeté de 2 x 2,5m (l'écran de la plate-forme PREWISE). Bien que cette interface apportât un problème d'offset entre la main de l'utilisateur et la main virtuelle affichée sur l'écran, elle fut préférée à l'utilisation d'un visiocasque, dont nous avons précédemment décrit les inconvénients (poids, faible résolution, étroitesse du champ de vision et recouvrement du monde réel). Par ailleurs, la stéréoscopie passive étant gérée par le système de projection de notre plate-forme, nous avons décidé de l'utiliser dans nos expérimentations. Il a été ajouté un second capteur de position, fixé sur la tête de l'utilisateur, afin qu'il bénéficiât du changement de position du point de vue apporté par la localisation de la tête.

3.3.3 Conception et développement logiciels

A la base, nous voulions développer un système réutilisable et facilement adaptable à nos besoins. L'application a donc été conçue avec un maximum d'éléments paramétrables, afin de pouvoir facilement concevoir de nouvelles expérimentations à partir du même environnement. Ce paramétrage s'effectue de manière simple et rapide, à l'aide de fichiers texte d'initialisation, sans avoir à éditer le code source du programme, de sorte qu'une personne ne sachant pas programmer puisse s'en servir. Le contenu de ces fichiers d'initialisation sera détaillé ci-après.

Le système a été développé en C/C++. Il a été fait appel à la librairie OpenGL pour la partie graphique, et à la librairie fmod pour la gestion des sons. Le simulateur étant principalement destiné à fonctionner dans un environnement muni du système d'exploitation Windows, du code MFC permet de créer la fenêtre et de gérer les messages et événements.

3.3.3.1 Machine à États

Application : Nous allons commencer par étudier le fonctionnement logiciel de l'application dans son ensemble :

- **Initialisation :**

La simulation débute par une phase classique d'Initialisation. Durant cette phase, hormis la création de la fenêtre et la configuration d'OpenGL, on parcourt les fichiers texte de configuration. A partir de là, on charge en mémoire les objets graphiques requis ainsi que leurs textures, on initialise les interfaces demandées (gants de données, capteurs de positions, SPIDAR) et on s'occupe des sons et de leur gestion.

– **Calibration :**

Il s'ensuit alors une phase de calibration, à la fois du gant de données et du capteur de position. Concernant le gant, il s'agit pour l'utilisateur d'effectuer, durant 5 secondes, des mouvements d'ouverture et de fermeture de la main afin de mesurer les valeurs seuil qui seront utilisées lors de la simulation pour contrôler l'état d'ouverture/fermeture de la main virtuelle. La calibration du capteur de position permet quant à elle de déterminer la position de la main de l'utilisateur qui servira de position de référence lors de la simulation.

En cas de problème avec cette première calibration obligatoire, cette formalité pourra être répétée. Lorsque la calibration du gant est terminée, on passe à la phase suivante.

– **Ok :**

On entre là dans une phase d'attente. On en profite généralement pour vérifier que les calibrations effectuées conviennent bien à l'utilisateur, notamment concernant l'ouverture et la fermeture de la main virtuelle, et si ce n'est pas le cas, on relance une phase calibration. Lorsque l'utilisateur est prêt, l'opérateur active le passage à la phase suivante par l'appui sur une touche du clavier.

A partir de là, le déroulement de la simulation diffère selon la Primitive étudiée.

Dans le cas du Lancer :

– **Début :**

Le projectile apparaît dans la simulation, à une position que nous appelons position initiale. Il est en suspension en l'air et n'est soumis à aucune force physique (aucune attraction, gravitation, etc.). Le seul moyen d'interagir avec lui est de s'en saisir. C'est pourquoi, durant cette phase, nous nous occupons de gérer la position et l'état de la main virtuelle. Lorsque l'utilisateur se saisit du projectile, on passe à la phase suivante.

– **Prise :**

Le projectile ayant été saisi, il est assujéti à l'état de la main virtuelle. On enregistre la position de la main virtuelle à chaque frame afin pouvoir calculer la vitesse et la direction initiale transmises au projectile au moment du lâcher. La méthode employée pour cela sera décrite plus loin. Lorsque la valeur d'ouverture de la main dépasse une certaine valeur seuil, on considère que le projectile est lâché par l'utilisateur, la vitesse et la direction

calculées lui sont transmises, et on passe à la phase suivante.

– **Vol :**

A partir du moment où le projectile a été relâché, il n'est plus assujéti à la main virtuelle. Une vitesse et une direction initiale lui ont été transmises, et il est dorénavant soumis aux forces extérieures telles que la gravité et éventuellement le vent. Nous effectuons les calculs de collisions entre le projectile et le sol en vérifiant simplement l'altitude du projectile. Lorsqu'il entre en collision avec le sol, il est immédiatement immobilisé. On calcul la distance entre le centre du projectile et le centre de la cible, puis cette phase prend fin 2 secondes plus tard. S'il s'agissait du dernier projectile à lancer, on passe à la phase Fin, sinon on retourne à la phase Début pour un autre lancer.

– **Fin :**

Il s'agit simplement d'une phase où l'on ne peut rien faire. Tous les lancers ont été effectués, le projectile ne réapparaît pas, l'expérience est terminée. On peut simplement continuer à déplacer la main virtuelle.

Dans le cas de l'Attraper :

– **Début :**

Le projectile apparaît dans la simulation, à une position que nous appelons position initiale. Il est en suspension en l'air et n'est soumis à aucune force physique (aucune attraction, gravitation, etc.). Il reste dans cet État Initial durant 5 secondes. Lorsque ce délai est écoulé, le projectile est envoyé vers l'utilisateur selon les paramètres de vitesse et direction initiales spécifiés dans le second fichiers de configuration. On passe dès lors à la phase suivante.

– **Vol :**

Le projectile a quitté son État Initial, il possède une vitesse et une direction, il est dorénavant soumis aux forces extérieures telles que la gravité et éventuellement le vent. Nous effectuons les calculs de collisions entre le projectile et le sol, entre le projectile et les murs et, surtout, entre le projectile et la main. Trois secondes après le commencement de cette phase, quel que soit l'état du projectile, celle-ci se termine. S'il s'agissait du dernier projectile à attraper, on passe à la phase Fin, sinon on retourne à la phase Début pour une autre tentative.

– **Fin :**

Il s'agit de la même phase de fin que pour le Lancer, aucune interaction n'est possible. Tous les projectiles ont été envoyés, l'expérience est terminée. On ne peut que déplacer la main virtuelle.

La figure 3.3 récapitule le déroulement de notre application.

La Main : Le processus complet concernant l'ouverture/fermeture de la main sera détaillé plus loin. Pour le moment, il suffit de savoir que la main possède 3 états distincts : "Ouvverte", "Saisie" et "Fermée". La main est constituée des 5 doigts, chacun possédant une valeur de pliure obtenue grâce au gant de données. On effectue la somme de ces valeurs, et on la confronte à deux valeurs seuils. On déduit du résultat de cette comparaison l'état de la main virtuelle.

La modélisation de la Main Virtuelle est visible sur la figure 3.4.

3.3.3.2 Ouverture/Fermeture de la Main Virtuelle

La valeur d'ouverture/fermeture de la main virtuelle est la somme des valeurs d'ouverture/fermeture des 5 doigts obtenues grâce au gant de données. Sachant qu'une valeur faible obtenue par le gant de données traduit un doigt complètement plié et qu'une valeur élevée signifie que le doigt est tendu, on peut donc écrire la propriété d'ouverture/fermeture de la main de la manière suivante :

– **Main complètement Ouvverte :**

$$Main_{Max} = P_{Max} + I_{Max} + M_{Max} + A_{Max} + O_{Max}$$

La valeur d'ouverture maximale de la main est égale à la somme des valeurs maximales d'ouverture des doigts.

– **Main complètement Fermée :**

$$Main_{min} = P_{min} + I_{min} + M_{min} + A_{min} + O_{min}$$

La valeur d'ouverture minimale de la main est égale à la somme des valeurs minimales d'ouverture des doigts.

– **Main à l'instant 't' :**

$$Main_t = P_t + I_t + M_t + A_t + O_t$$

La valeur d'ouverture de la main à l'instant 't' est égale à la somme des valeurs d'ouverture des doigts à l'instant 't'.

L'utilisateur ne peut toutefois pas aisément plier ou déplier ses doigts de manière à atteindre les valeurs maximales et minimales détectées par le gant. De ce fait, l'État d'ouverture/fermeture de la main virtuelle est caractérisé de la manière suivante :

– **Main Ouvverte, de 100 à 75 % :**

$$Main_t > Main_{min} + 0.75 * (Main_{Max} - Main_{min})$$

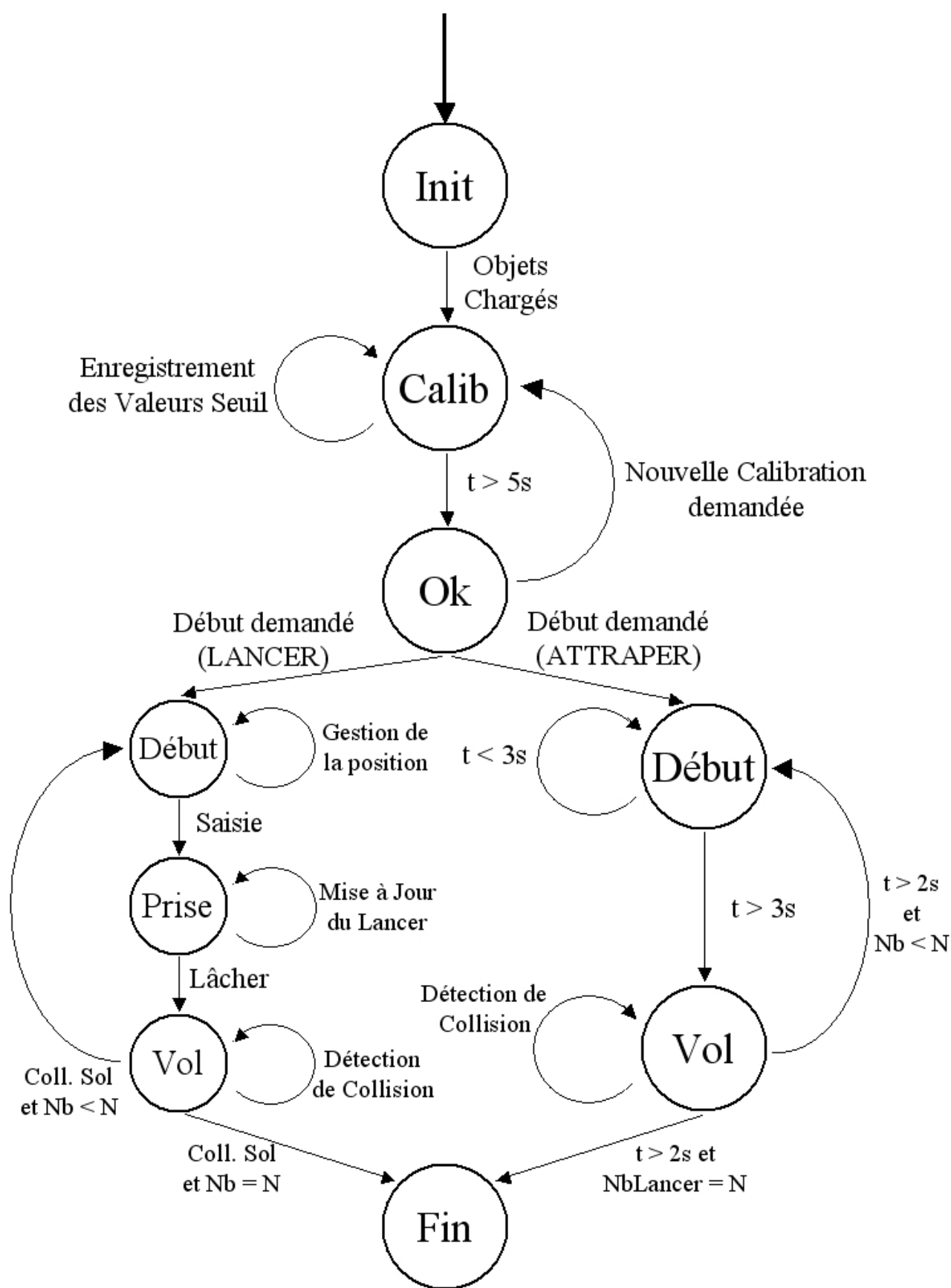


FIG. 3.3 – Diagramme d'États de l'Application

M_O : Main virtuelle Ouverte
 M_S : Main virtuelle Saisie
 M_F : Main virtuelle Fermée

S_1 : Valeur Seuil 75 %
 S_2 : Valeur Seuil 25 %

$M = \{p, i, m, a, o\}$
 p = pouce
 i = index
 m = majeur
 a = annulaire
 o = auriculaire

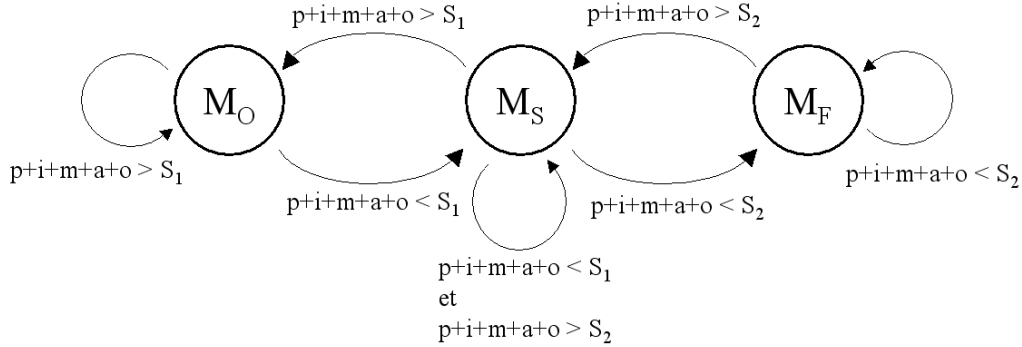


FIG. 3.4 – Diagramme d'États de la Main Virtuelle (Lancer-Attraper))

La valeur d'ouverture de la main à l'instant 't est au moins égale à 75 % de l'ouverture maximale de la main.

Si le projectile n'est pas saisi et qu'il entre en collision avec la main, il rebondira dessus et ne sera pas attrapé car la main est trop ouverte pour cela. Si en revanche le projectile était déjà saisi, il est relâché au moment où cet état est détecté, car la main est trop ouverte pour garder le projectile.

– **Main 'Saisie' de 75 à 25 % :**

$$Main_{min} + 0.25 * (Main_{Max} - Main_{min}) < Main_t < Main_{min} + 0.75 * (Main_{Max} - Main_{min})$$

La valeur d'ouverture de la main à l'instant 't est comprise entre 25 et 75 % de l'ouverture maximale de la main.

Si le projectile n'est pas saisi et qu'il entre en collision avec la main, on considère qu'il est attrapé par l'utilisateur. Si le projectile est déjà saisi, rien ne change.

– **Main Fermée de 25 à 0 % :**

$$Main_t < Main_{min} + 0.25 * (Main_{Max} - Main_{min})$$

La valeur d'ouverture de la main à l'instant 't est inférieure à 25 % de l'ouverture maximale de la main.

Si le projectile n'est pas saisi et qu'il entre en collision avec la main, il rebondira dessus et ne sera pas attrapé car la main (le poing) est trop fermée pour cela. Si le projectile est déjà saisi, rien ne se passe.

L'inconvénient de cette technique étant qu'il suffit de plier deux doigts au maximum pour que la main virtuelle soit considérée "Fermée". Il ne s'agit toutefois pas d'un inconvénient majeur puisque cette gestuelle n'est pas très naturelle et n'a pas lieu d'être mise en oeuvre durant notre simulation.

Le diagramme représenté par la figure 3.5 illustre ce fonctionnement.

3.3.3.3 Gestion du Lancer

Il a fallu implémenter les lois physiques de base afin de gérer la vitesse et les accélérations/décélérations des objets virtuels de manière à leur donner un comportement réaliste. Les objets possèdent donc des Caractéristiques Physiques (voir Chapitre 2, Modélisation des Primitives d'Interaction Dynamique) dont principalement la Masse, la Position du centre de gravité et le Coefficient de restitution pour les rebonds.

Afin de respecter l'esprit de souplesse avec lequel l'application a été conçue, chacun des paramètres cinématiques et physiques implémentés est modifiable. Ainsi la gravité a une valeur de base de 9.8 N/m^2 mais il est possible de totalement la supprimer en lui affectant la valeur zéro (tests en apesanteur). L'application respecte les équations fondamentales de la dynamique. Soient $\vec{V}_0 (0, V_{0y}, V_{0z})$ la vitesse initiale, $\vec{g} (0, -g, 0)$ la gravité, et $\vec{a}$ l'accélération, alors on a :

$$- a_x = 0 \Rightarrow V_x = V_{0x} = 0$$

$$- a_y = g \Rightarrow V_y = -gt + V_{0y}$$

$$- a_z = 0 \Rightarrow V_z = V_{0z} = \cos\theta V_0$$

Les coordonnées de l'objets sont donc :

$$- x = x_0$$

$$- y = -\frac{1}{2}Gt^2 + \sin\theta V_0 t + y_0$$

$$- z = \cos\theta V_0 t + z_0$$

3.3.3.4 Modèles 3D

Les grilles et les projectiles ont été modélisés avec de simples primitives OpenGL. En ce qui concerne la main virtuelle, elle a été créée avec le logiciel 3D Studio Max. Ensuite, la pliure de chaque doigt a été légèrement accrue, et le modèle ainsi obtenu sauvegardé. Cette opération a été renouvelée une dizaine de fois, afin d'obtenir 10 objets 3D de main, dont la pliure des doigts varie de "main tendue" à "poing fermé". Au total, ce sont 20 objets qui ont été créés et importés

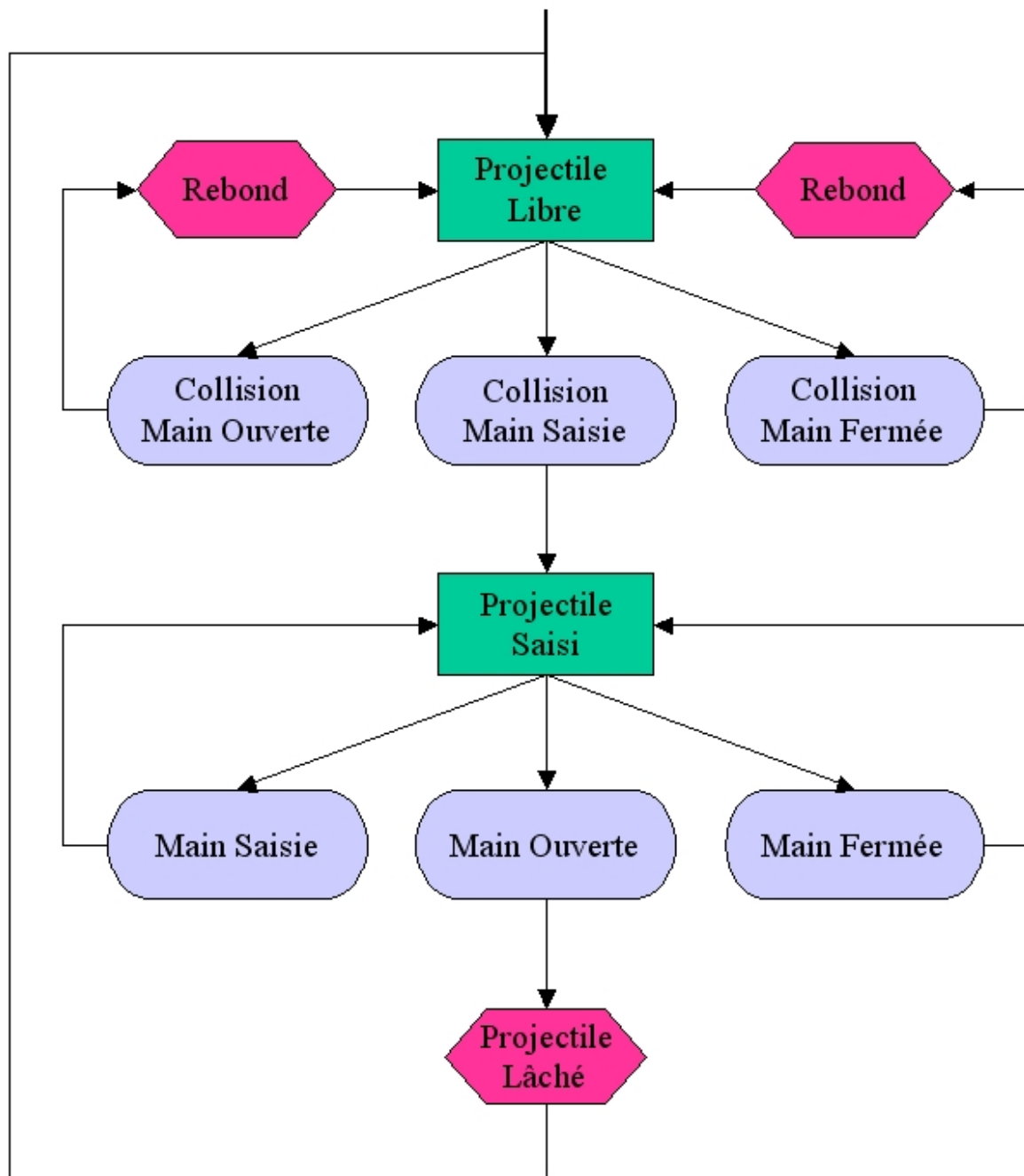


FIG. 3.5 – Diagramme de la Gestion du Projectile

dans l'application : 10 mains droites et 10 mains gauches.

Les 10 objets 3D de main gauche sont représentés sur la figure 3.6.

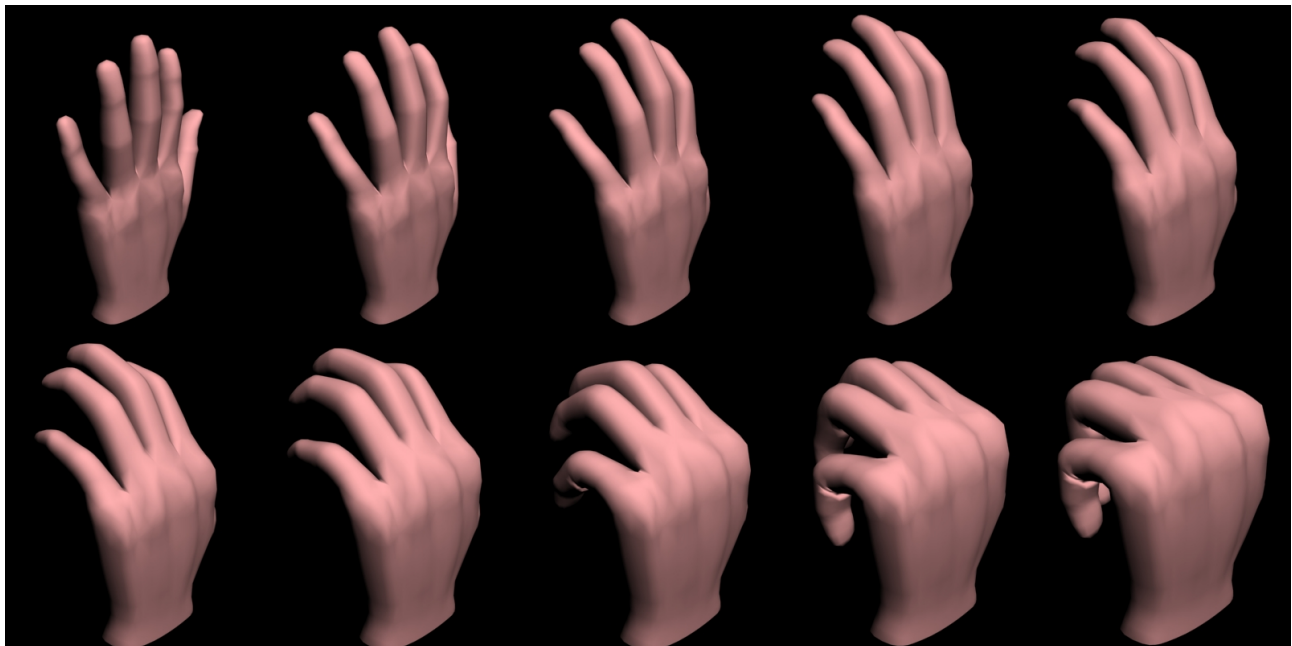


FIG. 3.6 – Les 10 modèles de main

3.4 Expérimentations

3.4.1 Lancer de balle sur une cible fixe

3.4.1.1 Objectif de l'expérience

Le but de cette expérience était d'étudier l'impact de l'affichage d'une trajectoire parabolique terminant au centre de la cible et de la trajectoire simultanée sur les performances de l'utilisateur lors d'une tâche de lancer de balle sur une cible fixe. Pour cela, nous enregistrons pour chaque lancer la distance entre la position finale de la balle et le centre de la cible (distance générale, mais aussi la distance latérale [droite-gauche] et la distance en profondeur [devant-derrière]), ainsi que le temps de visée du lancer.

3.4.1.2 Description de la tâche

Les sujets avaient pour simple consigne d'essayer d'atteindre le centre de la cible avec chaque balle. Le sujet devait commencer par s'équiper des interfaces décrites ci-dessus et placer

sa main devant le capteur magnétique à environ 15 cm (de manière à rester dans le champ du capteur durant le lancer). Il signalait alors qu'il était prêt et l'application était lancée, signifiant le début de l'expérience.

3.4.1.3 Environnement virtuel

L'enceinte de l'environnement virtuel était composée de grilles. Celle du sol était jaune, de manière à bien la différencier des grilles murales, dessinées en bleu. La cible était représentée sur le sol par des cercles concentriques de couleur verte, bleue, rouge et jaune. Enfin, on affichait la main virtuelle, la balle à lancer et, le cas échéant, la ou les aides logicielles actives. La figure 3.7) est une capture d'écran de la simulation de Lancer sans Aide Logicielle activée.

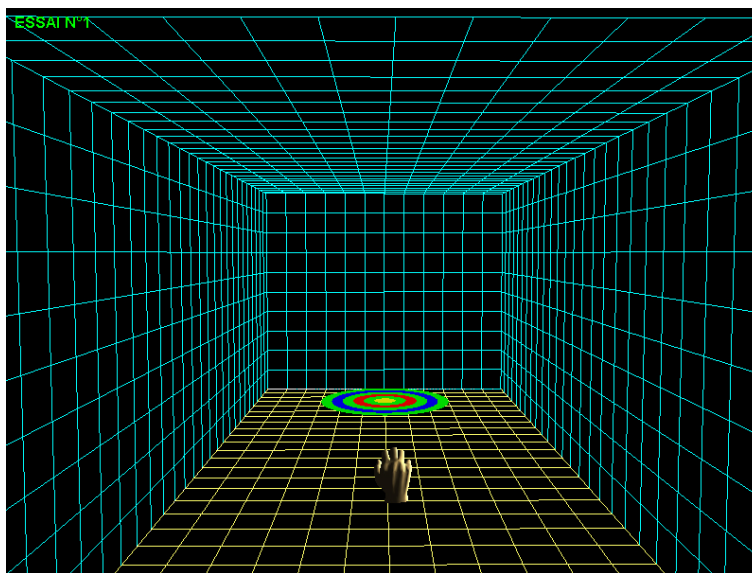


FIG. 3.7 – Capture d'écran de la simulation de Lancer

3.4.1.4 ALID n°1 : TrajectoireParabolique

L'idée est ici de déterminer, d'une manière empirique ou à l'aide de calculs physiques, les paramètres d'un "lancer de référence" aboutissant à un résultat parfait, c'est à dire que la balle atterrit au centre de la cible. Pour que ce lancé de référence soit acceptable, il faut qu'il soit ensuite reproductible par l'utilisateur, c'est à dire que l'utilisateur doit être capable de lancer la balle avec les mêmes paramètres initiaux que ceux du lancer de référence (mêmes position, vitesse et direction initiales).

On effectue alors ce "lancer de référence" et on enregistre les positions de la balle entre le

moment où elle est lâchée et l'instant où elle entre en collision avec le sol. On utilise ensuite ces positions pour tracer une courbe tri-dimensionnelle correspondant à la trajectoire du lancer de référence.

Le tableau 3.1 représente la structure de l'ALID TrajectoireParabolique.

Propriété	Valeur
Nom	TrajectoireParabolique
But	"Apprentissage"
PID	"Lancer"
Type	"Visuel"
Description	Cette aide consiste en l'affichage d'une courbe en 3 dimensions, composée de segments de ligne reliant deux positions successives de la balle lors d'un lancer aboutissant à un résultat parfait.
Genre	"Statique"
Pré-condition	aucune
Post-condition	aucune
État	État_TrajectoireI
Fonctionnement	aucune

TAB. 3.1 – Structure de l'ALID "TrajectoireParabolique".

Les Paramètres de l'ALID "TrajectoireParabolique" sont :

- **PID** → **Modèle PID** : $S_f G_c C_f$.
- **Type** → **Sous-Type** : Unique Objet 3D.
- **État** → **État_TrajectoireI** : Soit P_{lancer} la position de la balle au moment où elle est lâchée selon la trajectoire idéale pré-déterminée dans la simulation, alors on a :

$$P_{TrajectoireI} = P_{lancer}$$

3.4.1.5 ALID n°2 : TrajectoireSimultanée

L'Objectif de cette Aide est de mettre en relief la façon dont se déplace l'Objet avec lequel l'utilisateur interagit. Pour cela, nous allons afficher une courbe 3 dimensions, telle la traînée d'une comète, permettant de visualiser les dernières positions calculées de l'objet. Cette Aide est donc très liée au déplacement de l'Objet en mouvement sur lequel l'utilisateur porte son attention, et ce quelle que soit l'interaction mise en oeuvre. De ce fait, la particularité de cette ALID est qu'elle peut être employée pour tous les types de PID.

Le tableau 3.2 représente la structure de l'ALID TrajectoireSimultanée.

Propriété	Valeur
Nom	TrajectoireSimultanée
But	"Apprentissage"
PID	"Lancer" ; "Attraper" ; "Frapper"
Type	"Visuel"
Description	Cette aide consiste en l'affichage d'une courbe en 3 dimensions, composée de segments de ligne reliant deux positions successives de l'Objet en mouvement principal durant l'interaction en cours.
Genre	"Dynamique"
Pré-condition	aucune
Post-condition	aucune
État	État_TrajectoireS
Fonctionnement	aucune

TAB. 3.2 – Structure de l'ALID "TrajectoireSimultanée".

Les Paramètres de l'ALID "TrajectoireSimultanée" sont :

- **Type** → **Sous-Type** : Unique Objet 3D.
- **État** → **État_TrajectoireS** : On peut noter la Position de l'Aide Logicielle comme étant celle du dernier point utilisé pour la tracer. On a ainsi :

$$P_{TrajectoireS} = P_{Objet}$$

3.4.1.6 Protocole expérimental

Dans cette expérience, le sujet doit lancer quatre séries de dix balles, suivant quatre conditions, sur une cible située sur le sol de l'environnement à une distance de 7,5 mètres de la main. Les quatre conditions sont :

- Condition 1 (notée C1) : Pas d'affichage de trajectoire.
- Condition 2 (C2) : Avec affichage de la trajectoire idéale (ALID TrajectoireParabolique).
- Condition 3 (C3) : Avec affichage de la trajectoire simultanée (ALID TrajectoireSimultanée).
- Condition 4 (C4) : Avec affichage de la trajectoire idéale et de la trajectoire simultanée (ALID TrajectoireParabolique et TrajectoireSimultanée).

En plus de l'équipement déjà présenté, le sujet place dans sa main une balle en polystyrène expansé munie d'un velcro afin qu'elle reste accrochée au gant même lorsque la main est ouverte (voir la figure 3.8).



FIG. 3.8 – Dispositif de la balle fixée au gant de données

12 sujets ont participé à l'expérience, mais suite à un problème informatique, les données relative au premier d'entre eux ont été perdues. Il a donc été supprimé et l'on a au final 11 participants.

Afin d'éviter un transfert d'apprentissage entre les conditions, les sujets n'ont pas réalisé les 4 séries de lancer dans le même ordre. Le tableau 3.3 indique l'ordre des conditions par sujet.

3.4.1.7 Analyse des résultats

Afin d'évaluer la performance de l'utilisateur, nous avons enregistré les données suivantes : la distance entre la position finale de la balle et le centre de la cible (en centimètre), la distance latérale (droite-gauche, sur l'axe X) et la distance en profondeur (devant-derrrière, sur l'axe Z), toujours entre la position finale de la balle et le centre de la cible (en centimètre), et enfin le temps de visée du lancer (le temps écoulé entre la capture de la balle et son lâcher).

Ces données ont été analysée à l'aide de la méthode dite de "l'Analyse de Variance" (ANOVA, voir en Annexe).

Sujet	Ordre des Conditions			
Sujet 1	C1	C3	C2	C4
Sujet 2	C1	C4	C2	C3
Sujet 3	C2	C1	C4	C3
Sujet 4	C2	C3	C4	C1
Sujet 5	C2	C4	C3	C1
Sujet 6	C3	C2	C1	C4
Sujet 7	C3	C1	C4	C2
Sujet 8	C3	C4	C1	C2
Sujet 9	C4	C2	C1	C3
Sujet 10	C4	C3	C2	C1
Sujet 11	C4	C1	C3	C2

TAB. 3.3 – Ordre des Conditions par Sujet (Lancer).

Distance Générale : Aucun effet significatif des Aides Logicielles présentées n’a été révélé par l’ANOVA sur les performances de l’utilisateur ($F = 0.45$, $p = 0.7193$).

Les résultats montrent que toutes les conditions convergent vers des distances finales statistiquement équivalents (figure 3.9). Les distances Balle-Cible moyennes (en centimètres) enregistrées sont d’environ 238,5 (Std : 121,3) pour la condition 1, de 235,5 (Std : 119,8) pour la condition 2, de 217,7 (Std : 82,8) pour la condition 3, et de 276,1 (Std : 152,0) pour la condition 4.

Distance Latérale : Il n’y a pas non plus d’effet significatif des Aides Logicielles sur l’erreur latérale commise par les utilisateurs ($F = 1.16$, $p = 0.3378$).

Les erreurs moyennes en centimètres sur l’axe X sont d’environ 6,5 (Std : 17,3) pour la condition 1, de 18,1 (Std : 10,3) pour la condition 2, de 3,1 (Std : 30,2) pour la condition 3, et de 3,064 (Std : 24,6) pour la condition 4.

Ces résultats sont illustrés par la figure 3.10.

Distance en Profondeur : En ce qui concerne l’erreur réalisée sur l’axe Z, elle est bien entendu plus importante que celle commise sur l’axe X, mais aucune différence statistique n’est à relever entre les conditions ($F = 0.39$, $p = 0.7597$).

Ainsi, comme montré par la figure 3.11, les erreurs moyennes obtenues en profondeur (et en centimètres) sont de 93,4 (Std : 115,9) pour la condition 1, de 103,1 (Std : 122,5) pour la condition 2, de 60,6 (Std : 153,5) pour la condition 3, et de 113,4 (Std : 82,2) pour la condition 4.

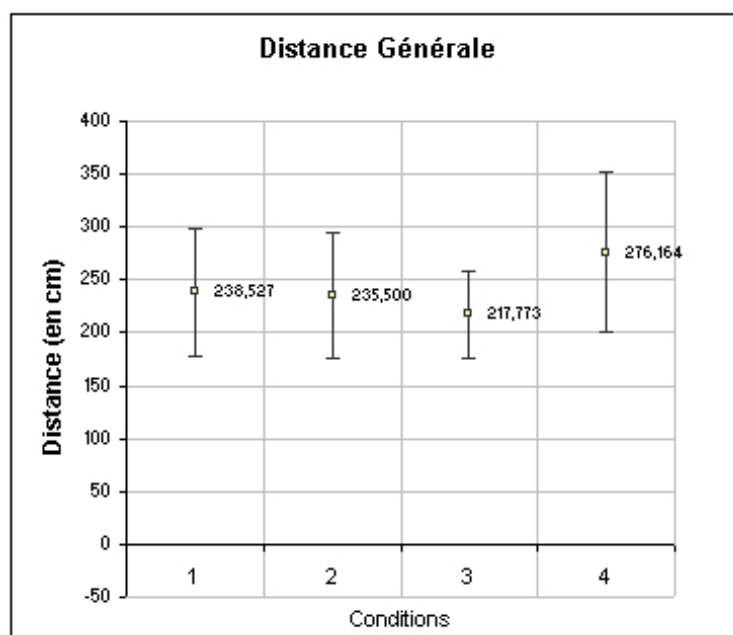


FIG. 3.9 – Distance moyenne par Condition

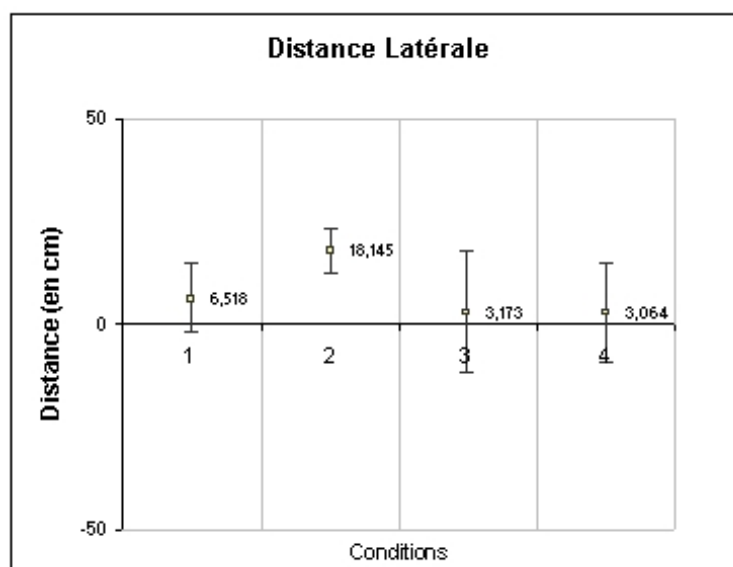


FIG. 3.10 – Distance Latérale par Condition

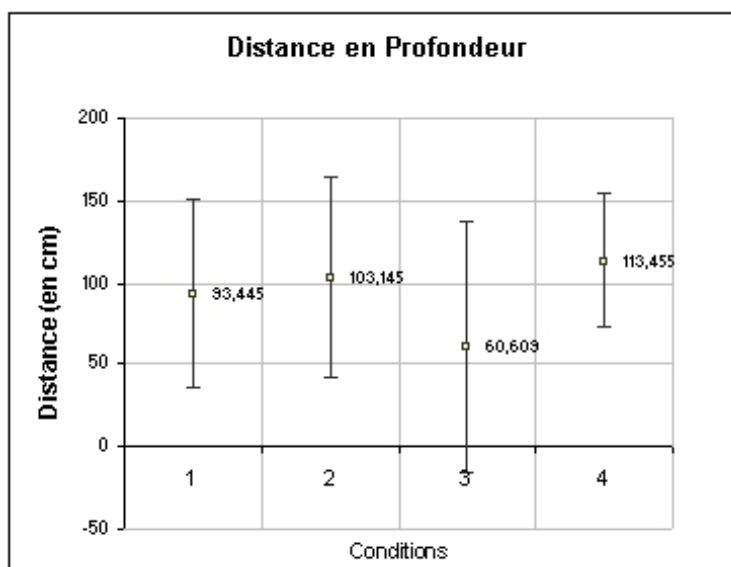


FIG. 3.11 – Distance en Profondeur moyenne par Condition

Temps de Visée : Enfin, le temps de visée pris par l'utilisateur n'est pas non plus statistiquement affecté par la présence ou non d'Aides Logicielles ($F = 0.63$, $p = 0.6012$).

La figure 3.12 nous montre ainsi que le temps de visée moyen en seconde est de 2,2 (Std : 0,5) pour la condition 1, de 2,3 (Std : 0,8) pour la condition 2, de 2,5 (Std : 0,8) pour la condition 3, et de 2,7 (Std : 0,9) pour la condition 4.

Au final, aucun effet ne ressort de l'utilisation des ALID ici mises en oeuvre. Néanmoins, plusieurs critiques vis à vis de la simulation en elle-même et à l'égard du protocole expérimental utilisé sont à émettre. Le même protocole ayant été utilisé pour l'expérience sur l'Attraper d'objets en approche frontale, nous réservons ces critiques pour l'analyse des résultats de cette seconde expérience.

En ce qui concerne les ALID "TrajectoireParabolique" et "TrajectoireSimultanée", il serait probablement utile de les visualiser avec différents angles de caméra, afin de mieux les percevoir dans l'espace, et d'autant plus l'une par rapport à l'autre si elles sont toutes les deux actives en même temps.

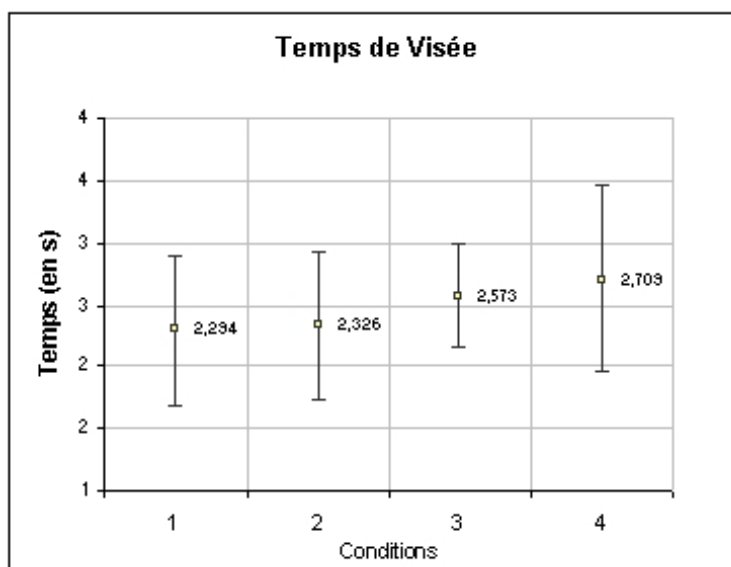


FIG. 3.12 – Temps de Visée moyen par Condition

3.4.2 Attraper d'objets en approche frontale

3.4.2.1 Objectif de l'expérience

L'objectif de cette première expérience d'attraper d'objets était de déterminer l'intérêt et l'efficacité de l'affichage de la trajectoire simultanée d'un objet en tant qu'aide logicielle pour l'interception d'un objet en mouvement en approche frontale. Nous voulions également étudier l'effet de l'affichage d'une grille comme indice visuel pour la perception de la profondeur.

3.4.2.2 Description de la tâche

Cette expérience se divisait en deux parties : interception d'une balle, puis interception d'un cylindre. Dans les deux cas, la tâche demandée au sujet était d'intercepter et d'attraper au vol l'objet lancé vers lui. Toutefois, il a été spécifié dans le cas du cylindre, en rotation sur lui-même durant le vol, d'essayer de le saisir lorsqu'il était en position verticale. Cette configuration ne survenait qu'à deux moments précis : lorsqu'il était dans le plan et 20 cm en avant du plan. L'application étant développée à l'échelle 1, une distance de 20 cm dans l'application correspondait par conséquent à une distance de 20 cm dans le réel. Toutefois, il n'y avait aucune contrainte supplémentaire dans la gestion du cylindre (trajectoire et saisie) par la simulation par rapport à la balle.

Comme précédemment, le sujet devait commencer par s'équiper des interfaces et par effectuer la calibration du capteur de position, puis signifiait son apprêtement, autorisant le lancement de

l'application et le début de l'expérience.

3.4.2.3 Environnement virtuel

L'environnement virtuel durant cette expérimentation était sensiblement le même que pour la précédente, la cible en moins. C'est à dire que l'on retrouve les mêmes grilles bleues et jaunes comme enceinte lorsqu'elles étaient activées, ainsi que la main virtuelle, la balle ou le cylindre à attraper et, le cas échéant, la TrajectoireSimultanée du projectile.

3.4.2.4 Protocole expérimental

Au total, 8 sujets volontaires ont participé à cette expérience. Chaque sujet a réalisé, chacun dans un ordre différent, les tâches décrites ci-dessus dans les conditions suivantes :

- Noir sans aide : dans le noir, sans trajectoire (Condition 1)
- Noir avec Trajectoire : dans le noir, avec affichage de la TrajectoireSimultanée (C2)
- Grille sans aide : avec affichage des grilles, sans trajectoire (C3)
- Grille avec Trajectoire : Avec affichage des grilles et de la TrajectoireSimultanée (C4)

Les figures 3.13 a,b,c et d sont respectivement des captures d'écran de la simulation dans les conditions 1, 2, 3, 4.

La différence d'ordre des conditions s'expliquait là encore par la volonté d'éviter le transfert d'apprentissage entre celles-ci. L'ordre utilisé (pour la balle et le cylindre) est indiqué dans le tableau 3.4.

Sujet	Ordre des Conditions			
Sujet 1	C2	C4	C3	C1
Sujet 2	C1	C2	C3	C4
Sujet 3	C3	C2	C1	C4
Sujet 4	C2	C1	C3	C4
Sujet 5	C1	C2	C4	C3
Sujet 6	C4	C3	C1	C2
Sujet 7	C4	C1	C3	C2
Sujet 8	C3	C4	C2	C1

TAB. 3.4 – Ordre des Conditions par Sujet (Attraper).

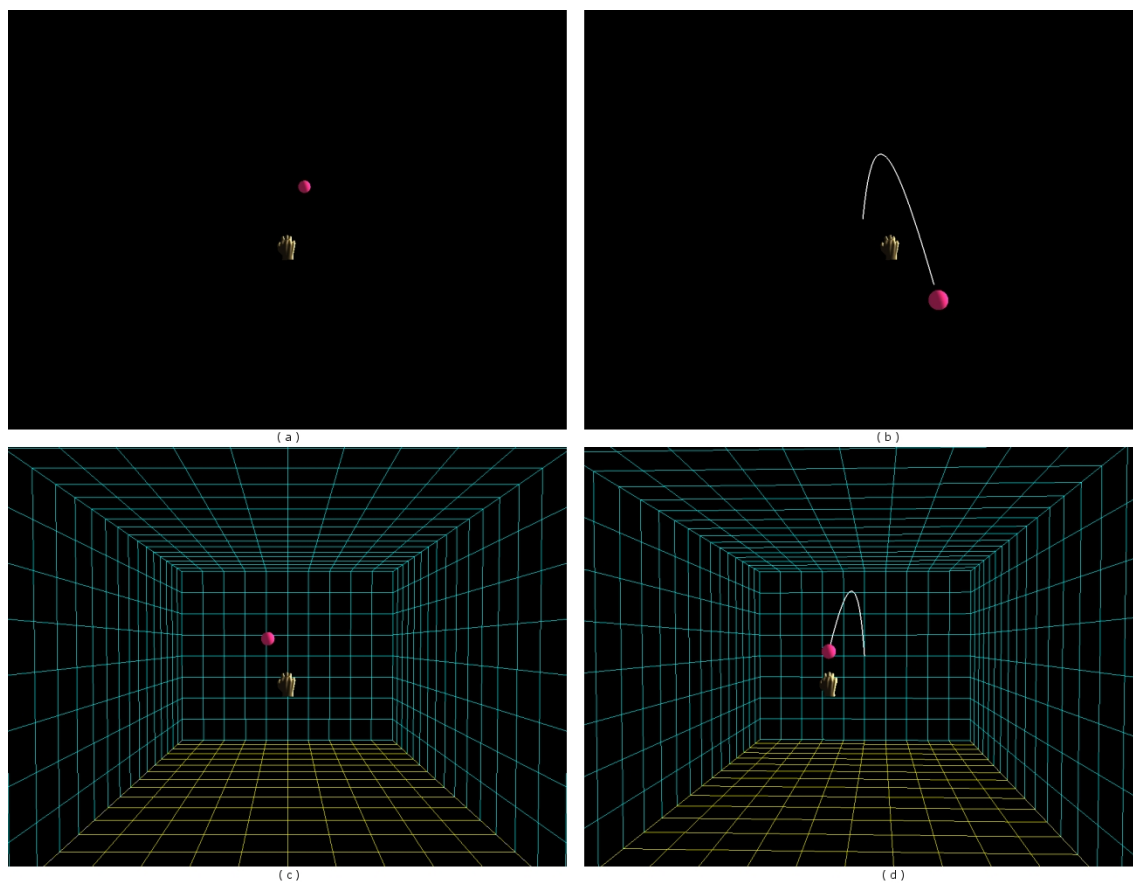


FIG. 3.13 – Capture d'écran de la simulation d'Attraper dans les Conditions 1 à 4

Chacun des deux projectiles (balle, cylindre) était envoyé 20 fois dans chaque condition, ce qui fait un total de 160 interceptions à réaliser par sujet (80 balles et 80 cylindres). Un projectile était envoyé toutes les 8 secondes. Il était positionné à son point de départ durant 5 secondes puis était lancé et restait en mouvement pendant 3 secondes. À peu près 1 seconde lui était nécessaire pour atteindre le plan où se situait la main virtuelle de l'utilisateur. Le projectile arrivait dans le plan sur l'un des quatre points pré-définis illustrés sur la figure 3.14, dans un ordre préalablement déterminé de manière à paraître aléatoire et dont le sujet n'avait pas connaissance.

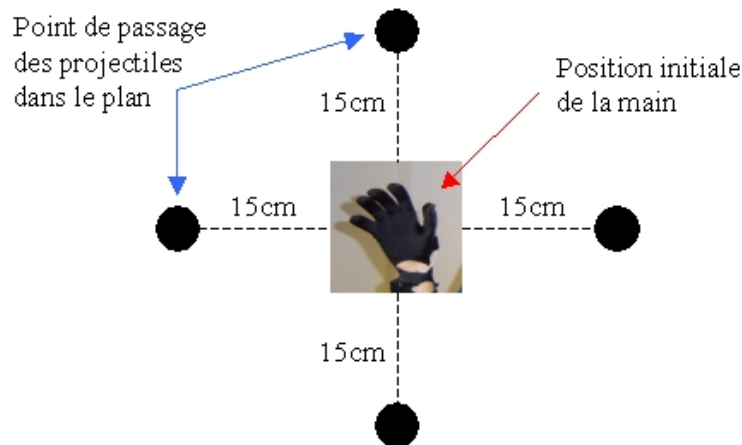


FIG. 3.14 – Points de passage du projectile dans le plan vertical de la Main

3.4.2.5 Analyse des résultats

Nous avons, pour chaque essai, enregistré la réussite ou l'échec de la tentative afin de pouvoir calculer le taux de balles et de cylindres attrapés (en pourcentage) selon les conditions expérimentales. Nous nous sommes également intéressés à la distance (en centimètres) entre le projectile (balle ou cylindre) et la main virtuelle lorsque celle-ci était fermée par l'utilisateur. Ces données ont, comme précédemment, été soumises à l'ANOVA.

Taux de Balles Attrapées : Aucun effet significatif de l'Aide Logicielle ou de la présence des grilles n'est relevé par l'ANOVA ($F = 0.39$, $p = 0.7632$).

La proportion de balles attrapées est sensiblement la même quelque soit la condition expérimentale (figure 3.15). Le pourcentage moyen de balles saisies est d'environ 43,1 (Std : 29,0) pour la condition 1, de 44,3 (Std : 25,2) pour la condition 2, de 38,7 (Std : 25,0) pour la condition 3,

et de 32,5 (Std : 16,4) pour la condition 4.

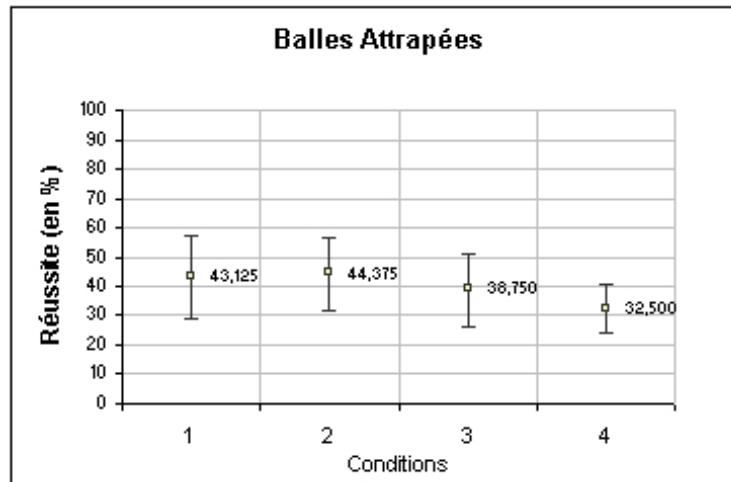


FIG. 3.15 – Taux de Balles Attrapées moyen par Condition

Taux de Cylindres Attrapés : On retrouve sensiblement les mêmes résultats pour les cylindres que pour les balles, et donc aucune différence statistique ($F = 0.62$, $p = 0.6060$).

En moyenne, le pourcentage de cylindres attrapés est de 46,2 (Std : 19,5) pour la condition 1, de 46,2 (Std : 19,4) pour la condition 2, de 38,1 (Std : 16,2) pour la condition 3, et de 36,8 (Std : 17,3) pour la condition 4.

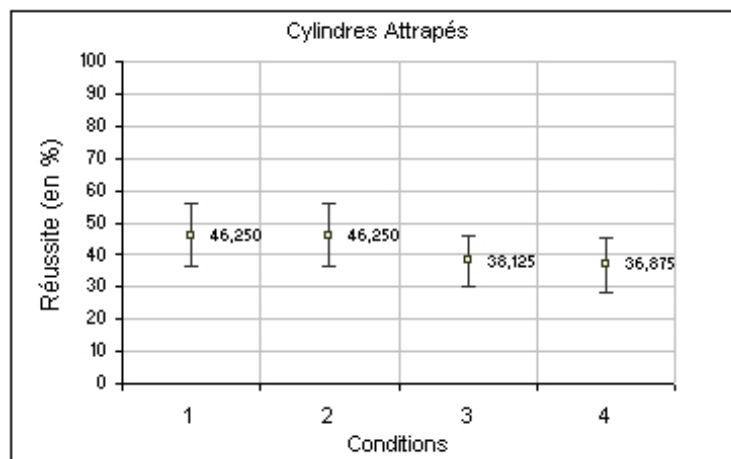


FIG. 3.16 – Taux de Cylindres Attrapés moyen par Condition

Distance Main Fermée - Balle : Il n'y a aucun effet remarquable non plus concernant la distance entre la balle et la main au moment où cette dernière est fermée ($F = 0.59$, $p = 0.6243$). La distance Main Fermée - Balle moyenne est de 78,5 (Std : 58,0) pour la condition 1, de 55,5 (Std : 49,1) pour la condition 2, de 54,1 (Std : 41,9) pour la condition 3, et de 96,4 (Std : 120,5) pour la condition 4.

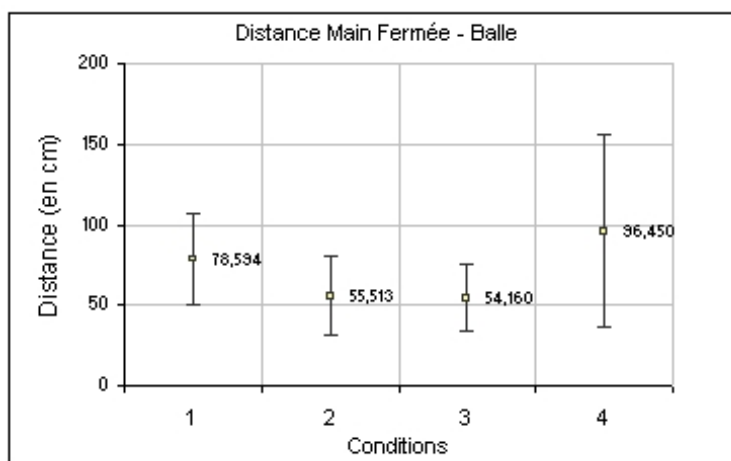


FIG. 3.17 – Distance Main Fermée - Balle moyenne par Condition

Distance Main Fermée - Cylindre : Enfin, aucun effet à noter concernant la distance entre le cylindre et la main au moment de sa fermeture ($F = 0.34$, $p = 0.7972$).

La distance Main Fermée - Cylindre moyenne est de 83,4 (Std : 130,5) pour la condition 1, de 68,0 (Std : 59,9) pour la condition 2, de 117,2 (Std : 192,3) pour la condition 3, et de 59,6 (Std : 59,2) pour la condition 4.

Comme pour l'expérience précédente, aucun effet ne ressort de l'utilisation de l'ALID mise en oeuvre, pas plus que de celle des grilles comme indice visuel de profondeur.

3.4.3 Discussion

Concernant l'AL TrajectoireSimultanée, il est à noter que l'utilisation qui en a été faite ici n'est probablement pas la meilleure. En effet, l'objet étant en approche frontale, et malgré le tracking de tête utilisé, la trajectoire affichée n'était pas très visible. Le manque de perception de cette aide n'aura alors certainement pas permis une grande compréhension de la trajectoire

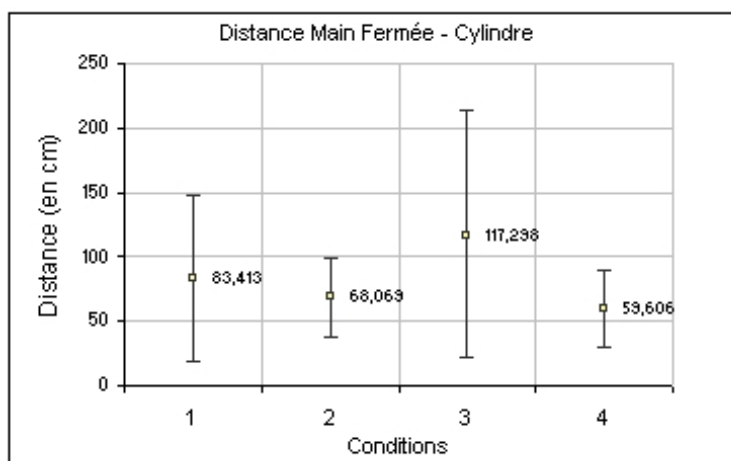


FIG. 3.18 – Distance Main Fermée - Cylindre moyenne par Condition

de l'objet, et par là même, à rendre efficace cette Aide.

Quant à l'utilisation des grilles, si elle permet de donner une impression visuelle de profondeur, elle ne permet pas à l'utilisateur, dans notre simulation, d'approximer la position de l'objet en mouvement. Il est possible en revanche que leur présence puisse renforcer la perception de la position de l'objet issue de la présence d'une tâche d'ombre de l'objet sur le sol.

Toutefois, il s'agit là de problèmes que l'on pourrait qualifier de mineurs, eu égard aux deux problèmes plus importants, concernant la simulation d'une part et le protocole expérimental d'autre part, qui sont à relever.

Comme il a été dit dans le premier chapitre de cette Thèse, lorsque l'on tente de recréer en réalité virtuelle une situation réelle, on utilise un modèle de la réalité qui ne peut être qu'une approximation de celle-ci, ce qui aboutit pour l'utilisateur placé dans la simulation à une dégradation, voire une perte, des informations qu'il possède par rapport à la même situation en réel. Nous proposons donc d'implémenter des Aides Logicielles, c'est à dire d'ajouter des informations dont l'utilisateur ne dispose pas dans le réel, pour contre-balancer la perte sus-dite.

Or, durant les expérimentations réalisées, nous avons encore d'avantage dégradé un environnement virtuel qui l'était déjà par rapport au réel, en proposant un environnement complètement noir, ou composé de simples grilles, dans lequel l'illumination était réduite au minimum, et où les ombres des objets sur le sol, pourtant bien implémentées, ont été désactivées.

L'idée était d'évaluer les ALID seules, sans que la présence des ombres notamment ne puisse introduire un biais dans l'évaluation. Mais en agissant de la sorte, nous avons supprimé un indice visuel de localisation des objets important pour l'utilisateur. Cet indice était d'autant plus

crucial que, l'environnement étant parfaitement dépouillé, il était impossible de déterminer la position ou la taille d'un objet en le comparant à d'autres objets.

Il n'a donc pas été pertinent de supprimer volontairement des informations naturellement présentes dans le réel, et dont doit disposer l'utilisateur, pour évaluer l'efficacité d'autres indices artificiellement ajoutés. Il aurait ainsi mieux valu proposer le meilleur environnement virtuel dont nous disposions, et évaluer l'intérêt de nos aides logicielles dans ces conditions.

Par ailleurs, le protocole expérimental, et plus précisément l'utilisation du système dit du "carré latin" spécifiant l'ordre des 4 conditions pour chaque sujet, n'était pas du tout adapté. En effet, l'intérêt de cette technique est qu'elle permet d'annihiler l'effet du transfert d'apprentissage entre les conditions. Or, toutes nos ALID ici évaluées ayant comme But l'"Apprentissage" de la tâche par l'utilisateur, c'est justement cette donnée qui nous aurait le plus permis de statuer sur la pertinence de l'implémentation de nos ALID.

Si en revanche nos Aides avaient été conçues dans le but de faciliter la "Réalisation" de la tâche, alors le protocole employé aurait été adapté et nous aurait permis de prendre connaissance de l'effet de nos Aides sur la performance de l'utilisateur, et ce sans que l'effet de l'apprentissage entre les conditions ne viennent biaiser nos résultats.

Groupe	1ère Série	2nde Série
Groupe 1	C1	C1
Groupe 2	C2	C1
...	...	...
Groupe i	Ci	C1
...	...	...
Groupe n	Cn	C1

TAB. 3.5 – Suggestion de Protocole pour l'évaluation de nos ALID.

Dans notre cas, il aurait été plus intéressant de créer des groupes d'utilisateurs. Chaque groupe aurait fait une première série de balles, chacun dans une condition expérimental différente, avant de faire une seconde série, tous avec la même condition de référence, c'est à dire sans aide. Ce protocole que nous proposons est résumé dans le tableau 3.5. Appliqué à notre expérience sur le Lancer de balle, on aurait eu le protocole illustré par le tableau 3.6.

C'est ce type de protocole expérimental que nous utiliserons dorénavant pour l'évaluation de nos Aides Logicielles pour l'Interaction Dynamique, même s'il a l'inconvénient de nécessiter un nombre plus élevé de sujets pour un même nombre de Conditions à tester que le Carré

Groupe	1ère Série	2nde Série
Groupe 1	C1 (aucune aide)	C1 (aucune aide)
Groupe 2	C2 (TrajectoireParabolique)	C1 (aucune aide)
Groupe 3	C3 (TrajectoireSimultanée)	C1 (aucune aide)
Groupe 4	C4 (TrajectoireParabolique et TrajectoireSimultanée)	C1 (aucune aide)

TAB. 3.6 – Suggestion de Protocole pour le Lancer.

Latin.

3.4.4 Attraper de balle en approche oblique

Le SPIDAR a été installé sur la plate-forme PREWISE de l'ISTIA après la fin de l'expérience précédente. Nous avons alors profité de la nouvelle disponibilité de cette interface pour la substituer aux capteurs de localisation Polhemus Patriot dans notre simulation. En effet, le SPIDAR nous apporte une plus grande précision dans la mesure de la position de la main de l'utilisateur, et nous permet de fournir un retour d'effort au niveau de la main au moment du contact avec la balle.

Pour cette expérience, nous avons opté pour un affichage monoscopique. En effet, la balle arrivant en biais, et non plus de face, et atteignant plus rapidement l'utilisateur, nous avons jugé inutile l'usage du tracking de tête et de la stéréoscopie. Par ailleurs, cette dernière nécessite un ajustement précis de paramètres (écartement des yeux, point de convergence) pour être intéressante, et nous ne voulions pas soumettre nos sujets à un temps de préparation trop long.

3.4.4.1 Objectif de l'expérience

Cette seconde expérience sur l'attraper de balle avait pour objectif la validation du protocole suggéré après le manque de réussite observé lors de la première expérimentation. Nous voulions également proposer de nouvelles ALID et les évaluer en tenant compte des réflexions portant sur la simulation formulées à l'issue des deux premières expériences.

Nous avons par ailleurs émis l'hypothèse que la durée pendant laquelle une aide logicielle était activée pouvait influencer sur son But même (Apprentissage ou Réalisation de la tâche), et ce du fait de la charge mentale qui en découlait. Nous souhaitons donc pouvoir vérifier la validité de cette théorie. Pour cela, nous mesurons la distance entre la main virtuelle et la balle au moment où cette dernière passe dans le plan de la main. Nous enregistrons évidemment le résultat de la tentative (balle saisie ou non).

3.4.4.2 Description de la tâche

La tâche consistait à attraper une balle lancée de biais vers l'utilisateur. Cette interception se faisait à une main, paume tournée vers le ciel, à peu près au niveau de la ceinture dans le *plan horizontal de référence* de la Main Virtuelle (le plan horizontal situé à la position initiale de la Main Virtuelle). Contrairement à l'expérience précédente, la saisie était automatiquement effectuée lorsque la balle entrait en contact avec la Main Virtuelle, il n'était donc pas nécessaire de fermer la main pour que la saisie s'effectue.

Compte tenu des interfaces utilisées dans cette expérience, le seul dispositif dont devait "s'équiper" le sujet était le SPIDAR, et cela consistait simplement à passer son majeur dans la bague destinée à cette effet (voir figure 3.19). Lorsqu'il était prêt, il le signalait à l'opérateur, qui appuyait alors sur une touche pour effectuer la calibration du système et lancer la série de balles à attraper.

La calibration consistait simplement à noter la position de la main réelle mesurée par le SPI-

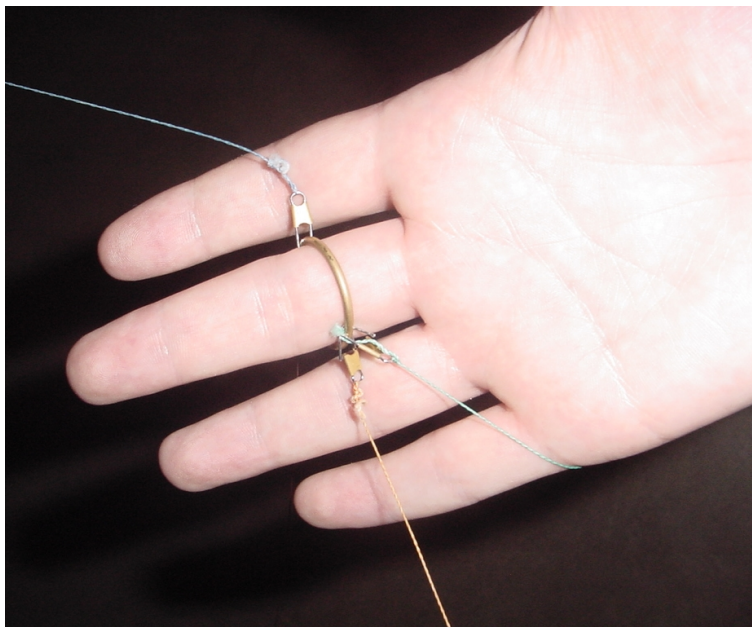


FIG. 3.19 – Photo de la bague du SPIDAR

DAR. La position ainsi obtenue correspondait alors à la position initiale de la Main Virtuelle.

3.4.4.3 Monde virtuel

Par rapport à l'expérimentation précédente, le monde virtuel est sensiblement le même, à ceci près que nous avons décidé d'ajouter des couleurs, dans les tons bleus, aux murs de l'enceinte virtuelle. Le sol est pour sa part en vert foncé afin de bien le différencier des murs latéraux

et de celui du fond. La balle et la main virtuelle sont identiques à ceux utilisés précédemment, si ce n'est que la main est cette fois-ci présentée paume vers le ciel. Enfin, lorsque l'une des ALID précédemment décrites est activée, un disque vert est affiché pour symboliser le point d'intersection entre la trajectoire de la balle et le plan horizontal de référence de la main. Une capture d'écran de ce monde virtuel est illustrée par la figure 3.20.

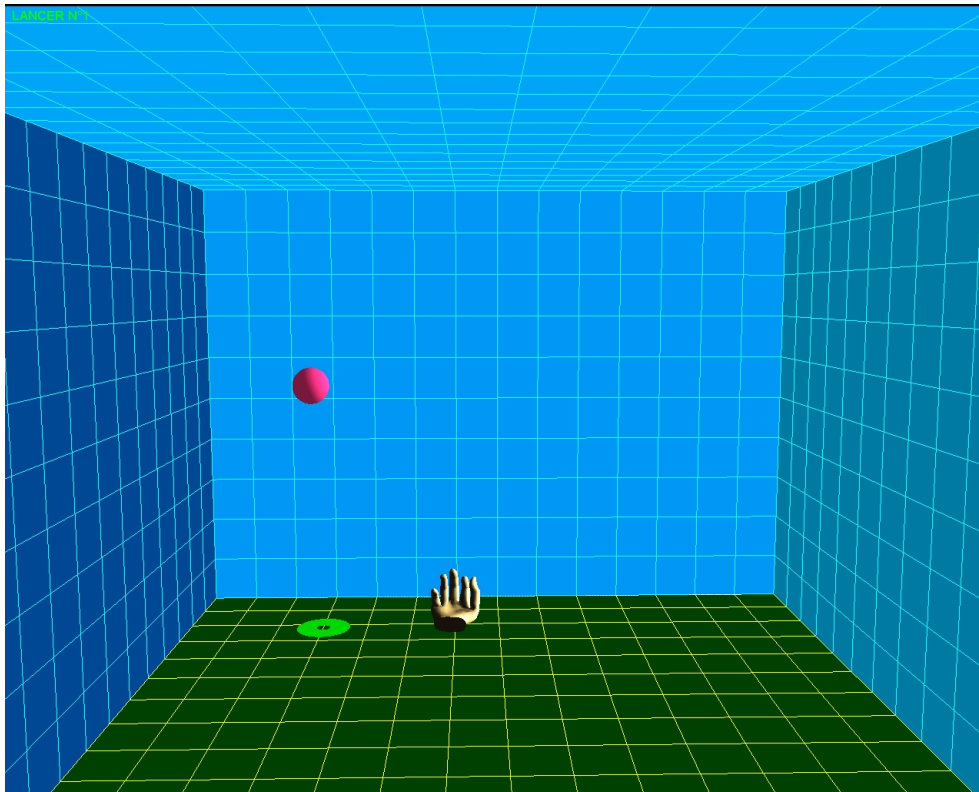


FIG. 3.20 – Capture d'écran de la simulation d'attraper de balle en approche oblique

3.4.4.4 ALID n°1 : DisqueVertPermanent

Cette aide s'inspire de la marque placée sur le sol pour faciliter la réalisation d'un smash au tennis dans les jeux video (voir chapitre 1). L'idée est de faciliter le placement de la main de l'utilisateur en lui fournissant un repère situé sur le plan de référence de la Main de l'utilisateur. La structure de l'ALID DisqueVertPermanent est représentée dans le tableau 3.7.

Les Paramètres de l'ALID "DisqueVertPermanent" sont :

- **PID** → **Modèle PID** : $E_f S_f G_d$.
- **Type** → **Sous-Type** : Unique Objet 2D.
- **État** → **État_DisqueVert** : Le disque vert est affiché à la position pré-établie où la balle doit traverser le plan horizontal de la main au moment de la calibration. Soit $P_{IntersectionPrvue}$

Propriété	Valeur
Nom	DisqueVertPermanent
But	"Réalisation"
PID	"Attraper"
Type	"Visuel"
Description	Cette aide consiste en l'affichage d'un disque vert positionné sur le plan horizontal de référence de la Main, à l'endroit où la balle doit le traverser. Le disque est toujours horizontal. Cela signifie que la normale de la surface du disque et le vecteur vertical sont colinéaires.
Genre	"Statique"
Pré-condition	aucune
Post-condition	aucune
État	État_DisqueVert
Fonctionnement	aucun

TAB. 3.7 – Structure de l'ALID "DisqueVertPermanent".

la position de l'intersection prévue pour le lancer en cours, on a alors :

$$P_{DisqueVert} = P_{IntersectionPrvue}$$

L'Orientation du disque est fixe, le disque est toujours parallèle au sol.

3.4.4.5 ALID n°2 : DisqueVertTemporaire

Cette aide reprend la même structure que la précédente, à ceci près qu'elle n'est pas affichée en permanence. En effet, si l'ALID DisqueVertPermanent est valide tout au long du lancer, l'ALID DisqueVertTemporaire n'est activée que quelques instants, et ce dans le but de ne fournir un repère à l'utilisateur que temporairement. Ce dernier a ainsi une indication visuelle pour le placement de sa main, mais il ne peut s'assurer de la précision de ce placement, le disque ayant disparu avant que la balle ne franchisse le plan horizontal de référence.

La structure de l'ALID DisqueVertTemporaire est donnée par le tableau 3.8.

Les Paramètres de l'ALID "DisqueVertTemporaire" sont :

- **PID** → **Modèle PID** : $E_f S_f G_d$.
- **Type** → **Sous-Type** : Unique Objet 2D.
- **État** → **État_DisqueVert** : Le disque vert est affiché à la position pré-établie où la balle doit traverser le plan horizontal de la main au moment de la calibration. Soit $P_{IntersectionPrvue}$ la position de l'intersection prévue pour le lancer en cours, on a alors :

Propriété	Valeur
Nom	DisqueVertTemporaire
But	"Apprentissage"
PID	"Attraper"
Type	"Visuel"
Description	Cette aide consiste en l'affichage d'un disque vert positionné sur le plan horizontal de référence de la Main, à l'endroit où la balle doit le traverser. Le disque n'est affiché que durant un court laps de temps : de 0.1s à 0.6s après le début du lancer. Le disque est toujours horizontal. Cela signifie que la normale de la surface du disque et le vecteur vertical sont colinéaires.
Genre	"Statique"
Pré-condition	$t_{lancer} = 0.1s$ avec t_{lancer} le temps écoulé à partir du moment où la balle est projetée.
Post-condition	$t_{lancer} = 0.6s$ avec t_{lancer} le temps écoulé à partir du moment où la balle est projetée.
État	État_DisqueVert
Fonctionnement	aucun

TAB. 3.8 – Structure de l'ALID "DisqueVertTemporaire".

$$P_{DisqueVert} = P_{IntersectionPrvue}$$

L'Orientation du disque est fixe, le disque est toujours parallèle au sol.

3.4.4.6 Protocole expérimental

Pour cette expérience, nous avons fait appel à 15 sujets, tous droitiers, répartis en 3 groupes de 5 personnes. Chaque sujet devait effectuer 2 séries de 50 interceptions chacune, soit un total de 100 interceptions à réaliser. Trois conditions expérimentales étaient évaluées :

- Condition 1 (notée C1) : Aucune aide activée.
- Condition 2 (C2) : Avec affichage d'un disque vert temporaire (ALID DisqueVertTemporaire).
- Condition 3 (C3) : Avec affichage d'un disque vert permanent (ALID DisqueVertPermanent).

Le tableau 3.9 révèle les conditions utilisées pour chaque série de chaque groupe. Ce protocole est conforme à celui que nous avons proposé à l'issue de l'expérimentation précédente.

Groupe	1ère Série	2nde Série
Groupe 1	C1 (aucune aide)	C1 (aucune aide)
Groupe 2	C2 (DisqueVertTemporaire)	C1 (aucune aide)
Groupe 3	C3 (DisqueVertPermanent)	C1 (aucune aide)

TAB. 3.9 – Conditions pour chaque série de 50 balles, par groupe.

Les retours obtenus sur notre précédente expérimentation d'attraper d'objets indiquaient que le temps d'attente durant une interception était ressenti comme trop élevé. Nous avons donc réduit cette attente : une balle est envoyée toutes les 3 secondes (au lieu de 8). La balle est fixe, à sa position initiale, durant 1 seconde (contre 5 précédemment) avant d'être propulsée vers l'utilisateur. Son temps de vol total est de 2 secondes (3 dans l'expérience précédente), mais elle franchit le plan horizontal de référence après seulement 0.813 seconde.

Le projectile arrivait dans le plan sur l'un des 5 points de passage pré-définis illustrés sur la figure 3.21. Lors d'une série de 50 balles, la balle passait 10 fois par chaque point, mais dans un ordre préalablement déterminé de manière à paraître aléatoire et dont le sujet n'avait pas connaissance.

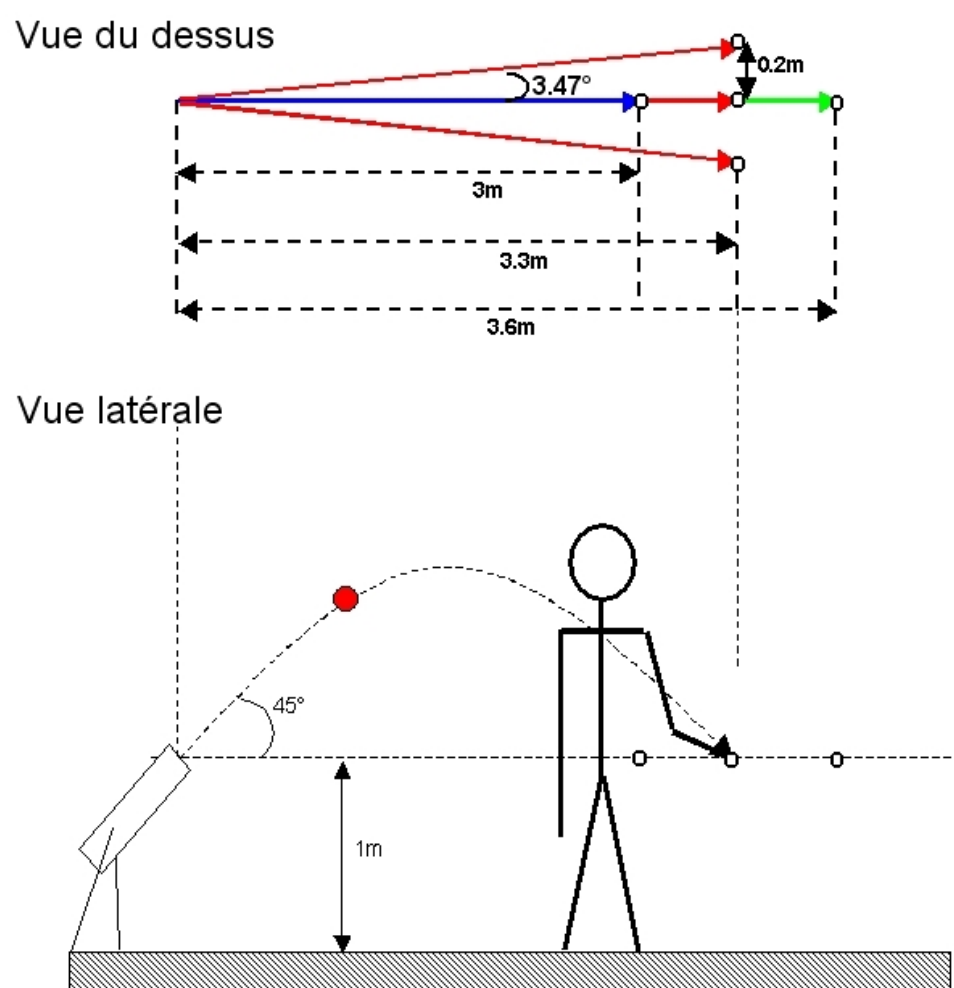


FIG. 3.21 – Points de passage de la balle dans le plan horizontal de la Main

3.4.4.7 Analyse des résultats et discussion

De la même manière que précédemment, nous avons, pour chaque tentative, enregistré la réussite ou l'échec de l'interception. Nous avons également mesuré la distance entre la main et la balle lorsque celle-ci passait dans le plan de la main.

Ces données ont été soumises à l'ANOVA.

Nombre de Balles Attrapées : Le tableau 3.10 synthétise les résultats obtenus par Session et par Groupe, pour les 5 sujets du groupe.

1ère Session				2nde Session		
Groupe 1 : C1	Groupe 2 : C2	Groupe 3 : C3		Groupe 1 : C1	Groupe 2 : C1	Groupe 3 : C1
19	11	28		39	12	13
0	24	27		14	8	11
7	26	16		23	5	2
11	18	15		13	21	1
18	24	25		10	23	13

TAB. 3.10 – Balles attrapées par les sujets, par groupe et par session

Au total, le Groupe 1 a intercepté 55 balles lors de la première session (C1, pas d'ALID) alors que les Groupes 2 et 3 en ont attrapées respectivement 103 et 111 (ALID DisqueVertTemporaire et DisqueVertPermanent). En revanche, lors de la seconde Session, le groupe 1 a réalisé la meilleure performance avec 99 balles saisies, contre 69 et 40 pour les groupes 2 et 3. Ces résultats sont illustrés par la figure 3.22.

En appliquant la méthode statistique ANOVA sur les résultats obtenus par Session, on obtient l'analyse suivante :

– 1ère Session

Pour cette première session, où une ALID était fournie aux groupes 2 et 3, l'ANOVA relève une différence significative entre les groupes ($F = 3.96$, $p = 0.0478 < 0.05$). Si les groupes 2 et 3 sont relativement homogènes, avec des moyennes de balles attrapées respectivement égales à 20.6 (Std : 6.148) et 22.2 (Std : 6.221), le Groupe 1 quant à lui affiche une moyenne de balles saisies bien inférieure, égale à 11 (Std 7.906) (figure 3.23).

Ces résultats montrent l'effet très nettement positif des Aides Logicielles pour l'Interaction Dynamique qui ont été proposées aux Groupes 2 et 3 pour la réalisation de la tâche.

– 2nd Session

Pour cette seconde Session, où aucun Groupe ne disposait d'aide logicielle, l'ANOVA ne dénote aucune différence significative ($F = 2.20$, $p = 0.1535 > 0.05$). Toutefois, si l'on

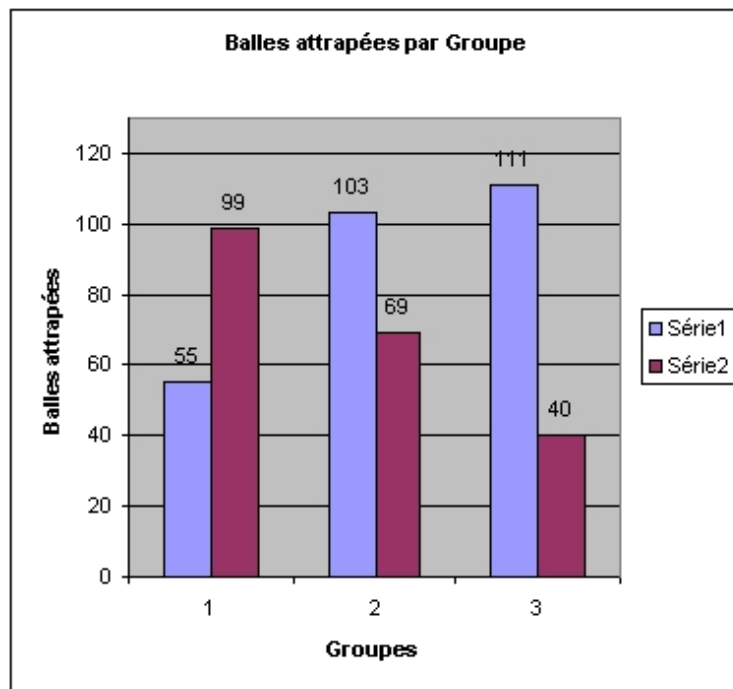


FIG. 3.22 – Balles Attrapées par Groupe

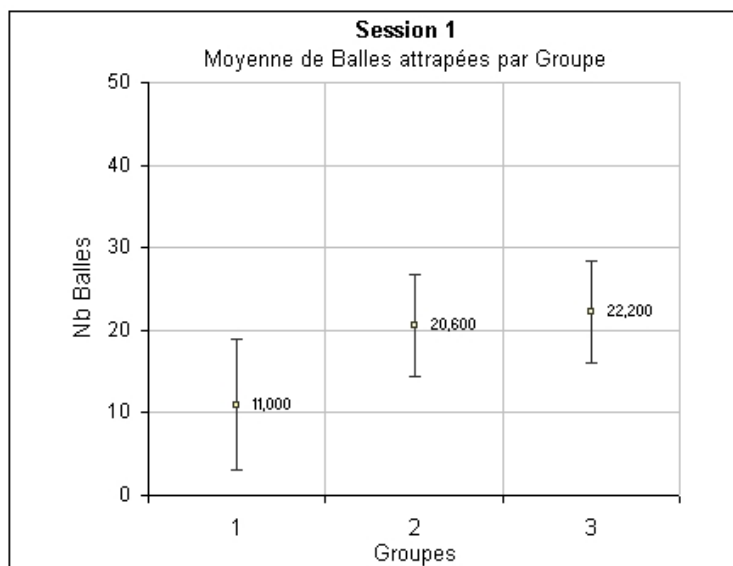


FIG. 3.23 – Moyenne de Balles Attrapées par Groupe pour la Session 1

se penche sur le nombre moyen de balles attrapées selon les groupes pour cette session, on se rend compte que la meilleure performance a été réalisée par le Groupe 1, avec une moyenne de 19.8 balles (Std : 11.777), contre 13.8 pour le Groupe 2 (Std : 7.918) et surtout 8 pour le Groupe 3 (Std : 6) (voir figure 3.24).

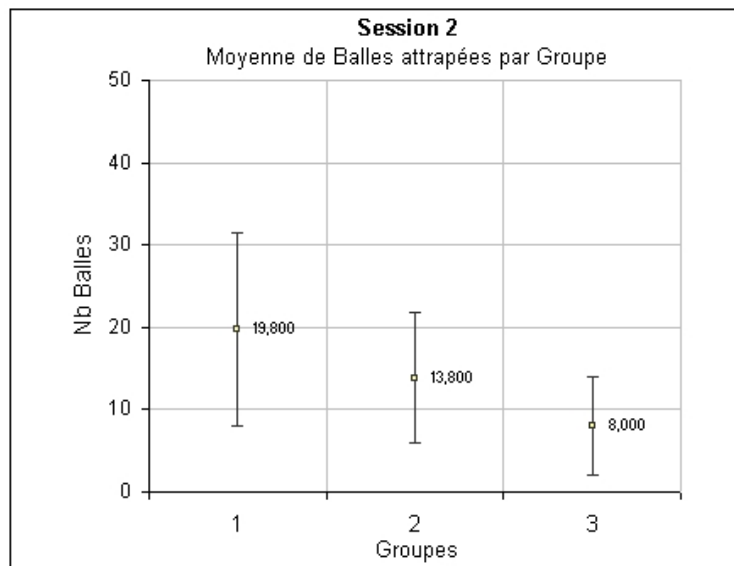


FIG. 3.24 – Moyenne de Balles Attrapées par Groupe pour la Session 2

Ces écarts entre les groupes 2 et 3, ayant disposés des ALID lors de la 1ère Session, et le Groupe 1, qui n'a jamais bénéficié d'aide logicielle, ne sont, à notre sens, pas négligeables. Ils tendent à prouver que, si les ALID présentées ici ont un effet fortement positif pour la réalisation de la tâche lorsqu'elles sont activées, elles entraînent également un effet nuisible pour l'apprentissage de la tâche.

Il a en effet été observé, et relevé par plusieurs sujets, que l'attention de l'utilisateur était davantage fixée sur le disque vert affiché que sur la trajectoire quand celui-ci était présent. Ceci explique qu'aucun apprentissage ne se fasse lors de la première session pour l'utilisateur bénéficiant d'une des deux ALID proposées ici, et qu'il se sente un peu "perdu" lorsque le disque lui est retiré lors de la seconde session.

Enfin, nous avons émis l'hypothèse que l'ALID DisqueVertTemporaire serait plus intéressante pour l'apprentissage de la tâche que l'ALID DisqueVertPermanent, de part son caractère temporaire. Cette hypothèse n'est pas réellement validée, ni complètement réfutée non plus, dans le sens où le Groupe disposant de l'ALID DisqueVertTemporaire a attrapé 72.5% plus de balles lors de la seconde Session que le Groupe qui avait disposé de l'ALID DisqueVertPermanent. Néanmoins, le nombre de balles attrapées dans cette

condition reste bien moindre que dans le cas où aucune n'était fournie en premier lieu.

Distance Main-Balle : Comme nous allons le voir, on retrouve les mêmes informations que précédemment en s'intéressant à la distance entre la main et la balle au moment où cette dernière passe dans le plan de la main. Dans le tableau 3.11, les valeurs correspondent à la somme des distances Main-Balle (en centimètre) des 50 tentatives de la série, pour les 5 sujets du groupe.

1ère Session			2nde Session		
Groupe 1 : C1	Groupe 2 : C2	Groupe 3 : C3	Groupe 1 : C1	Groupe 2 : C1	Groupe 3 : C1
8.27	9.09	5.62	3.94	10.07	20.39
30.78	7.25	5.46	12.35	17.66	14.99
13.16	6.49	8.97	8.31	26.29	28.13
18.17	8	8.51	11.81	7.68	37.38
9.49	5.68	6.92	15.89	6.74	11.96

TAB. 3.11 – Distances Main-Balle totales, par groupe et par session

– 1ère Session

L'ANOVA révèle une différence significative entre les groupes ($F = 4.39$, $p = 0.0370 < 0.05$), et plus précisément entre les groupes 1 et 2 d'une part, et les groupes 1 et 3 d'autre part (les groupes 2 et 3 étant homogènes). La moyenne de la somme des distances Main-Balle est d'environ 15.974 pour le groupe 1 (Std : 9.129) contre respectivement 7.302 et 7.096 pour les groupes 2 et 3 (Std : 1.321 et 1.612). Ces résultats sont illustrés par la figure 3.25.

On peut donc en conclure que les ALID utilisées pour cette expérimentation permettent de réduire significativement les marges d'erreurs entre la position de la main et la position de la balle lorsqu'elles sont activées.

– 2nd Session

Pour cette seconde Session, l'ANOVA n'indique globalement aucune différence significative entre les groupes. ($F = 3.03$, $p = 0.0861 > 0.05$). En revanche, si on regarde plus en détail, elle en révèle une entre le Groupe 1 et le Groupe 3. Les Groupes 1 et 2 d'une part, et 2 et 3 d'autre part, sont eux considérés homogènes. La moyenne de la somme des distances Main-Balle pour cette Session est approximativement de 10.460 pour le groupe 1 (Std : 4.528), de 13.688 pour le groupe 2 (Std : 8.247), mais surtout de 22.570 pour le groupe 3 (Std : 10.308). La figure 3.26 illustrent ces résultats.

Cette fois encore, ces chiffres ont tendance à prouver que l'utilisation des ALID lors de

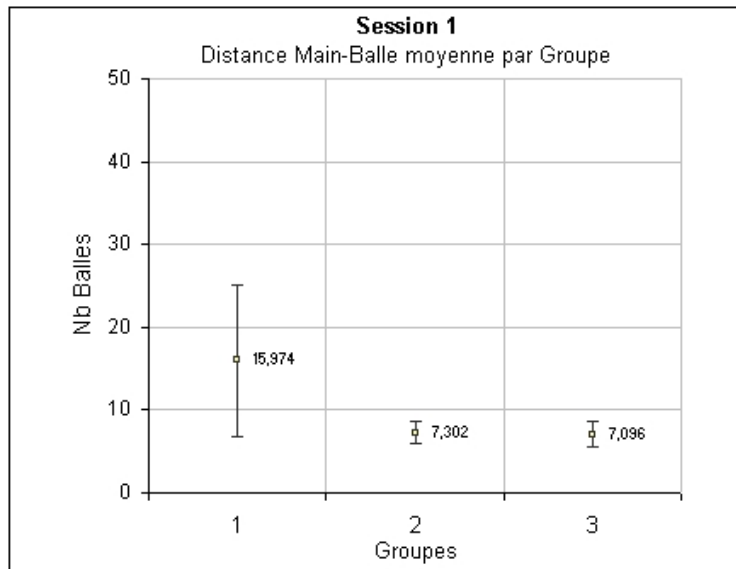


FIG. 3.25 – Moyenne des sommes des distances Main-Balle, par Groupe, pour la Session 1

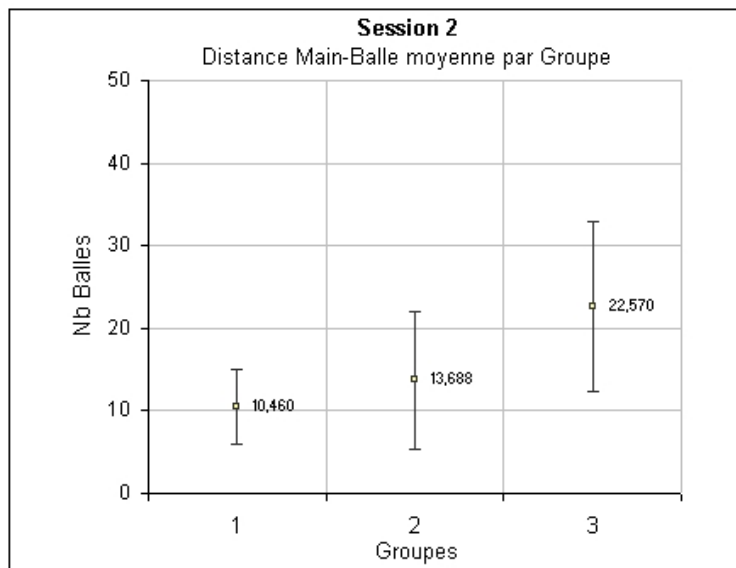


FIG. 3.26 – Moyenne des sommes des distances Main-Balle, par Groupe, pour la Session 2

la première Session des groupes 2 et 3 n'a pas eu de conséquences positives sur les sujets pour l'apprentissage de la tâche. Au contraire, on peut relever un effet plutôt néfaste, au vu des performances moindres pour les sujets de ces deux groupes que celles des sujets du groupe 1.

Points de passage de la Balle : Nous nous sommes intéressés au nombre de balles attrapées en fonction de son point de passage dans le plan de la main. Les points de passage étaient numérotés de la manière suivante :

- **1** : devant la main de l'utilisateur
- **2** : à gauche de la main de l'utilisateur
- **3** : sur la main de l'utilisateur
- **4** : à droite de la main de l'utilisateur
- **5** : derrière la main de l'utilisateur

Cette disposition est illustrée sur la figure 3.27.

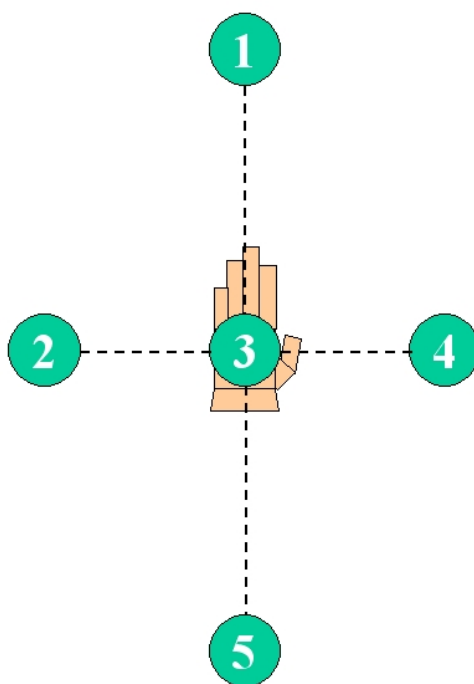


FIG. 3.27 – Numérotation des Points de passage de la balle dans le plan de la main

Au total, sur l'ensemble des deux séries, 45 balles ont été attrapées au Point de passage n°1, 132 au Point n°2, 116 pour le n°3, 97 au n°5 et 87 pour le Point de passage n°5. L'ANOVA réalisée sur le nombre de balles attrapées à chaque Point de passage, pour les deux séries de chaque groupe, révèle une différence significative entre les Points ($F = 2.80$, $p = 0.0477 < 0.05$).

Plus précisément, les Points 1 et 2 d'une part, et 1 et 3 d'autres part sont statistiquement et significativement différents.

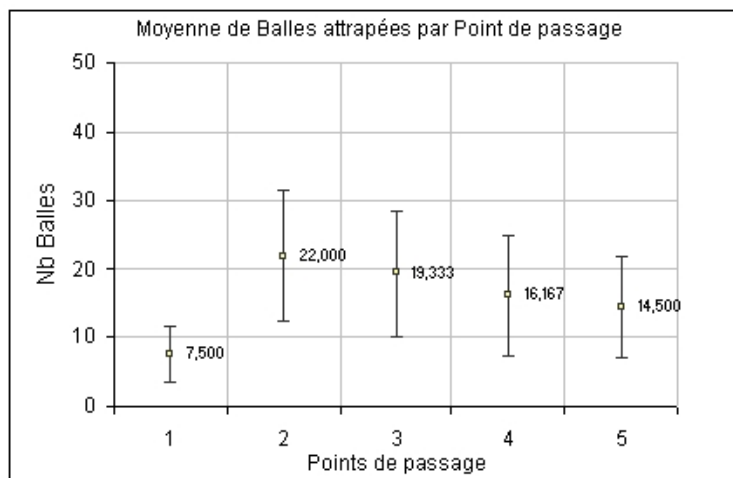


FIG. 3.28 – Moyenne de balles attrapées par Points de passage

En effet, comme indiqué sur la figure 3.28, en moyenne 7.5 balles ont été attrapées quand la balle passait devant la main (Std : 4.037), contre 22.0 quand elle passait à gauche (Std : 9.529) et 19.333 au point de référence de la main (Std : 9.245). Cette moyenne est de 16.147 pour les balles arrivant à droite de la main (Std : 8.750) et de 14.5 pour celles arrivant derrière la position de référence (Std : 7.530).

Ces chiffres confirment les commentaires laissés par un certain nombre de sujets, à savoir qu'il était difficile de percevoir la profondeur à laquelle se situait les objets. Cette difficulté était d'autant plus marquée lorsque la balle devait passer par le Point n°1, ce qui explique ces résultats, et démontre que la perception de la profondeur reste un problème majeur en Réalité Virtuelle, et ce d'autant plus durant les tâches d'interception d'objets en mouvement.

Au final, cette expérimentation a permis de valider le protocole expérimental proposé à l'issue des deux premières expériences infructueuses qui avaient été menées. Par ailleurs, elle a également démontré que les Aides Logicielles pour l'Interaction Dynamique pouvait être utilisées avec succès pour améliorer la Réalisation d'une tâche d'interaction dynamique. Enfin, elle a révélé que les ALID qui avaient été testées n'étaient pas adaptées pour l'Apprentissage de la tâche, et pouvaient même avoir un effet assez néfaste sur cet apprentissage.

3.5 Conclusion

Dans ce chapitre, nous avons expliqué pourquoi nous avons choisi les tâches basiques de lancer de balle et d'interception d'objets comme cadre à l'évaluation de nos Aides Logicielles pour l'Interaction Dynamique. Nous avons par la suite décrit comment l'application que nous avons utilisée avait été conçue et développée, et nous avons détaillé les deux premières expérimentations qui ont été conduites dessus : le lancer de balle sur cible fixe et l'interception d'objets en approche frontale. Confrontés à des résultats peu encourageants, nous avons alors porté un regard très critique sur l'environnement virtuel et sur le protocole expérimental qui avaient été utilisés lors de ces évaluations. Cette réflexion nous a amenés à proposer un nouveau protocole expérimental, adapté à l'évaluation de nos Aides Logicielles pour l'Interaction Dynamique. Pour finir, nous avons mené une seconde expérimentation sur l'interception d'objets en mouvement, cette fois-ci en approche oblique, qui a mis en lumière tant l'utilité des ALID testées pour la Réalisation de la tâche que leur inefficacité, voire leur caractère nuisible, pour l'Apprentissage de la tâche.

Toutes les tâches à réaliser jusque là nécessitaient l'exécution d'un Geste Continu et comportaient un objectif simple. Nous allons donc nous intéresser dans le prochain chapitre à la mise en oeuvre d'ALID dans le cas d'une activité de lancer requérant un Geste Discret dans un but complexe : le Tir à l'arc.

Chapitre 4

Application au Lancer Discret : Le Tir à l'Arc

4.1 Introduction

Nous commencerons dans cette partie par présenter les raisons qui nous ont amenées à considérer le tir à l'arc comme application possible pour l'évaluation de nos Aides Logicielles pour l'Interaction Dynamique. Ensuite, nous décrirons le processus de conception et de développement de l'application de tir à l'arc que nous avons créée. Enfin, nous détaillerons les deux expérimentations qui ont été conduites sur cette simulation.

4.2 Justification

Nous avons décidé d'évaluer l'apport des Aides Logicielles pour l'Interaction Dynamique dans le cadre du tir à l'arc car cette activité présentait plusieurs aspects intéressants. Tout d'abord, il s'agit d'un cas où l'énergie est fournie au projectile de manière quasi-instantanée. Dans notre modélisation, c'est un Lancer avec un Geste Discret, alors que nous avons vu dans le chapitre 1 que la plupart des systèmes qui ont fait l'objet d'études font appel à un Geste Continu. De plus, la vitesse initiale de la flèche est comparativement bien plus faible que la vitesse initiale des projectiles tirés par des armes à feu. Il faut donc davantage gérer et compenser l'effet de la gravité sur la trajectoire de la flèche qu'avec des armes dont les projectiles ont une trajectoire quasiment rectiligne sur la distance les séparant de la cible. Par ailleurs, le tir à l'arc implique un contrôle bi-manuel : la main gauche tient l'arc et l'oriente vers la cible pendant que la main droite effectue la traction sur la corde et provoque l'échappement de la flèche. Enfin, le tir à l'arc nécessite un contrôle d'effort en simultané avec l'orientation de l'arc, dû à la force

exercée par la corde en réponse à la traction effectuée par l'archer.

4.3 Conception et développement du système

4.3.1 Analyse et modélisation de l'interaction

Basiquement, le tir à l'arc consiste à projeter une flèche dans le centre d'une cible à l'aide d'un arc, et généralement à répéter cette action un certain nombre de fois. On détecte deux notions importantes : la précision et la reproductibilité du geste. Trois grands types de facteurs déterminant la performance d'un archer peuvent être distingués :

- des facteurs mécaniques
- des facteurs d'environnement
- des facteurs humains

Les facteurs mécaniques sont principalement constitués de la vitesse de la flèche et de la stabilité de sa trajectoire. La précision du tir nécessite une vitesse initiale et une stabilité en vol de la flèche. La reproductibilité dépend essentiellement de la qualité des matériaux et de la fabrication.

Le vent et la pluie sont les facteurs environnementaux agissant directement sur la performance, de par le fait que leur impact se situe sur la trajectoire de la flèche et sur l'équilibration du tireur.

Néanmoins, les facteurs d'environnement comprennent également les éléments pouvant influencer sur les capacités psychiques de l'individu, tels que le niveau de compétition ou le lieu de compétition par exemple (dans le cadre d'un tir sportif). En fait, tous les éléments ou représentations de ces éléments par le tireur pouvant induire des réactions émotionnelles modifiant la précision et, plus généralement, la répétitivité des tirs, sont considérés comme des facteurs environnementaux.

Enfin, les facteurs humains restent les plus difficiles à cerner et leur investigation fait appel à plusieurs domaines de recherche (physiologie, psychologie, psychophysiologie, biomécanique...). L'observation de ces facteurs se résume à celle de l'activité du tireur, c'est-à-dire sa technique.

La séquence de tir peut se décomposer en trois phases :

- phase de placement : le tireur se positionne par rapport à la cible et assure sa tenue d’arc.
- phase de mise en tension : recherche d’un équilibre dynamique entre la force de traction à l’arrière (bras de corde) et la force de répulsion à l’avant (bras d’arc). Lorsque cette phase s’achève, le tireur prend différents points de repère, notamment un alignement œil-viseur-cible.
- phase terminale : étape durant laquelle les actions de visée et de traction-répulsion se poursuivent jusqu’à la fin du tir.

Il faut toutefois noter que cette séquence de tir est envisagée en tant que programme de tir global, et non pas comme une suite d’actions additionnées.

Comme dans la plupart des activités, l’apprentissage ou l’entraînement au tir à l’arc passe par la répétition d’actions et la prise de connaissance du résultat de ces actions. Dans les deux cas, cela nécessite une prise d’informations :

– **En cours d’action :**

Le tireur perçoit le mouvement par différents canaux sensoriels. Ce sont des sensations tactiles pour le placement des mains de corde et d’arc, des sensations visuelles qui lui permettent de se situer dans l’espace et d’analyser les variations du milieu extérieur, et des sensations proprioceptives lui rendant compte de ses placements articulaires et de ses tensions musculaires. Ces prises d’informations donnent lieu après analyse à une modification du mouvement au cours de l’action ou lors de l’action suivante. Les moyens extérieurs pouvant être utilisés sont les interventions verbales de l’entraîneur ou la visualisation du tireur dans un miroir. Notons toutefois que ces moyens demandent une attention intellectuelle supplémentaire.

– **En fin de tir :**

Le tireur peut prendre des informations sur la réaction de son arc, la réaction de la corde après l’échappement, ainsi que sur le vol de la flèche et son impact sur la cible. Pendant l’entraînement, les instructions ou conseils de l’entraîneur basés sur ses observations et la visualisation d’enregistrements vidéo sont les différents moyens de prise de connaissance du résultat les plus communément utilisés.

La réalité virtuelle étant basée sur une approche centrée utilisateur, il est intéressant de déduire de cette analyse du tir à l’arc en réel une modélisation des actions que l’utilisateur sera amené à réaliser dans une simulation de tir à l’arc virtuel. La figure 4.1 représente cette modélisation.

U : Utilisateur
 $U = \{U_r, U_p, U_a\}$

U_r : Utilisateur au repos
 U_p : Utilisateur en placement
 U_a : Utilisateur en action

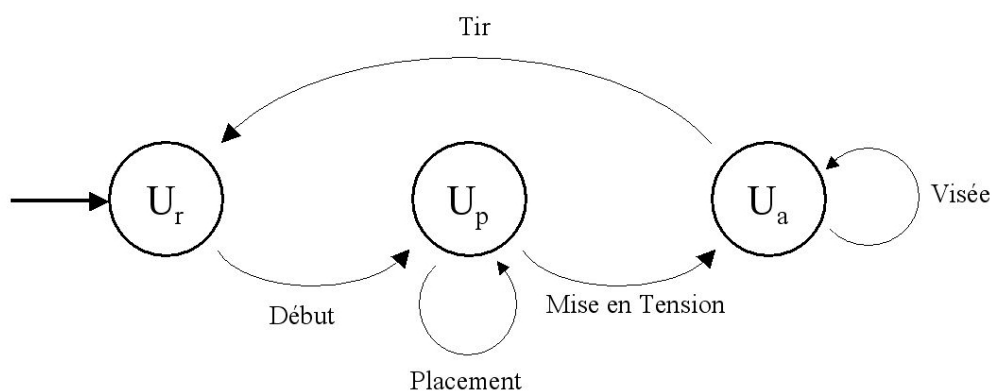


FIG. 4.1 – Automate de l'Archer

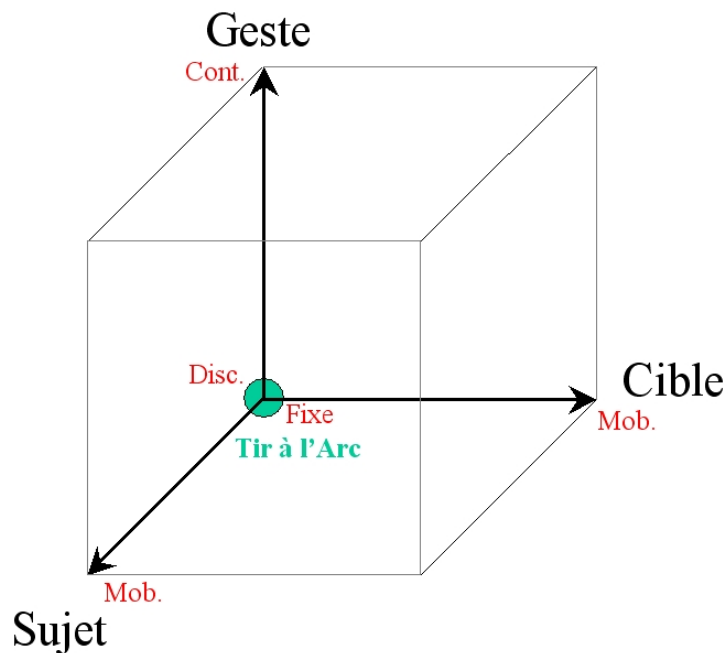


FIG. 4.2 – Le Tir à l'Arc situé sur le cube de la Modélisation du Lancer

Si l'on se réfère à notre Modélisation des Interactions Dynamiques (Chapitre 2), le tir à l'arc sur cible fixe tel que nous venons de le présenter est l'archétype de la situation $S_f G_d C_f$ (Sujet Fixe, Geste Discret, Cible Fixe) du Lancer. Il est illustré par la figure 4.2.

4.3.2 Périphériques et technique d'interaction

Plusieurs possibilités techniques permettant à un utilisateur de tirer à l'arc dans une simulation informatique ont été envisagées, y compris pour l'archerie montée [IKM⁺02]. L'interface la plus basique et déjà utilisée dans de nombreux jeux vidéo reste l'utilisation du clavier et de la souris. Cependant, cette technique d'interaction reste trop éloignée du tir réel pour la simulation que nous voulions.

On peut également penser à un arc virtuel, qui serait affiché dans la simulation, l'utilisateur mimant le geste du tir à l'arc avec ses bras. Cette possibilité se rapproche du tir réel, mais comme nous l'avons vu, les sensations haptiques sont très importantes dans cette activité, or elles ne seraient pas restituées avec cette technique. C'est pour cette raison que nous avons opté pour l'utilisation d'un arc réel, avec une corde réelle. Pour des raisons de sécurité évidentes, la flèche devait rester virtuelle, ce qui pose un problème d'offset entre l'arc, tenu en main et donc proche visuellement de l'utilisateur, et la flèche qui sera affichée sur un écran quelconque, et qui ne serait donc pas vue à sa place normale dans le réel. Compte tenu des caractéristiques assez médiocres des casques virtuels, et de la position de la tête du tireur, nous avons conclu que nous ne pouvions pas les choisir comme système de visualisation, d'autant qu'un environnement complètement virtuel nous semblait inutile et potentiellement gênant pour une activité de précision telle que le tir à l'arc.

Enfin, la flèche étant située très proche des yeux de l'utilisateur, il ne nous semblait pas approprié d'utiliser un système de réalité augmentée, la caméra ne pouvant être située à une position permettant un affichage correct de la flèche virtuelle.

Pour connaître la vitesse et la direction du tir, il nous fallait connaître l'orientation de l'arc et la traction de la corde au moment de l'échappement. Un capteur de position électromagnétique a été positionné sur l'arc et un autre sur la corde. Le vecteur formé par les deux positions 3D obtenues nous permet de connaître la direction du tir d'une part, et la force de traction d'autre part, grâce à la norme du vecteur.

En effet, comme indiqué sur la figure 4.3, on peut calculer les angles de tir à partir des positions des deux capteurs (angle θ par rapport à l'horizontale sur la figure). De même la force de traction est calculée en comparant la distance entre les capteurs "D" par rapport à la distance de référence lorsque l'arc est repos "d".

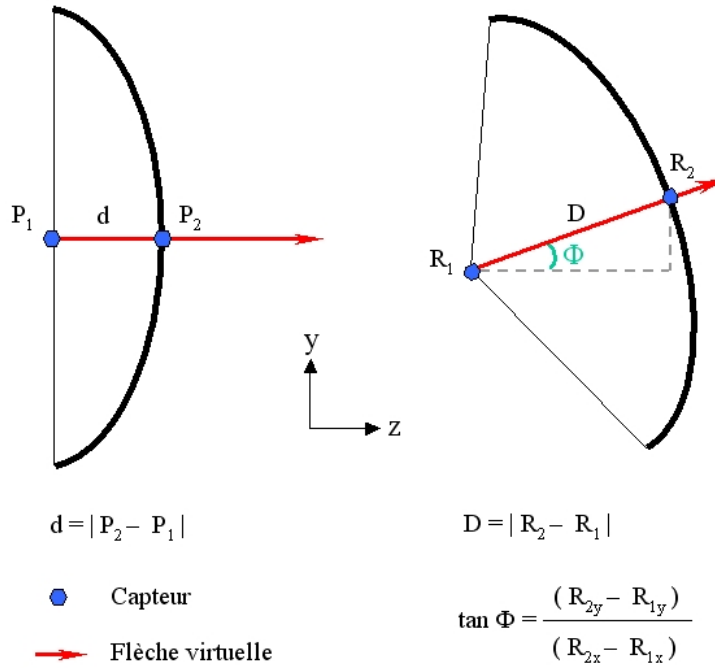


FIG. 4.3 – Position idéale des capteurs de position sur l’arc, au repos à gauche, en action à droite

Malheureusement, le placement du capteur ne fut pas possible pour des raisons de sécurité, le capteur risquant d’être éjecté à chaque tir. Il a donc été décidé de le fixer sur la main de l’utilisateur, et de tenir compte, lors d’une calibration, de l’offset généré par ce changement de positionnement (figure 4.4).

La détection de l’échappement fut beaucoup plus problématique compte tenu du geste lui-même. Il s’agit en fait d’une très légère modification de la flexion des doigts, et plus particulièrement de la dernière phalange des doigts. Nous avons opté pour l’utilisation d’un gant de données renvoyant la valeur de flexion des doigts, mais comme ce dernier n’était pas d’une grande précision, il a fallu modifier légèrement l’interaction par rapport à l’échappement réel. Dans notre simulation, lorsque les valeurs de flexion des deux doigts commandant l’échappement (l’index et le majeur pour la méthode de préhension couramment utilisée en Europe) dépassent simultanément une valeur seuil, l’échappement est considéré effectué. Cette valeur est, pour chaque doigt, celle du doigt dans son ensemble, et non pas celle de la dernière phalange. De ce fait, pour être certain que l’échappement sera détecté, l’utilisateur doit de préférence tendre bien nettement les doigts plutôt que se contenter de la dernière phalange. Cette différence par rapport au réel entraîne un petit temps d’adaptation, que nous réduisons grâce à une calibration du gant pour chaque utilisateur. La calibration consiste à demander à l’utilisateur de faire de

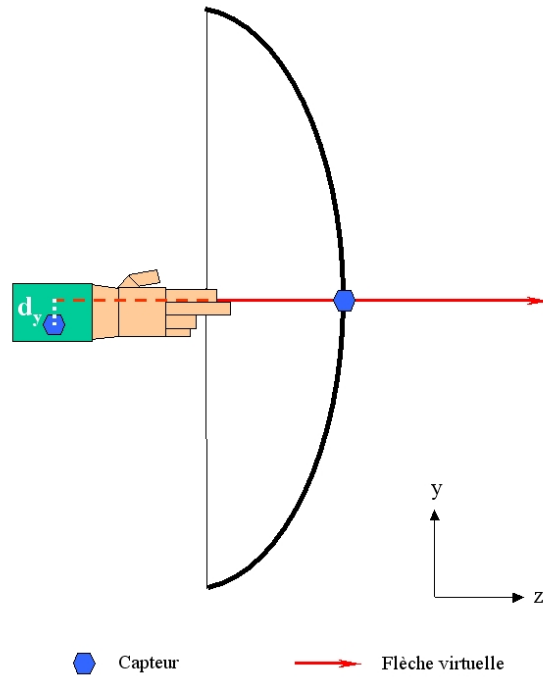


FIG. 4.4 – Position des capteurs pour notre simulation et Offset vertical d_y généré

légères flexions-extensions successives des doigts, afin de réduire la valeur seuil nécessaire à l'échappement. L'utilisateur a ainsi moins besoin d'exagérer son geste pour qu'il soit pris en compte.

Enfin, nous avons choisi un grand écran plat comme système de visualisation, afin de permettre un affichage, si ce n'est à l'échelle 1, tout au moins avec un champ de vision plus large qu'avec un simple écran d'ordinateur. Un écran rétro-projeté, comme celui de la plate-forme PREWISE, est préférable, et ce afin d'éviter la projection de l'ombre de l'utilisateur sur l'écran, ou bien le positionnement du projecteur dans le champ de vision du tireur.

La figure 4.5 illustre un utilisateur en action. Par mesure de sécurité, la corde d'origine a été remplacée par une corde plus longue, ceci afin de réduire les contraintes subies par les branches de l'arc et donc de minimiser les risques de brisure de l'arc.

4.3.3 Conception et développement logiciels

Le système a été développé en C/C++. Il a été fait appel à la librairie OpenGL pour la partie graphique, et à la librairie openAL pour la gestion des sons. Ce simulateur étant principalement destiné à fonctionner avec le système d'exploitation Windows, du code MFC permet de créer la fenêtre et de gérer les messages et événements.



FIG. 4.5 – Utilisateur de notre simulation de tir à l'arc

4.3.3.1 Machine à États

Application : Commençons par étudier le fonctionnement logiciel de l'application dans son ensemble :

- **Initialisation :**

Notre simulation commence par une phase classique d'Initialisation. Durant cette phase, outre la création de la fenêtre et la configuration d'OpenGL, on charge en mémoire les objets graphiques et leurs textures, on initialise le gant de données et les capteurs de positions et on s'intéresse aux sons et à leur gestion.

- **Calibration :**

Il s'ensuit alors une phase obligatoire de calibration, qui a pour but de vérifier la position initiale des capteurs de position afin de prendre en compte le décalage d'alignement entre celui situé sur l'arc et celui fixé sur le gant, ainsi que de mesurer les valeurs seuil qui seront utilisées pour la détection de l'ouverture et de la fermeture du gant de données. En cas de changement d'utilisateur ou de problème avec cette première calibration obligatoire, cette formalité pourra être réitérée.

- **Ok :**

On entre là dans une phase d'attente. C'est généralement dans cette période que l'on règle certains paramètres, comme le mode de visée, le mode de vue, l'affichage des aides logicielles, etc. L'utilisateur peut également en profiter pour se reposer entre deux tirs.

Lorsqu'il est prêt à tirer une nouvelle flèche, il ferme la main, ce qui initialisera une flèche virtuelle et marquera le début de la phase suivante.

– **Visée :**

A partir du moment où une flèche a été initialisée, l'utilisateur manipule son arc, en l'orientant vers la cible d'une part et en tirant sur la corde d'autre part, dans le but de projeter la flèche dans le centre de la cible. Il faut donc ici effectuer les calculs d'orientation de l'arc et de la flèche en fonction de données fournies par les capteurs de position. Si une aide logicielle est activée, il faut également lancer la procédure de calcul prédictif du point d'impact et du score. Ceci se réalise en continu tant que l'ouverture du gant de données, consécutive à l'échappement et provoquant le tir dans la simulation, n'aura pas été détectée.

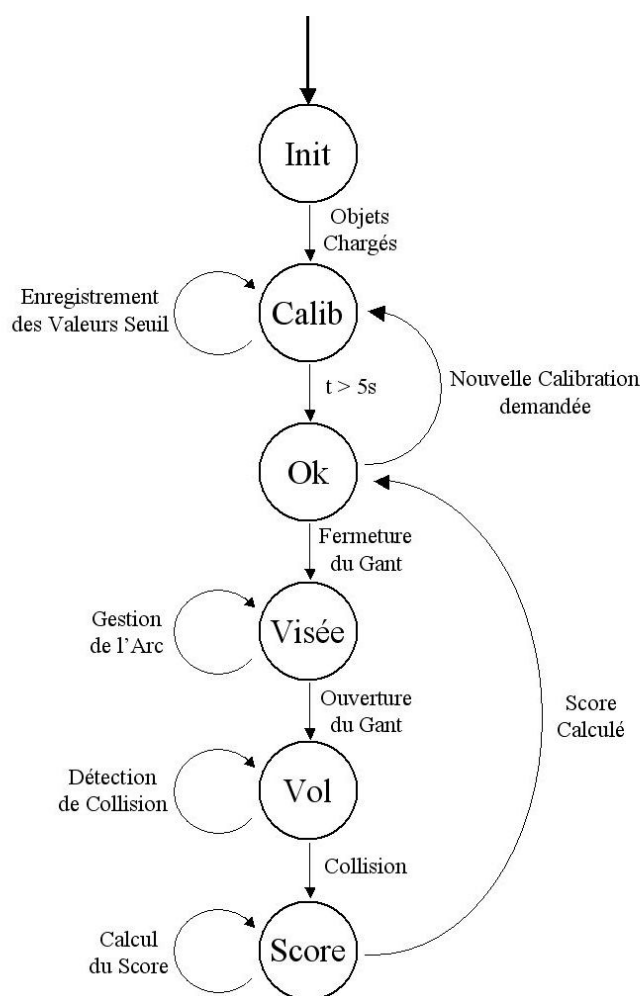


FIG. 4.6 – Diagramme d'États de la Simulation

– **Vol :**

Lorsque l'échappement a été effectué, la flèche, qui était jusque là assujettie à l'arc, est libérée avec une vitesse et une direction initiales, calculées durant la phase précédente. Une fois en vol, elle n'est soumise qu'à la gravité (les autres paramètres physiques étant ignorés) et sa trajectoire parabolique ne sera interrompue qu'en cas de collision avec la cible ou avec le sol. Nous effectuons donc les calculs de détection de collision entre la flèche et la cible, et vérifions l'altitude de la flèche, durant cette phase. Lorsque le vol de la flèche a été stoppé, d'une manière ou d'une autre, cette phase prend fin.

– **Score :**

Une fois la flèche immobilisée, nous pouvons calculer le score réalisé par ce tir. Il sera nécessairement de 0 si la collision a eu lieu avec le sol. Si elle s'est produite avec la cible, le calcul suivant est réalisé :

$$\text{Score} = 1000 - \left(\frac{1000 * \text{DistanceCentre}}{\text{RayonCible}} \right) \quad \text{avec :}$$

DistanceCentre : la distance entre le point d'impact de la flèche et le centre de la cible.

RayonCible : Le rayon de la cible (cf figure 4.7).

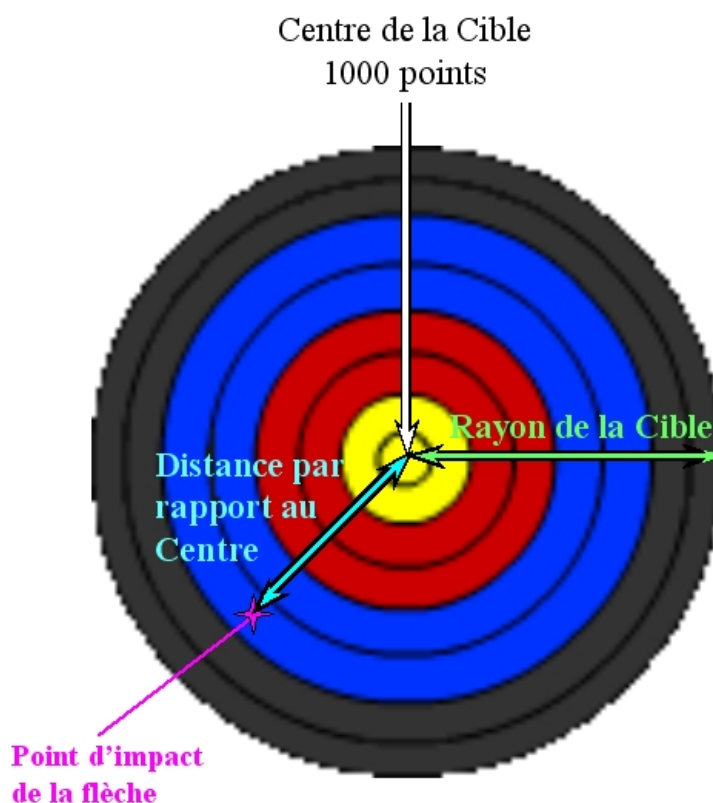


FIG. 4.7 – Schéma du Calcul du Score

Lorsque le score a été déterminé, nous nous retrouvons en phase d'attente (Ok).

La figure 4.6 récapitule le déroulement de notre simulation.

La Main : Comme indiqué précédemment, l'interaction la plus importante et la plus problématique reste l'échappement, qui est détecté en terme d'interface par le gant de données. Nous avons donc d'un côté la Main Réelle de l'utilisateur, et de l'autre une Main Virtuelle qui sera commandée à partir des données collectées par le gant.

Nous partons du principe qu'une Main M est constituée de cinq doigts. Une main peut être ouverte ou fermée, selon que ses doigts sont "ouverts" ou "fermés". Toutefois, compte tenu du fait que l'échappement s'effectue, pour la technique retenue, avec seulement l'index et le majeur, nous ne prenons en compte que les valeurs de ces deux doigts pour gérer la Main Virtuelle. La figure 4.8 illustre la modélisation des doigts virtuels en fonction des doigts réels et d'une valeur seuil, détectée lors de la calibration.

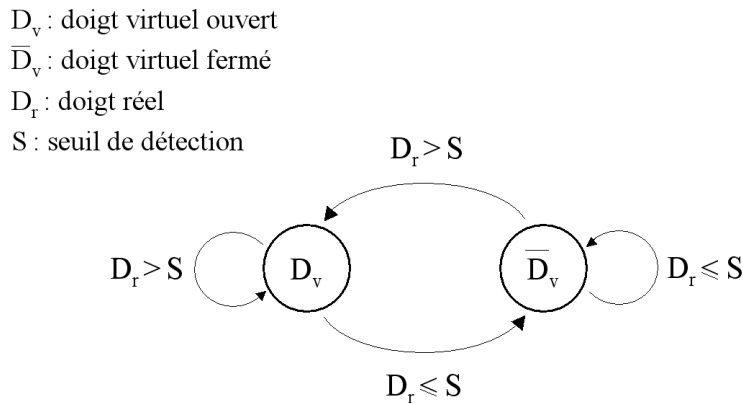


FIG. 4.8 – Diagramme d'États des Doigts Virtuels

Il en découle la modélisation de la Main Virtuelle visible sur la figure 4.9.

L'Arc et la Flèche : Nous considérons que l'Arc dispose de trois états : il peut être au repos, simplement tenu par la main la plus faible de l'utilisateur (généralement la gauche) ; il peut être initialisé, c'est à dire que la main forte (principalement la droite) s'est refermée sur la corde, "encochant" une flèche ; et il peut être mis en tension, l'échappement provoquant le tir. Les changements d'état de l'arc se font ici par l'intermédiaire de la Main Réelle, la seule pouvant agir dessus. La figure 4.10 illustre cette analyse.

L'action qui amène l'initialisation de l'arc (fermeture de la main réelle sur la corde) entraîne normalement l'initialisation de la flèche (puisque la main virtuelle doit logiquement être fermée). Cette dernière restera solidaire de l'arc tant que l'échappement n'aura pas eu lieu.

M_v : Main virtuelle ouverte

$\overline{M}_v$: Main virtuelle fermée

$M_v = \{p_v, i_v, m_v, a_v, o_v\}$

$\oplus$: opérateur booléen " ou exclusif "

$M = \{p, i, m, a, o\}$

p = pouce

i = index

m = majeur

a = annulaire

o = auriculaire

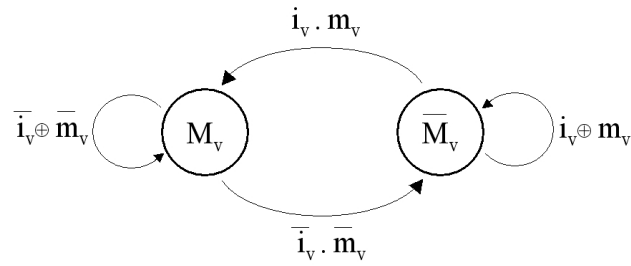


FIG. 4.9 – Diagramme d'États de la Main Virtuelle (Tir à l'Arc)

A : Arc

$A = \{A_r, A_i, A_t\}$

A_r : arc au repos

A_i : arc initialisé

A_t : arc tendu

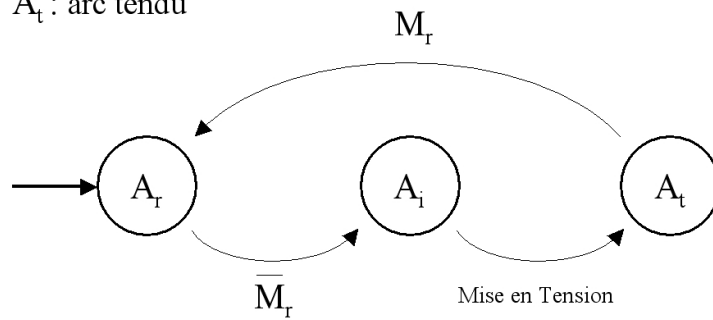


FIG. 4.10 – Diagramme d'États de l'Arc

Lorsqu'il aura été effectué, la flèche sera "en vol", dans un état qui n'est que transitoire. En effet, elle se fixera à la première collision détectée, et sera alors dans son état final. Ce n'est que lorsque la main virtuelle sera à nouveau fermée qu'elle retrouvera son état initial. Ce mécanisme est résumé par la figure 4.11

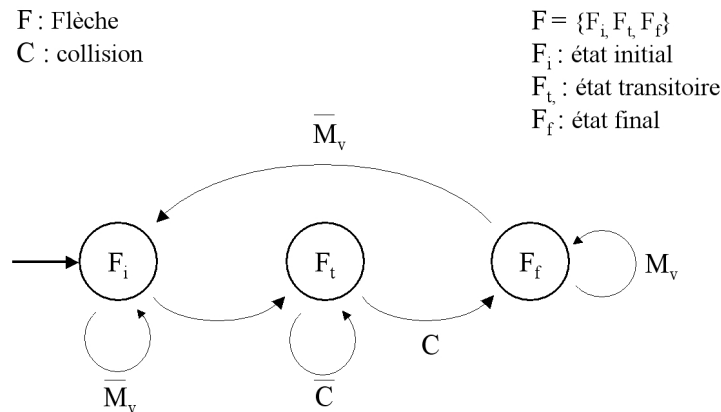


FIG. 4.11 – Diagramme d'États de la Flèche

4.3.3.2 Monde Virtuel

– Graphique :

L'environnement graphique est composé d'une skybox (cube texturé représentant un paysage de fond) dépeignant une chaîne de montagnes enneigées à l'arrière plan. La cible à atteindre est un parallélépipède sur lequel a été plaquée une texture représentant les cercles concentriques d'une cible réelle sur la face avant, et une texture de bois sur les autres faces. A droite et à gauche de la cible ont été placées une tente circulaire et une tour de garde moyenâgeuse. La tente est en-deçà du plan de la cible, donc plus proche du point de vue de l'utilisateur, alors que la tour est située au-delà, et en est donc plus éloignée. La flèche est composée d'un cylindre texturé de bois avec des anneaux de couleurs, d'une pointe assez grossière grise, et d'un empennage constitué de simples polygones avec une texture de plumes. La caméra, devant à la base représenter l'œil de l'utilisateur, est au départ placée à une hauteur de 1.60m et est dirigée vers la cible. Lorsqu'une flèche est initialisée, celle-ci apparaît à 4cm en dessous de la caméra (de "l'œil" de l'utilisateur) et est orientée en direction de la cible avant de s'ajuster conformément à la position des capteurs.

Cet environnement graphique riche, visible sur la figure 4.12, a été préféré à un environnement plus dépouillé par souci de réalisme d'une part, mais aussi parce qu'il donne à l'utilisateur des indices visuels de profondeur permettant d'estimer la taille et la distance

de la cible d'autre part. La présence d'objets de hauteur différente (tentes et tours) améliore l'identification et la perception de l'échelle des objets. Enfin, il a été pensé avec un souci d'évolutivité : l'ajout de la navigation dans le monde virtuel a été envisagé, il était donc préférable de pouvoir se déplacer dans un environnement riche plutôt que désert.



FIG. 4.12 – Capture d'écran de la simulation de tir à l'arc

– **Caractéristiques physiques :**

La flèche et la cible étant les deux seules entités virtuelles réellement importantes dans notre simulateur de tir à l'arc, ce sont les seules à disposer de caractéristiques physiques. Au moment de l'échappement, la vitesse initiale V_0 de la flèche est calculée en assimilant le système à un ressort, comme suit :

$$V_0 = \Delta \sqrt{\frac{k}{m}} \quad \text{avec :}$$

- Δ : amplitude du recul
- k : coefficient de raideur de la corde
- m : masse de la flèche

Une fois en vol, la flèche est soumise à une force de gravité de 9.8 ms^{-2} et on lance la procédure de détection de collision.

Pour celle-ci, la flèche est modélisée comme un simple segment de droite et la cible comme un quadrilatère représentant la face avant. A chaque frame (calcul de rendu), nous déterminons s'il y a intersection entre la ligne (la flèche) et le quadrilatère (la cible), non seulement en prenant en compte la position "actuelle" de la flèche (la position à la frame

courante), mais aussi sur les 25 positions intermédiaires de la flèche : entre sa position à la frame courante et celle à la frame précédente. Si une collision avec la cible est détectée à l'une des positions intermédiaires, la flèche est positionnée et stoppée à cette position. En effet, un unique test effectué sur la seule position courante de la flèche serait insuffisant : la vitesse de la flèche étant relativement élevée, il est très probable qu'à une frame donnée la flèche serait située devant la cible, et qu'à la frame suivante elle serait située derrière. La flèche "traverserait" alors la cible sans qu'aucune collision ne soit détectée (voir figure 4.13).

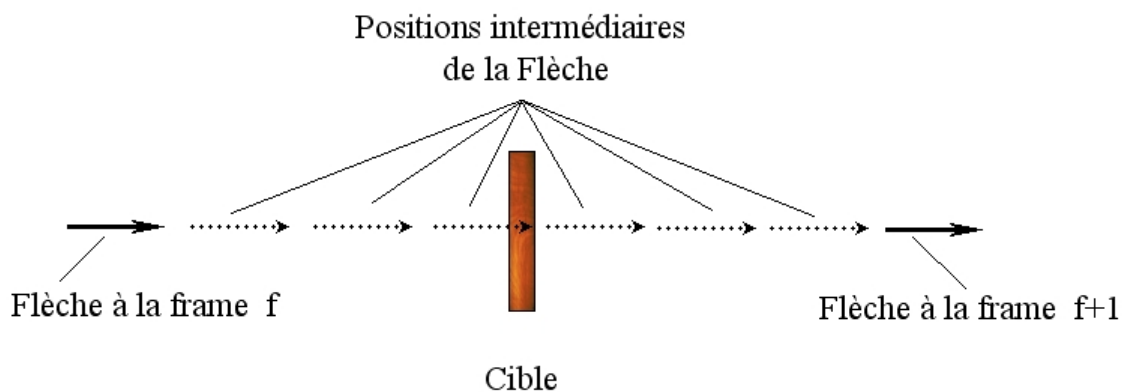


FIG. 4.13 – Détection de Collision Flèche-Cible : nécessité des "Positions intermédiaires"

– Sonore :

Pour les expériences, l'environnement sonore était dépouillé, c'est à dire qu'aucune ambiance sonore de fond n'était activée. En revanche, au moment de l'échappement et lorsqu'une collision était détectée (flèche-herbe ou flèche-cible), un son choisi aléatoirement dans la bibliothèque de fichiers "wav" correspondante était joué (4 fichiers pour l'ambiance sonore, 7 fichiers pour la collision avec la cible, 2 fichiers pour la collision avec le sol et 4 fichiers pour l'échappement) .

4.3.3.3 Modèles 3D

Les modèles 3D de la flèche et des tours de garde sont des modèles .3ds libres de droit, mis à disposition gratuitement sur Internet par leurs auteurs. La cible virtuelle a été créée directement en code OpenGL. Les autres modèles ont été conçus dans le cadre de projet-étudiants de niveau Master dont le but était l'apprentissage de l'outil de modélisation 3D Studio Max. La solution technique la plus simple et la plus rapide à l'époque pour intégrer dans le simulateur tous ces

objets 3D était de convertir directement les fichiers 3ds en code OpenGL natif grâce à un logiciel de conversion de formats de fichiers 3D nommé Polytrans.

La figure 4.14 illustre cette procédure d'intégration.

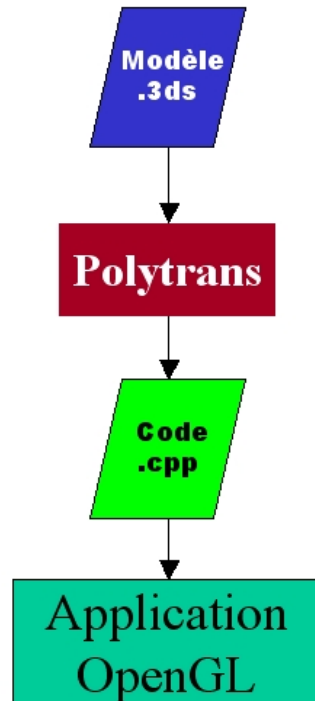


FIG. 4.14 – Schéma de la Procédure d'Intégration des Objets 3D

4.3.3.4 Arrière-plan : Skybox

Afin de renforcer l'immersion de l'utilisateur dans la simulation, il était indispensable de lui donner la sensation d'être situé dans un environnement complexe, ne se limitant pas aux quelques objets proches de lui. Afin de créer l'illusion d'un paysage d'arrière-plan réaliste, nous avons utilisé la technique dite de la "skybox". Cette technique consiste à plaquer des textures créées d'une manière bien spéciale sur les six faces internes d'un cube et à placer le point de vue de l'utilisateur à l'intérieur de celui-ci. Pour que l'illusion de paysage soit parfaite, il ne faut pas que les arêtes du cube soient visibles, que ce soit à cause d'un effet d'ombrage ou par discontinuité des textures.

Nous avons donc besoin de six textures : avant, droite, arrière, gauche, haut et bas, plaquées de la manière indiquée sur la figure 4.15.

Pour créer ces textures, il faut créer une scène 3D avec un logiciel de modélisation, placer la

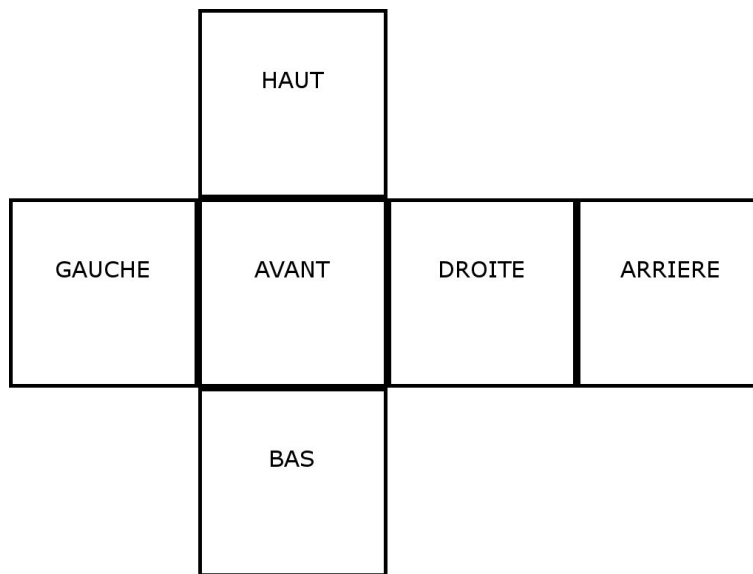


FIG. 4.15 – Théorie de la Skybox

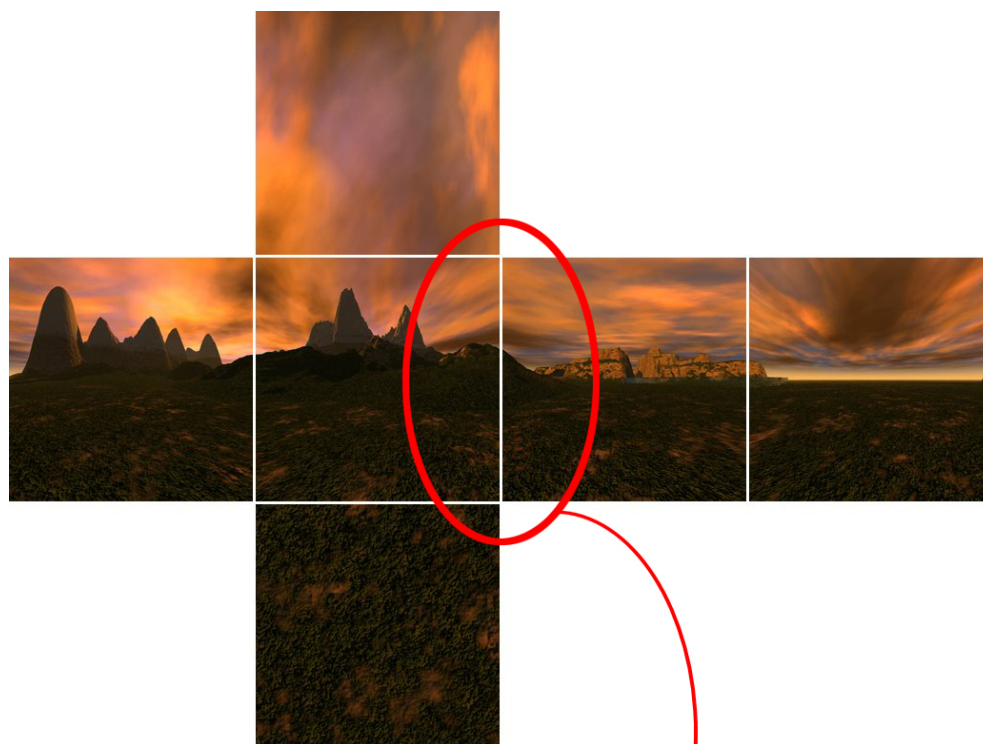
caméra à un endroit fixe, régler l'angle de vision de la caméra à 90° et faire un rendu de la scène 3D dans une texture carrée. On tourne alors la caméra de 90° vers la droite et on effectue de nouveau le rendu pour obtenir la texture "droite". On répète la procédure pour les autres faces afin d'obtenir les 6 textures désirées. Pour notre application, le logiciel Vue 4 Professional a été utilisé. La figure 4.16a illustre les textures obtenues tandis que la figure 4.16b est une capture d'écran du résultat final, la prise de vue est orientée vers une arête du cube.

Cette environnement cubique permettait également d'améliorer la visualisation du déplacement de la flèche lors de l'utilisation de la "caméra embarquée" (caméra positionnée sur la flèche durant son vol).

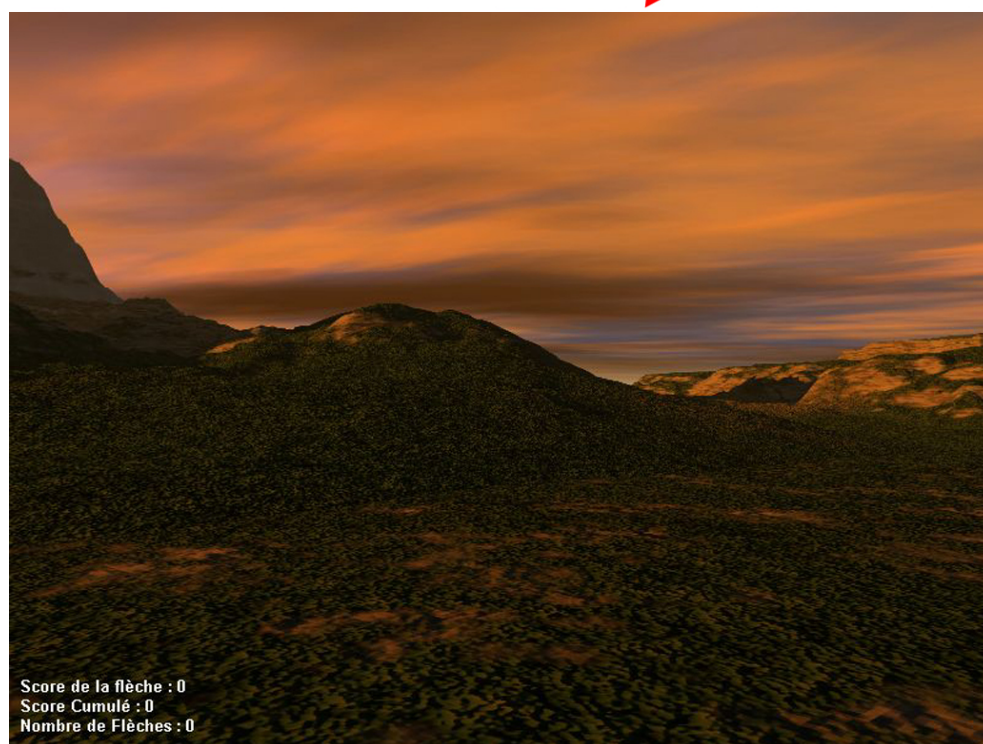
4.4 Expérimentations

4.4.1 Influence du coefficient de recul

Lorsque l'archer effectue la mise en tension dans le cas du tir à l'arc réel, la flèche étant fixée à la corde, elle coulisse vers l'arrière le long de l'arc. Ce mouvement de la flèche, vers l'arrière et consécutif à la traction de la corde, est visuellement perceptible par l'archer. Nous appelons ce phénomène le *recul visuel de la flèche*.



(a)



(b)

FIG. 4.16 – (a) Textures de la Skybox ; (b) Capture d'écran orientée vers une arête du cube

4.4.1.1 Objectif de l'expérience

Cette première expérience avait pour but d'étudier l'effet du recul visuel de la flèche sur les performances humaines lors d'une séance de tirs sur cible fixe en environnement virtuel. Pour cela, outre le score réalisé par chaque flèche, on enregistre la distance entre les deux capteurs de positions au moment de l'échappement, afin d'en déduire la force de traction exercée par l'utilisateur (voir la calibration ci-dessous). On prend également en compte le temps qui sépare l'initialisation de la flèche de l'échappement, ainsi que l'angle d'élévation du tir (l'angle par rapport à l'horizontale).

Pour connaître la force de traction, nous avons effectué la calibration suivante. On appelle l'*allonge* d'un arc la distance entre deux positions du point milieu de la corde, quand l'arc est bandé et quand l'arc est au repos. Pour différentes valeurs d'allonge (calculée dans la simulation grâce à la distance séparant les deux capteurs magnétiques), nous avons mesuré la force appliquée correspondante en utilisant la méthode illustrée sur la figure 4.17.

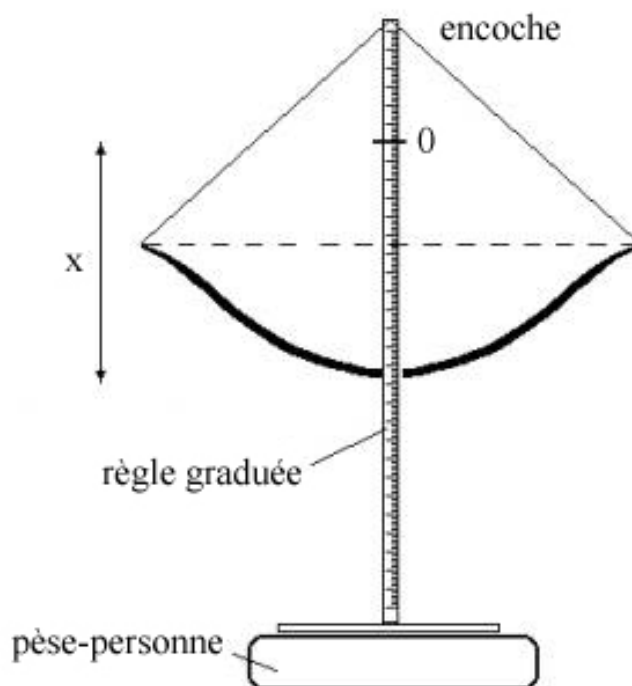


FIG. 4.17 – Méthode de Calibration

Les résultats de cette calibration sont illustrés sur la figure 4.18. On observe que les données de la calibration peuvent être représentées par une équation du troisième ordre. Selon des archers confirmés, le point d'inflexion (30 cm dans notre cas) correspond à l'amplitude de traction

la plus confortable. Cette traction représente pour notre cas une force appliquée d'environ 45N.

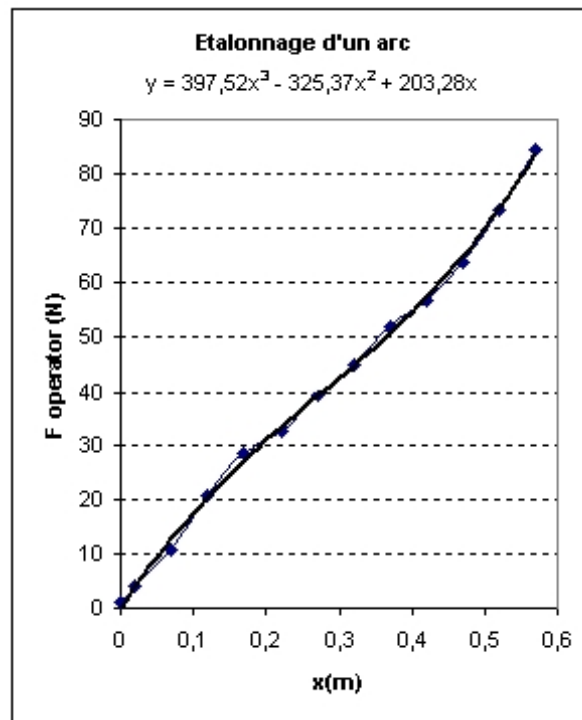


FIG. 4.18 – Force appliquée par l'utilisateur en fonction de l'allonge

4.4.1.2 Description de la tâche

La tâche consistait à essayer d'atteindre le centre de la cible située en environnement virtuel avec chaque flèche. Avant le début de l'expérience, une calibration du gant et des capteurs de position était effectuée. En ce qui concerne plus particulièrement la calibration du gant, le sujet devait ouvrir et fermer la main plusieurs fois de suite afin de calculer la valeur seuil d'ouverture de la main utilisée pour détecter le lâcher de la corde. La calibration des capteurs de position s'effectuait quant à elle avec le sujet positionné en face de l'écran et visant le centre de ce dernier. Enfin, chaque sujet avait droit à 4 flèches non comptabilisées, une dans chacune des conditions expérimentales (explicitées au paragraphe 4.4.1.4), sans cible affichée, afin de pouvoir s'approprier le système avant le début de l'expérience proprement dite.

Le sujet se tenait à 2 mètres d'un écran rétro-projeté de 2m x 2.5m et tenait un arc réel comme en tir réel. La cible, de diamètre égal à 80cm, était positionnée dans l'environnement virtuel à une distance de 18 mètres.

4.4.1.3 Monde virtuel

Le monde virtuel qui a été utilisé lors de cette expérimentation est celui qui a été décrit précédemment, aucun élément nouveau n'a été ajouté.

4.4.1.4 Protocole expérimental

Au total, 24 sujets volontaires, droitiers et novices en réalité virtuelle comme en tir à l'arc ont participé à cette expérience. Chaque sujet a réalisé, chacun dans un ordre différent, la tâche décrite ci-dessus dans les conditions suivantes :

- Aucun recul visuel (Condition 1) : la flèche virtuelle n'effectuait aucun mouvement vers l'arrière lorsque la corde était tirée, sa position était fixe.
- Recul visuel moitié moindre que le recul réel (Condition 2) : la flèche virtuelle effectuait un mouvement vers l'arrière moitié moins important que le recul réel normal.
- Recul visuel correspondant au recul réel (Condition 3) : la flèche virtuelle effectuait un mouvement vers l'arrière conforme à celui qu'une flèche aurait effectué en réel.
- Recul visuel double du recul réel (Condition 4) : la flèche virtuelle effectuait un mouvement vers l'arrière deux fois plus important que celui qu'une flèche réelle aurait effectué.

La différence d'ordre des conditions s'explique là encore par la nécessité d'éviter le transfert d'apprentissage entre celles-ci.

Cette expérience s'est déroulée en 4 sessions (1 session par condition expérimentale) constituée de 15 tirs, avec 30 secondes de récupération entre chaque série de 5, et 1 minute de repos entre les sessions. Au final, chaque sujet a donc tiré 60 flèches. A noter que le mode de vue choisi tout au long de cette expérience était celui de la caméra statique afin de ne pas s'éloigner davantage du tir réel.

4.4.1.5 Analyse et discussion des résultats

Dans le but d'évaluer la performance de l'utilisateur, les données suivantes ont été enregistrées pour chaque condition expérimentale : le score de la flèche (entre 0 et 1000), la force de traction de la corde (calculée à partir de la distance entre les deux capteurs de positions), le temps de visée (le temps entre l'initialisation et le lâcher d'une flèche) et la valeur de l'angle d'élévation. Une fois recueillies, ces données ont été analysées à l'aide de la méthode ANOVA.

Score : L'ANOVA a révélé un effet significatif du recul visuel de la flèche sur les performances de l'utilisateur ($F(3,23) = 5.84, p < 1.0^{-4}$).

Les résultats montrent que les conditions 1 (aucun recul), 2 (demi recul) et 3 (recul normal) convergent vers des scores statistiquement équivalents (figure 4.19). Les scores moyens enregistrés sont d'environ 327 (Std : 206) pour la condition 1, de 326 (Std : 218) pour la condition 2, et de 363 (Std : 217) pour la condition 3.

En revanche, nous observons que pour la condition 4, où le recul est double du recul normal, les performances des utilisateurs ont chuté, avec un score moyen d'environ 164 (Std : 119). Cela montre que les sujets ont éprouvé des difficultés à correctement viser la cible.

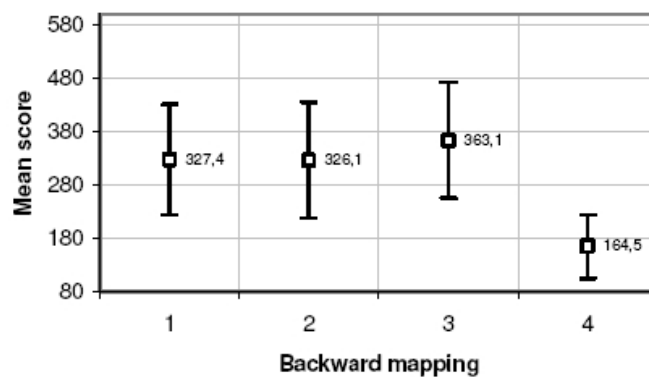


FIG. 4.19 – Score moyen par Condition

Force de traction : Les résultats illustrés par la figure 4.20 indiquent que le recul visuel de la flèche a un effet certain sur la force de traction appliquée ($F(3,23) = 5.55$, $p < 1.0^{-4}$). Bien que pour les trois premières conditions les forces de traction enregistrées soient sensiblement les mêmes avec respectivement 43.38 N (Std : 10.11), 45.24 N (Std : 10.74) et 44.03 N (Std : 9.37), nous avons observé une baisse significative de cette force pour la condition 4 (environ 41.23 N - Std : 7.47).

Temps de visée : Le recul visuel de la flèche peut aussi avoir une influence sur le temps de visée, comme le montrent les résultats obtenus sur la figure 4.21 ($F(3,23) = 15.48$, $p < 1.0^{-4}$). Une fois encore, les valeurs pour les trois premières conditions sont relativement équivalentes, avec respectivement 5.79 (Std : 2.9), 5.82 (Std : 2.2) et 5.62 sec. (Std : 2.5). En revanche, la dernière condition de recul visuel engendre un besoin d'ajustement de la visée plus important qui se traduit par un temps de visée sensiblement plus long (6.38 sec. - Std : 2.8).

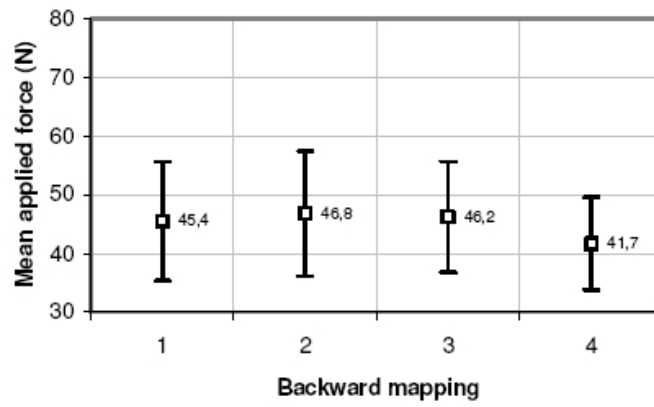


FIG. 4.20 – Force moyenne appliquée par Condition

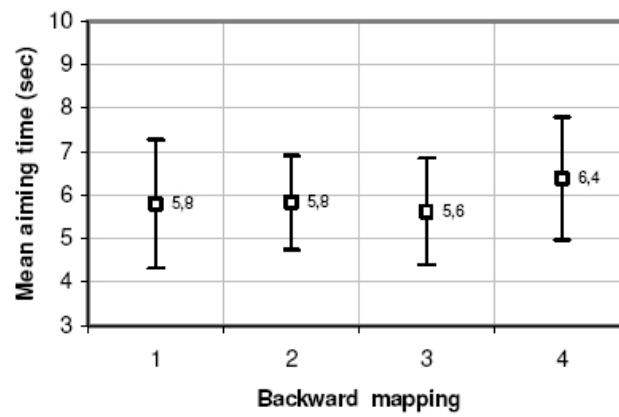


FIG. 4.21 – Temps de visée moyen par Condition

Angle d'élévation : L'angle d'élévation du tir est également significativement affecté par le coefficient de recul visuel de la flèche ($F(3,23) = 3.39, p < 1.0^{-4}$). Si l'on constate une augmentation constante de l'angle d'élévation pour les trois premières conditions, avec un angle moyen de respectivement -3.3 (Std : 0.9), -3.0 (Std : 1.84) et -2.5 (Std : 1.18) degrés, elle est encore plus flagrante pour la condition 4, avec un angle d'élévation moyen d'environ -1.5 degré (Std : 1.42). Ces résultats montrent que les sujets ont essayé de compenser la plus faible vitesse initiale de la flèche, due à une force de traction de la corde plus faible, par une trajectoire moins plate, en visant plus haut.

Finalement, ces résultats montrent qu'un coefficient de recul visuel double du recul normal (condition 4) produit une gêne se traduisant par un temps de visée plus long pour un résultat final moins élevé. La force de traction appliquée étant moindre dans la condition 4, on peut en déduire qu'il y a bien une dominance de l'information visuelle sur l'information haptique puisque les utilisateurs se basent davantage sur le recul visuel de la flèche que sur la sensation kinesthésique produite par la corde.

En ce qui concerne la condition pour laquelle il n'y a aucun recul visuel (condition 1), on peut expliquer les bonnes performances par le fait que, la flèche étant fixe dans l'espace, elle fournit un point de repère pour la visée : l'utilisateur tire fortement sur la corde pour que la flèche ait une trajectoire très rectiligne, aligne la pointe de la flèche avec un point situé sur le haut de la cible, et procède à l'échappement. Cette stratégie est plus difficilement applicable si la flèche est agitée de petits mouvements au gré des tremblements de l'utilisateur.

Dans le cas où le recul visuel de la flèche correspond au recul réel (condition 3), l'utilisateur se trouve dans une situation "normale", il n'y a pas de décalage entre son action sur la corde et le résultat visuel qu'elle implique, le geste est donc plus naturel que pour les autres conditions, ce qui explique que ce soit dans cette condition que l'on observe les meilleures performances.

Les bons résultats obtenus pour la condition où le recul visuel de la flèche est moitié moindre que le recul réel (condition 2) pourraient paraître surprenants, mais l'explication est en fait très simple : ce coefficient n'implique pas un décalage suffisamment important par rapport aux conditions réelles pour gêner les utilisateurs. Beaucoup d'entre eux ont confirmé dans leurs commentaires qu'ils considéraient cette condition comme étant équivalente à la condition 3.

4.4.2 Influence des ALID visuelles

4.4.2.1 Objectif de l'expérience

Dans cette seconde expérience, nous nous intéressons aux effets sur la performance humaine de l'affichage d'informations prédictives (affichage du point d'impact de la flèche sur la cible ou le sol, et/ou du score qui en résulterait) d'une part, et à ceux issus de l'utilisation du mode de vue dynamique (caméra embarquée) d'autre part.

De plus, nous souhaitons évaluer l'efficacité de ces aides logicielles pour l'apprentissage.

4.4.2.2 Description de la tâche

Comme pour la première expérience, le but des sujets était d'essayer d'atteindre le centre de la cible avec chaque flèche. Les mêmes techniques de calibration du gant et des capteurs de position ont été utilisées. Là encore, les sujets avaient droit à 4 flèches non comptabilisées afin de s'habituer au système.

L'expérience était constituée de deux sessions consécutives, la première dite d'"Entraînement" et la seconde de "Test", chacune consistant en 30 tirs, avec 30 secondes de récupération entre chaque série de 5, et 1 minute de repos entre les deux sessions.

4.4.2.3 Monde virtuel

Le monde virtuel utilisé pour cette expérience est sensiblement le même que précédemment, à l'exception des modifications suivantes. Lors de l'utilisation de l'aide logicielle "PointRouge", un point rouge situé dans l'environnement indique en temps réel où la flèche va finir sa course si l'échappement est effectué. Ce point est vert si une collision avec la cible est pressentie.

Dans le même ordre d'idée, un nombre situé entre 0 et 1000 correspondant au score prévu en cas d'échappement est affiché à peu près au centre de l'écran lorsque l'aide logicielle "Score-Prédictif" est activée. Cette indication numérique est affichée en vert pour un score strictement positif et en rouge sinon. Les deux aides peuvent être affichées simultanément ("PointRouge et ScorePrédictif"), comme sur la capture d'écran illustrée par la figure 4.22.

La dernière modification concerne l'ajout du mode de vue "CaméraEmbarquée". Lorsque cette aide logicielle est activée, la caméra est placée juste sur la flèche durant son vol et jusqu'à ce qu'une nouvelle flèche soit initialisée.



FIG. 4.22 – Point d'impact et Score prédictifs

4.4.2.4 ALID n°1 : PointRouge

Cette Aide Logicielle est inspirée des visées laser, à la base installées sur certains types de fusil de sniper et maintenant répandues comme dispositif de pointage pour conférences. La particularité du laser est qu'il émet un faisceau rectiligne. Or, même si la vitesse de propulsion d'une flèche peut s'élever jusqu'à 350 km/h (avec un arc à poulies), une balle de carabine gros calibre peut atteindre 850 m/s, soit plus de 3000 km/h. Ainsi, une flèche n'aura une trajectoire rectiligne que sur une très faible distance, beaucoup plus courte que dans le cas d'une balle de fusil. Ainsi, le point d'impact avec le sol ou la cible, sur lequel le point rouge sera affiché, doit être calculé avec précision en tenant compte de l'aspect parabolique de la trajectoire de la flèche : cette trajectoire ne pourra pas être assimilée à une ligne droite par rapport à la direction du tir.

La structure de l'ALID PointRouge est représentée dans le tableau 4.1.

Les Paramètres de l'ALID "PointRouge" sont :

- **PID** → **Modèle PID** : $S_f G_d C_f$.
- **Type** → **Sous-Type** : Unique Objet 2D.
- **État** → **État_PointRouge** : Soit $P_{collision}$ la position de la première collision de la flèche avec le sol ou la cible si l'échappement était effectué au moment du calcul, on a alors :

$$P_{PointRouge} = P_{collision}$$

Propriété	Valeur
Nom	PointRouge
But	"Réalisation"
PID	"Lancer"
Type	"Visuel"
Description	Cette aide consiste en l'affichage d'un disque rouge à l'endroit précis où la flèche irait se planter (sol ou cible) si elle était lâchée à l'instant présent. Le disque devient vert si l'impact prévu est dans la cible, et redevient rouge lorsque l'impact pré-calculé ressort de la Cible. Le disque doit toujours être visible, il est donc affiché en superposition des autres Objets.
Genre	"Dynamique"
Pré-condition	aucune
Post-condition	aucune
État	État_PointRouge
Fonctionnement	PointVert

TAB. 4.1 – Structure de l'ALID "PointRouge".

L'Orientation du disque doit toujours être telle que sa normale pointe vers la caméra, de manière à être toujours complètement visible de l'utilisateur.

Le tableau 4.2 détaille la Modification PointVert. On note P_C l'ensemble des points de l'espace considérés dans la Cible.

Composante	Valeur
Nom	PointVert
Type	"Discret"
Caractéristique	Couleur
Effet	Couleur = Vert
Pré-condition	$P_{PointRouge} \in P_{Cible}$
Post-condition	$P_{PointRouge} \notin P_{Cible}$

TAB. 4.2 – Composantes de PointVert.

4.4.2.5 ALID n°2 : ScorePrédictif

Cette Aide s'inscrit dans la continuité de la précédente. En effet, puisque nous pouvons pré-calculer le point d'impact de la flèche si l'échappement était réalisé au moment du calcul, nous pouvons également en déduire le score que réaliserait la flèche. C'est ce score prédictif que nous affichons en temps réel.

La structure de l'ALID ScorePrédictif est donnée par le tableau 4.3.

Propriété	Valeur
Nom	ScorePrédictif
But	"Réalisation"
PID	"Lancer"
Type	"Visuel"
Description	Cette aide consiste en l'affichage textuel du score que réaliserait la flèche si elle était lâchée à l'instant présent. Le texte est affiché en rouge si le point d'impact estimé est hors de la cible, et en vert si au contraire le point d'impact pré-calculé est à l'intérieur de la Cible. Comme le PointRouge, le score doit également toujours être visible, il est donc aussi affiché en superposition des autres Objets.
Genre	"Statique"
Pré-condition	aucune
Post-condition	aucune
État	État_Score
Fonctionnement	ScoreVert

TAB. 4.3 – Structure de l'ALID "ScorePrédictif".

Les Paramètres de l'ALID "ScorePrédictif" sont :

- **PID** → **Modèle PID** : $S_f G_d C_f$.
- **Type** → **Sous-Type** : Texte.
- **État** → **État_Score** : Au centre de la fenêtre de l'application.

La Modification ScoreVert est illustrée par le tableau 4.4.

4.4.2.6 ALID n°3 : CaméraEmbarquée

Cette Aide s'inspire des caméras placées sur les missiles Air-Sol produits, entre autres, par les États-Unis d'Amérique. Ici, on place une "caméra" sur la flèche, et c'est cette vue qui est utilisée jusqu'à ce qu'une nouvelle flèche soit initialisée. Outre la trajectoire de vol de la flèche,

Composante	Valeur
Nom	ScoreVert
Type	"Discret"
Caractéristique	Couleur
Effet	Couleur = Vert
Pré-condition	$P_{collision} \in P_C$
Post-condition	$P_{collision} \notin P_C$

TAB. 4.4 – Composantes de ScoreVert.

cette caméra permet également, le cas échéant, de mieux visualiser le point d'impact sur la cible. On trouvera la structure de l'ALID CaméraEmbarquée détaillée dans le tableau 4.5.

Propriété	Valeur
Nom	CaméraEmbarquée
But	"Apprentissage"
PID	"Lancer"
Type	"Visuel"
Description	Cette aide consiste en l'affichage textuel du score que réaliserait la flèche si elle était lâchée à l'instant présent. Le texte est affiché en rouge si le point d'impact estimé est hors de la cible, et en vert si au contraire le point d'impact pré-calculé est à l'intérieur de la cible. Comme le PointRouge, le score doit également toujours être visible, il est donc aussi affiché en superposition des autres Objets.
Genre	"Dynamique"
Pré-condition	La Flèche en cours a été tirée
Post-condition	Une nouvelle Flèche est Initialisée
État	État_Caméra
Fonctionnement	aucune

TAB. 4.5 – Structure de l'ALID "CaméraEmbarquée".

Les Paramètres de l'ALID "CaméraEmbarquée" sont :

- **PID** → **Modèle PID** : $S_f G_d C_f$.
- **Type** → **Sous-Type** : Caméra.
- **État** → **État_Score** : Soit *OffsetPosition* le décalage de la Caméra sur les axes X,Y,Z par rapport à la position de la flèche, alors on a :

$$P_{Camra} = P_{Flèche} + OffsetPosition$$

$$O_{Camra} = O_{Flèche}$$

4.4.2.7 Protocole expérimental

Nous avons fait appel à 25 sujets volontaires, droitiers et novices tant en réalité virtuelle qu'en tir à l'arc. Ils étaient séparés en 5 groupes de 5 sujets, comme illustré dans le tableau 4.6. Chaque groupe était associé à des conditions expérimentales différentes pour la session d'entraînement, mais équivalentes pour la session de test, ce qui devait nous permettre d'évaluer le transfert d'apprentissage entre les deux sessions.

Groupe n°	Sessions	
	Entraînement	Test
1	Aucune aide	Aucune aide
2	PointRouge (caméra statique)	Aucune aide
3	PointRouge et ScorePrédictif (caméra statique)	Aucune aide
4	ScorePrédictif (caméra statique)	Aucune aide
5	CaméraEmbarquée	Aucune aide

TAB. 4.6 – Conditions expérimentales par Groupe.

Ainsi, pour la session d'entraînement, chaque groupe était associé à l'une des conditions suivantes :

- Pas d'affichage d'informations prédictives, Caméra Statique (Condition 1)
- Affichage du point d'impact prédictif, Caméra Statique (Condition 2)
- Affichage du point d'impact prédictif et du score prédictif, Caméra Statique (Condition 3)
- Affichage numérique du score prédictif, Caméra Statique (Condition 4)
- Pas d'affichage d'informations prédictives, Caméra Dynamique (Condition 5)

Pour la session de test, aucune aide logicielle n'était disponible (Pas d'affichage d'informations prédictives et caméra statique), tous les groupes se retrouvant donc, pour cette session, dans les mêmes conditions expérimentales, équivalentes à celles de la Condition 1 de la session d'entraînement. On notera alors ces conditions comme suit : condition 1b, 2b, 3b, 4b, 5b.

4.4.2.8 Analyse des résultats et discussion

Pour évaluer la performance des utilisateurs et le transfert d'apprentissage spécifique à chacune des aides logicielles, nous avons enregistré les données suivantes, qui sont les mêmes que pour l'expérience précédente : le score de la flèche, la force de traction de la corde, le temps de visée et la valeur de l'angle d'élévation. Une fois encore, ces données ont été analysées à

l'aide de la méthode ANOVA. Nous n'avons pas observé de différence significative entre les différentes conditions concernant la force de traction de la corde, le temps de visée et la valeur de l'angle d'élévation. Nous avons cependant pu mettre en évidence l'effet des aides logicielles sur la performance de l'utilisateur en terme de score.

Effet de l'affichage d'informations prédictives :

– Session d'entraînement

L'analyse de variance a mis en évidence l'impact significatif de l'affichage d'informations prédictives sur la performance humaine ($F(3,4) = 5.24$, $p < 1.0^{-4}$). Statistiquement, les scores moyens des conditions 2 (point d'impact prédictif), 3 (score prédictif) et 4 (point d'impact prédictif et du score prédictif) sont équivalents (Figure 4.23). Les valeurs enregistrées sont d'environ 696 (Std : 136) pour la condition 2, de 617 (Std : 129) pour la condition 3 et de 585 (Std : 185) pour la condition 4. En comparaison, le 351 (Std : 64) réalisé durant la condition 1 (aucune aide logicielle) apparaît nettement inférieur. Cela démontre l'efficacité de l'affichage d'informations prédictives pour la réalisation d'une telle tâche.

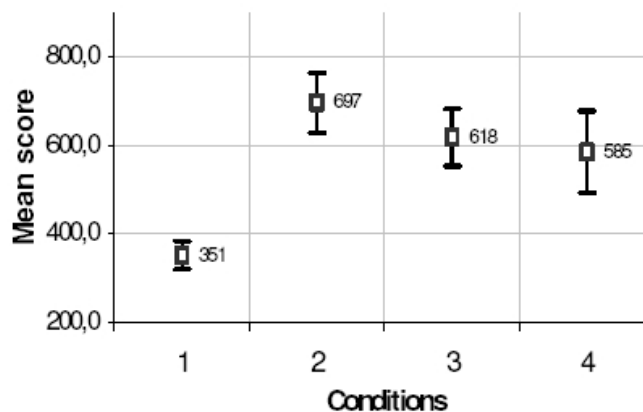


FIG. 4.23 – Moyenne de Score en Session d'Entraînement (Conditions 1 à 4)

– Session de test

Dans cette session, qui suit la session d'entraînement, l'affichage d'informations prédictives a été désactivé. L'ANOVA ne révèle aucune différence significative ($F(3,4) = 0,54$) entre les conditions 1b, 2b, 3b et 4b. Les scores moyens enregistrés sont d'environ 462 (Std : 78) pour la condition 1b, de 429 (Std : 212) pour la condition 2b, de 439 (Std : 145) pour la condition 3b et de 527 (Std : 205) pour la condition 4b (Figure 4.24). Ces résultats

démontrent qu'aucun transfert d'apprentissage n'apparaît après utilisation de l'affichage d'informations prédictives.

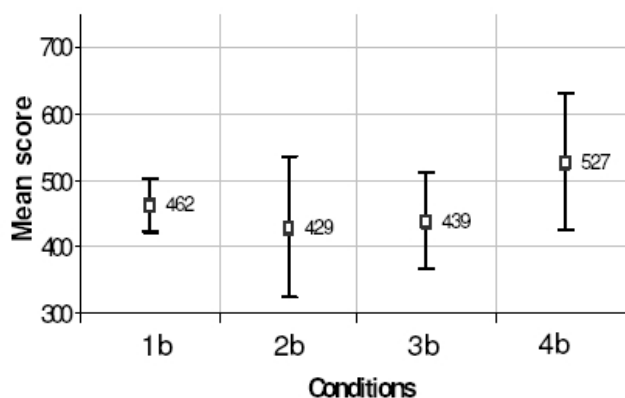


FIG. 4.24 – Moyenne de Score en Session de Test (Conditions 1b à 4b)

Effet du mode de vue en caméra embarquée :

– Session d'entraînement

Les résultats enregistrés durant cette session, et représentés sur la figure 4.25 montrent que la condition 5 (caméra embarquée) conduit à un meilleur score moyen (environ 471 ; Std : 158) que pour la condition 1 (aucune aide logicielle) (351 ; Std : 64), mais cet effet reste très inférieur à celui observé précédemment dans les conditions 2, 3 et 4. De ce fait, le mode de vue en caméra embarquée n'est pas aussi efficace pour la réalisation d'une telle tâche que l'affichage d'informations prédictives.

– Session de test

En revanche, les scores enregistrés durant cette session de test étaient d'environ 462 (Std : 78) pour la condition 1b et 581 (Std : 103) pour la condition 5b (figure 4.25) : ceci démontre l'efficacité du mode de vue en caméra embarquée pour le transfert d'apprentissage lors de ce type de tâche.

Ces résultats mettent en lumière la dualité des aides logicielles. Elles peuvent être utilisées pour aider à la réalisation d'une tâche ou pour permettre un apprentissage plus rapide d'une tâche.

L'affichage du point d'impact prédictif et celui du score prédictif sont des aides à la réalisation

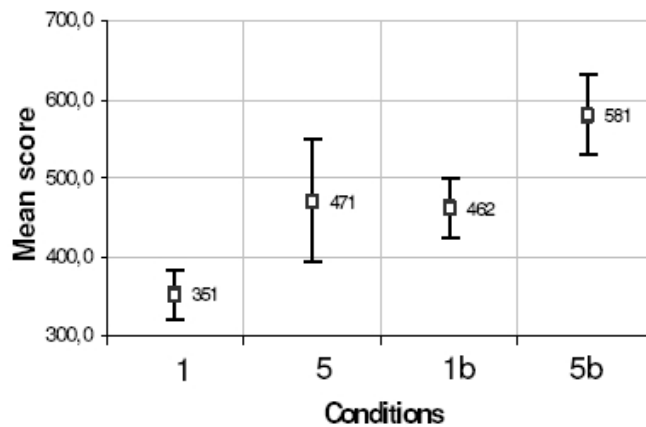


FIG. 4.25 – Moyenne de Score en Session d’Entraînement (Conditions 1 et 5) et de Test (Conditions 1b et 5b)

de la tâche : elles ont pour effet d’améliorer très nettement les performances de l’utilisateur quand elles sont activées, mais une fois désactivées, l’utilisateur retrouve son niveau de base, aucun transfert d’apprentissage ne s’étant effectué.

La caméra embarquée se classe plutôt dans la catégorie des aides pour l’apprentissage : quand elle est activée, la performance de l’utilisateur ne s’en trouve que légèrement améliorée, mais l’utilisateur apprend la tâche grâce à elle et conserve un bon niveau une fois désactivée.

La différence dans le processus mental lié à ces aides logicielles peut expliquer cet écart concernant le transfert d’apprentissage. Dans le cas des informations prédictives utilisées ici, l’utilisateur ne fait qu’ajuster son geste pour faire correspondre le point d’impact (ou le score) à une position (ou à une valeur) qu’il juge satisfaisante. Mentalement, il ne fait que déplacer un point dans l’espace, sans s’occuper de l’interface lui permettant de déplacer ce point.

L’utilisation de la caméra embarquée, en revanche, nécessite un effort intellectuel plus important. Tout d’abord parce que l’utilisateur n’a pas un retour en temps réel sur l’effet de son geste, mais plutôt une information visuelle plus ou moins détaillée sur les conséquences de son geste tel qu’il était au moment de l’échappement. Par rapport à la caméra statique, ce mode de vue permet de mieux visualiser la trajectoire que la flèche a suivie, et d’identifier *a posteriori* le facteur d’erreur (trop haut, trop à droite, etc.). Cette identification de l’erreur permet ensuite à l’utilisateur de savoir comment modifier son geste **pour le prochain tir** (et non pas pour le tir en cours).

4.5 Conclusion

Après avoir justifié l'exploitation du tir à l'arc comme activité comportant une interaction dynamique de Lancer pour l'évaluation de nos Aides Logicielles pour l'Interaction Dynamique, nous avons expliqué comment nous avons conçu notre simulation. Une première expérimentation nous a permis de valider cette application, en mettant en exergue le besoin de ne pas trop perturber l'utilisateur dans sa perception du tir. Les résultats obtenus lors de la seconde expérimentation ont validé le protocole expérimental proposé à la fin du chapitre 3, et ont mis en valeur l'apport des Aides Logicielles développées dans le cadre de ce type de Lancer. Enfin, ils ont démontré qu'il existait bien deux types d'ALID : celles destinées à la réalisation de la tâche, et celles permettant un meilleur apprentissage de la tâche.

Nous allons maintenant nous pencher sur l'utilisation d'ALID dans le cadre d'une activité de Lancer impliquant un Geste Continu dans un but complexe : le Bowling.

Chapitre 5

Application au Lancer Continu : Le Bowling

5.1 Introduction

Le Bowling est un jeu dans lequel les joueurs essayent de marquer des points en faisant rouler une boule le long d'une surface plane pour renverser des objets appelés des quilles. Nous débuterons ce chapitre par une énumération des aspects du bowling que nous avons jugés pertinents pour l'évaluation de nos Aides Logicielles pour l'Interaction Dynamique. Comme pour les précédentes applications que nous avons développées, nous expliquerons comment nous avons conçu notre simulation de bowling, et nous terminerons alors par la description de l'expérimentation qui a été menée afin de valider l'application et d'évaluer une première ALID destinée à ce sport.

5.2 Justification

En tant qu'activité comportant des interactions dynamiques, et outre le fait qu'il s'agisse d'un lancer d'objet impliquant un Geste Continu réalisé dans un but complexe, le bowling possède plusieurs aspects de choix qui nous ont incités à nous intéresser à ce sport pour l'évaluation de nos aides logicielles.

Pour commencer, la boule, une fois lancée, n'évolue spatialement que sur deux dimensions. En effet, son altitude va être fixe puisqu'elle va rouler sur le sol, le long de la piste. Par rapport au lancer de balle développé dans le chapitre 3, l'utilisateur va donc ignorer la composante verticale pour la réalisation de la tâche. De plus, du fait de ce contact permanent avec le sol, les effets de rotation imprimés à la boule auront une influence très élevée sur sa trajectoire (beau-

coup plus importante que lorsque les seuls frottements en présence sont les frottements de l'air). Les trajectoires courbes qui résultent de l'utilisation d'effets de rotation sur la boule sont d'ailleurs primordiales dans ce sport, et elles sont difficiles à maîtriser. Ainsi, et même si, comme nous le verrons plus tard, nous ne les avons pas encore implémentées dans notre simulation, cet aspect du Bowling a retenu notre attention et nous pensons qu'il est possible de faciliter la manière dont les débutants appréhendent ce type de lancer de boule. Enfin, en plus du positionnement de départ en triangle des 10 quilles, il existe plusieurs configurations type de deux ou plusieurs quilles, qui résultent d'un premier lancer imparfait. Ces cas de figure classiques sont autant d'exercices potentiels pour lesquels nous pouvions vérifier l'apport de nos aides logicielles.

5.3 Conception et développement du système

5.3.1 Analyse et modélisation de l'interaction

Il y a plusieurs types de Bowling, et la première forme connue date de l'Égypte antique. Le Bowling le plus connu aujourd'hui est la version américaine à 10 quilles. Cette version se pratique tant au niveau amateur (dans le monde entier) que professionnel (principalement aux USA). La figure 5.1 indique le positionnement relatif des quilles, ainsi que leur poids et dimensions.

Une quille mesure ainsi 15 pouces, soit 38.1 cm de haut, pour un poids variant de 3lb 6oz à 3lb 10oz, ou 1.530 Kg à 1.643 Kg.

La boule a une circonférence de 27 pouces, ou 68.6 cm, et un diamètre de 8.62 pouces, soit 21.8 cm. Son poids est compris entre 6 et 16 lbs, autrement dit entre 2.72 et 7.26 Kg. En moyenne, la vitesse de la boule au moment du lancer est comprise entre 17 et 19 mph (de 27.353 à 30.571 kmh). A cause des frictions dues au roulement sur la piste, la boule aura perdu entre 3 et 3.5 mph (de 4.827 à 5.6315 kmh) au moment de toucher la première quille.

Les joueurs débutants ont tendance à utiliser une trajectoire de boule rectiligne, tandis que les joueurs professionnels préfèrent les trajectoires dites "en crochet", c'est à dire avec une courbe ramenant la boule de l'extérieur de la piste vers le centre dans les derniers mètres précédant les quilles (voir la figure 5.2).

N'importe qui peut lancer une boule avec une trajectoire rectiligne et obtenir un Strike ¹, mais ce style de lancer limitera la plupart des personnes à une moyenne de 180 points. Le lan-

¹un Strike consiste à renverser les 10 quilles avec la première boule

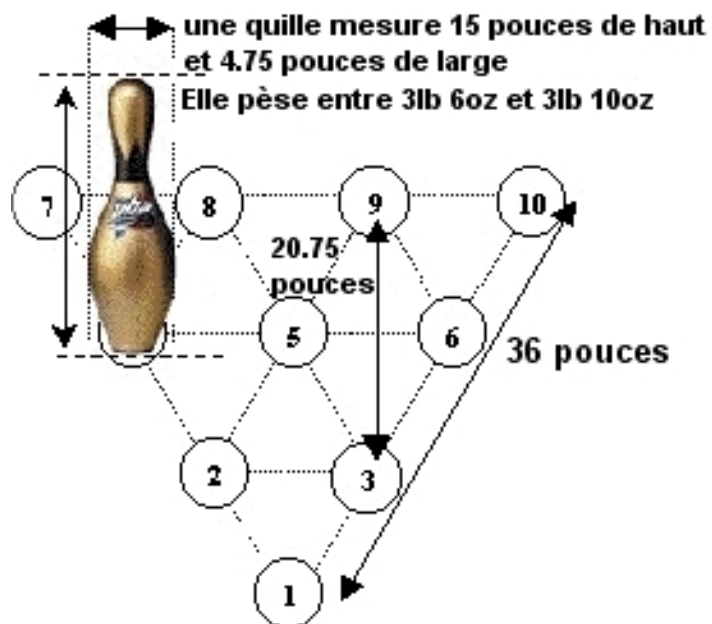


FIG. 5.1 – Positionnement relatif des quilles



FIG. 5.2 – Trajectoire en crochet

cement d'une boule en crochet est une façon de marquer davantage.

Il n'y a rien de mauvais en soi avec les trajectoires rectilignes. Presque tout le monde a commencé le Bowling en lançant une boule droite, ne serait-ce que parce que les autres styles de lancer exigent un certain temps d'apprentissage et de pratique. La question survient donc lorsque l'on veut avoir une moyenne dépassant les 180 points. Une boule lancée droite en plein milieu de la piste aboutit régulièrement à un split (voir la figure 5.3)



FIG. 5.3 – Split 7-10

L'autre facteur limitant avec une boule rectiligne est la déviation de la boule : la boule a tendance à "pousser" les quilles sur son passage. Or les dix quilles pèsent ensemble plus de 30 livres (13.6 Kg) et, comme la boule ne peut pas peser plus de 16 livres (7.26 Kg), la boule sera significativement déviée au fur et à mesure des collisions avec les Quilles, malgré la vitesse initiale. Une fois déviée, il lui devient plus difficile de renverser les dix quilles, et l'on se retrouve alors généralement avec des quilles centrales encore debout. De ce fait, une trajectoire rectiligne avec une bonne visée amène une moyenne de score de 180, que l'on appelle un "spare-game", car si l'on ne fait que des spares, on atteint environ 180-190 points par partie.

5.3.1.1 Angle d'Attaque

Une étude a été menée il y a plusieurs années pour trouver le meilleur moyen de réaliser un strike. Afin de rendre les résultats valides et répétables, les expérimentateurs ont mis en place un système de rampes (placées à différents endroits de la piste, pas uniquement à l'entrée) et ont lancé des "boules droites" (boules ayant une trajectoire rectiligne) vers les quilles. Ils ont découvert que l'angle avec lequel la boule roulait vers les quilles était très important : avec un

angle optimal, ils pouvaient faire strike sur strike. Cependant, le point clé de cette découverte était que cet angle d'attaque optimal ne pouvait être obtenu avec une boule ayant une trajectoire rectiligne lancée depuis l'entrée de la piste, mais uniquement avec une trajectoire en crochet. Pour lancer une "boule en crochet" (boule ayant une trajectoire en crochet) le joueur doit réaliser un lâcher spécial, en sortant le pouce de la boule et en utilisant les doigts pour lui transmettre une rotation latérale.

La boule va commencer par glisser le long de la piste, grâce à l'huile présente sur les planches, avant que la rotation ne finisse par rabattre la boule vers les quilles. Étant donné que la boule roule "à travers" les quilles au lieu de les pousser, comme dans le cas d'une boule droite, elle subira moins de déviations et renversera plus facilement les quilles centrales.

Nous reviendrons sur la manière dont la boule et les quilles interagissent pour réaliser un strike, mais commençons plutôt par analyser la physique mise en oeuvre dans ce sport.

5.3.1.2 Considérations énergétiques

Selon la hauteur de laquelle le joueur lâche la boule, cette dernière aura une certaine énergie potentielle. Si le joueur lance une boule droite, l'énergie potentielle n'affectera pas ou peu son jeu. Il est donc dans l'intérêt de ce type de joueur de lâcher la boule le plus près possible de la piste.

Pour les joueurs qui utilisent une trajectoire en crochet en revanche, cela peut avoir une incidence. En effet, plus l'énergie potentielle sera élevée au moment du lâcher, plus la boule aura de rebonds dans sa trajectoire le long de la piste. Ceci se traduira pour la boule par de moindres frictions, empêchant la rotation latérale de prendre effet, et donc amenuisant le crochet. C'est aussi dû au fait que la vitesse horizontale est indépendante de la vitesse verticale, et que donc, si l'on suppose que la boule est lancée avec une force parallèle à l'horizontale, la boule mettra exactement le même temps pour atteindre les quilles, quel que soit le temps passé en l'air, en n'étant pas en contact avec la piste.

La figure la figure 5.4 illustre l'énergie potentielle d'un lancer.

Soit U l'énergie potentielle d'un lancer, on a :

$$U = mgh \quad \text{avec}$$

U = Energie potentielle de gravité

m = Masse de la boule de bowling



FIG. 5.4 – Lanceur en situation réelle durant la course

g = Accélération de la gravité

h = Hauteur de la boule de bowling

5.3.1.3 Direction de la Force Initiale

Pour obtenir un impact maximal, le joueur doit lancer la boule avec une force parfaitement parallèle au sol. Puisque la vitesse horizontale est indépendante de la vitesse verticale, n'importe quelle force qui n'est pas horizontale perd de l'énergie dans la direction verticale. Puisqu'il n'y a aucun bénéfice à faire rebondir la balle, toute l'énergie doit être utilisée dans la direction horizontale. L'énergie cinétique horizontale qui reste au moment où la boule a fini de traverser la piste est l'énergie qui sera utilisée pour percuter les quilles, elle n'est donc pas négligeable.

L'amplitude des forces exercées dans les directions horizontale et verticale peut être calculée en utilisant la trigonométrie. L'amplitude totale de la force multipliée par le cosinus de son angle avec l'horizontale donnera la force horizontale. De la même manière, l'amplitude totale de la force multipliée par le sinus de son angle avec l'horizontale donnera la force verticale (voir figure 5.5). On obtient donc les équations suivantes :

$$F_h = F \cos \theta \text{ et } F_v = F \sin \theta \quad \text{avec :}$$

F_h = Force horizontale

F_v = Force verticale

F = Force d'inertie totale

θ = Angle entre le sol et la force



FIG. 5.5 –

5.3.1.4 Friction, Crochet et Angle d'Impact

Quand un joueur utilise un lancer en crochet, il tire avantage de la friction. La piste étant huilée, elle ne produit qu'un faible coefficient de friction cinétique. De ce fait, il faut beaucoup de temps et de rotation pour que la boule se rabatte vers les quilles. Elle accélère vers la gauche (ou la droite pour un gaucher) à mesure qu'elle prend le peu de friction qu'elle trouve.

La raison pour laquelle les joueurs professionnels utilisent une boule en crochet se rapporte à la direction des collisions avec les quilles. Lancer une boule droite le long de la piste, bien que facile à réaliser, ne distribue pas l'impact de manière efficace. Le mieux étant que la boule vienne percuter la quille n°1 avec un angle de 30° afin de distribuer la force d'impact uniformément selon la disposition des quilles. Si un groupe de quilles est éjecté avec violence alors qu'un autre groupe reste parfaitement stable, c'est que la force d'impact a été mal distribuée.

5.3.1.5 Pin Action

On appelle "Pin action" l'interaction entre les quilles à partir de la première collision avec la boule.

Une fois que la boule a touché des quilles, elle envoie ces quilles heurter d'autres quilles et ainsi de suite. Ça ne s'arrête pas là, puisque ces quilles, en rebondissant les unes contre les autres, sont finalement envoyées dans n'importe quelle direction, y compris celle d'où vient la

boule, jusqu'à ce qu'elles soient toutes tombées ou à court d'énergie cinétique.

Chaque interaction entre deux quilles ou entre une quille et la boule est une collision élastique. Les quilles rebondissent les unes contre les autres parce que leur Moment est conservé durant les collisions.

Pour obtenir le plus de "pin action", le joueur doit transmettre beaucoup d'énergie dans le système, et trouver un angle d'impact permettant de distribuer efficacement l'énergie cinétique.

la figure 5.6 montre un exemple d'angle d'impact favorable aboutissant à un renversement maximal.



FIG. 5.6 – Renversement des quilles

Pour chaque collision Boule-Quille ou Quille-Quille (choc élastique), l'énergie cinétique est calculée selon l'équation :

$$\frac{1}{2}m_1v_{1i}^2 = \frac{1}{2}m_1v_{1f}^2 + \frac{1}{2}m_2v_{2f}^2$$

avec :

m_1 = masse de la boule ou de la quille

v_{1i} = vitesse de la boule ou de la quille avant collision

v_{1f} = vitesse de la boule ou de la quille après collision

m_2 = masse de la quille qui a été touchée

v_{2f} = vitesse de la quille qui a été touchée

Si l'on se recentre maintenant sur l'Utilisateur, on peut modéliser sa tâche de lancer de la manière suivante :

– **Repos :**

C'est une phase durant laquelle l'utilisateur est relâché. Il n'a pas encore commencé son lancer, ou vient d'en finir un. Lorsque l'utilisateur se saisit d'une boule, il passe à la phase suivante.

– **Placement :**

L'utilisateur a la boule en main, il va maintenant devoir se placer sur la largeur de la piste. C'est également durant cette phase qu'il réfléchit au lancer qu'il va effectuer, et se concentre sur le geste à effectuer pour réaliser le lancer imaginé. Lorsqu'il a bien visualisé son geste, il entame sa course d'élan.

– **Action :**

Durant cette phase, l'utilisateur est en action. Il court vers la ligne de faute de la piste et, simultanément, effectue un large mouvement du bras du haut vers le bas, et de l'arrière vers l'avant, afin de donner de la vitesse à la boule. Lorsque la boule est lâchée, l'utilisateur est de nouveau au repos, il attend le résultat de son lancer, et son prochain tour de jeu.

La figure 5.7 représente cette modélisation :

U : Utilisateur

$U = \{U_r, U_p, U_a\}$

U_r : Utilisateur au repos

U_p : Utilisateur en placement

U_a : Utilisateur en action

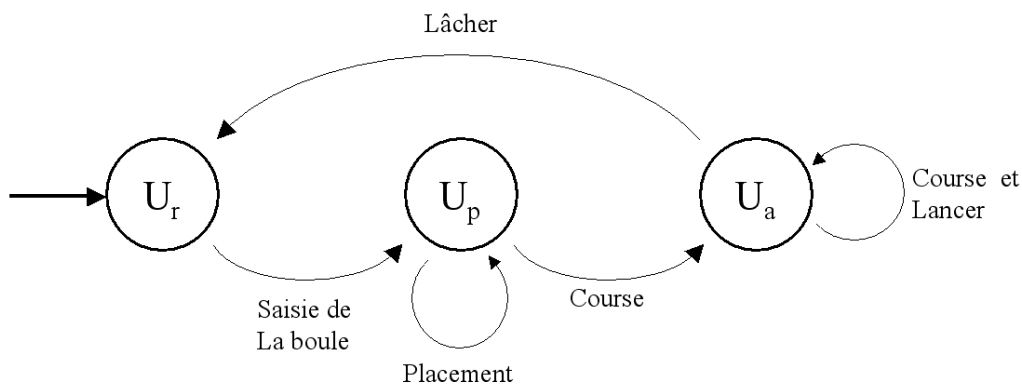


FIG. 5.7 – Automate du joueur de bowling

5.3.1.6 Le Bowling et la Modélisation des Interactions Dynamiques

Par rapport à notre Modélisation des Interactions Dynamiques (Chapitre 2), le bowling réel correspond à la situation $S_m G_c C_f$ (Sujet Mobile, Geste Continu, Cible Fixe) du lancer sur le cube du Lancer.

Toutefois, pour des raisons techniques et de conception du système expliquées ci-après, notre simulation de bowling virtuel se modélise par la situation $S_f G_c C_f$ (Sujet Fixe, Geste Continu, Cible Fixe), le déplacement de l'utilisateur étant nettement restreint.

La figure 5.8 résume cela.

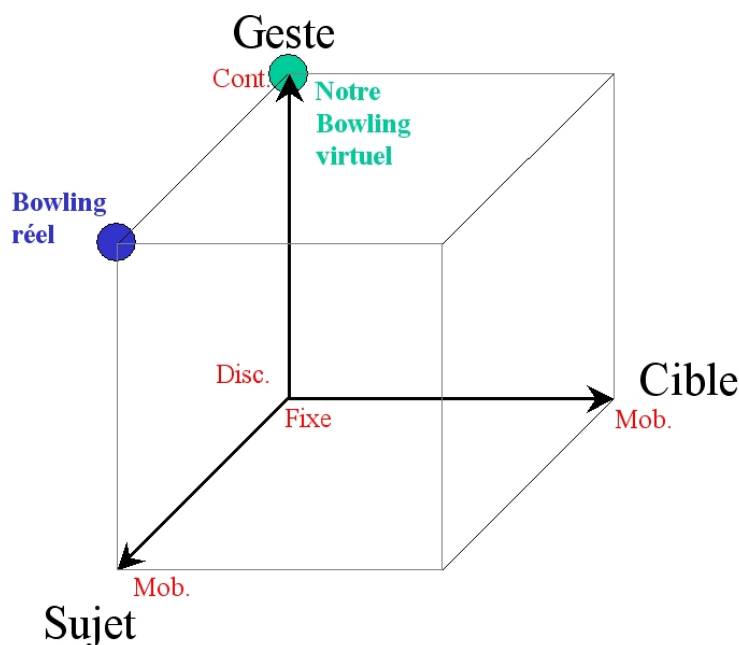


FIG. 5.8 – Bowling réel et virtuel situés sur le Cube de la Modélisation du Lancer

5.3.2 Périphériques et technique d'interaction

Contrairement à la simulation de tir à l'arc, le réalisme dans l'interaction n'était pas le principal objectif. Ce que nous voulions mettre en avant ici était la compréhension des lancers, c'est à dire pouvoir visualiser et assimiler la trajectoire d'un lancer suivant la vitesse et la rotation donnée à la boule, ainsi que le résultat obtenu en terme de "pin action". Cette simulation a donc pour but d'aider le joueur dans sa phase de Placement, en l'aidant à se positionner latéralement sur la piste et en l'aidant à imaginer un lancer idéal ; et non pas dans sa phase d'action proprement dite, ce qui impliquerait une aide sur le mouvement en lui-même.

D'autre part, puisque le geste du lancer en lui-même ne nous intéressait pas, nous avons décidé de nous débarrasser des inconvénients inhérents au bowling, à savoir la nécessité d'une piste, nécessitant un espace conséquent et produisant un volume sonore parfois très élevé. Par ailleurs, si une adaptation au grand public venait un jour à se faire, il n'était pas concevable d'imposer la disponibilité d'un espace libre de grande dimension. Nous avons donc voulu réduire l'espace de travail à un minimum acceptable.

De ce fait, la solution du système de piste raccourcie avec une boule réelle et une caméra pour la filmer afin de transmettre sa vitesse et sa rotation à une boule virtuelle, utilisée par Z. Pan pour son Easybowling [PXH⁺03], n'a pas été sélectionnée.

Il en est de même avec le SPIDAR, qui n'est pas transportable. Nous aurions tout de même voulu l'utiliser pour ajouter un retour haptique à notre application, en simulant le poids de la boule, mais, comme indiqué dans le chapitre 1, l'espace haptique du SPIDAR dont nous disposons ne nous le permettait pas. En effet, cet espace haptique est situé au centre du cube alors que la position de la main lors du lancer de la boule est situé près du sol. Le retour haptique n'est alors pas applicable.

Il était toutefois inconcevable de retenir l'utilisation du clavier et de la souris, ou d'un joystick/joypad comme périphérique d'interaction. Nous jugions cette technique trop éloignée du lancer réel et pas suffisamment immersive pour notre simulation. Nous voulions que l'utilisateur fasse au moins un geste de lancer, même si les conditions de course et les sensations haptiques et kinestésiques n'étaient pas du tout celle du bowling réel.

Nous avons donc décidé que l'utilisateur devrait avoir un objet sphérique en main, et la position de la boule durant le lancer étant assimilable à la position de la main du joueur, il était alors naturel de placer un capteur de position à l'intérieur du-dit objet sphérique. Il fallait également pouvoir détecter le moment où l'utilisateur lâcherait la boule, aussi avons nous inséré un bouton poussoir à la surface de l'objet sphérique faisant office de clic de souris. Si le bouton est enfoncé, l'utilisateur est en action et on enregistre les positions renvoyées par le capteur. Lorsque le bouton est relâché, on procède au calcul de la vitesse initiale et la boule est lancée. Ce système utilise pour le moment une petite boule de polystyrène expansé, qui a été un peu évidée afin d'y insérer le capteur de position et le bouton. Il s'agit juste d'un prototype mis en place pour faire une première expérimentation. Il est malheureusement trop sommaire pour satisfaire tous les besoins que nous avons. Ainsi, l'orientation du capteur de position n'étant ni stable, ni visible, on ne peut utiliser convenablement les informations de rotation qu'il ren-

voie. De ce fait, aucune rotation n'est actuellement appliquée à la boule, même si quelques tests logiciels nous ont permis d'observer que le moteur physique la gèrerait correctement et de manière réaliste. Avec une interface un peu plus soignée, nous pourrions utiliser cette rotation et appréhender le tir en crochet, ainsi que les aides logicielles appropriées. De plus, le champ magnétique du capteur étant limité, l'utilisateur est très limité dans sa course d'élan, l'interaction est par conséquent restreinte au geste de lancer du bras. L'utilisation du SPIDAR pour obtenir la position de la main/boule aurait pu résoudre ce problème s'il n'y avait pas eu le problèmes des fils au niveau des jambes de l'utilisateur, le gênant dans ses déplacements.

Le choix du système de visualisation s'est effectué avec les mêmes critères que pour la simulation de tir à l'arc, nous avons donc opté pour le grand écran plat, si possible rétro-projeté.

5.3.3 Conception et développement logiciels

Ce système a été développé de manière très similaire à l'application de tir à l'arc. Il s'agit donc de code C/C++, faisant appel à la librairie OpenGL pour le traitement graphique, mais c'est cette fois-ci la librairie Fmod qui a été retenue pour gérer l'environnement sonore, du fait de son abord plus aisé qu'OpenAL. La gestion de la fenêtre, de ses messages et événements, est assurée par du code MFC.

Les réelles différences concernant le développement, par rapport à la simulation de tir à l'arc, proviennent de l'emploi de l'API dotXSI pour le chargement des modèles 3D, et du moteur physique NovodeX pour toute la partie physique de la simulation. Les utilisations de ces deux librairies seront décrites ultérieurement.

5.3.3.1 Machine à États

Application : Comme précédemment, nous allons débiter par le fonctionnement logiciel de l'application dans son ensemble :

- **Initialisation :**

Le programme débute par l'incontournable phase d'Initialisation. Comme pour la simulation précédente, on y crée la fenêtre, on configure OpenGL, on charge en mémoire les objets graphiques et leurs textures, on initialise le capteur de positions et on s'occupe des sons et de leur gestion. De plus, on crée les objets "physiques" et on ajuste leurs coefficients de friction et d'amortissement.

– **Ok :**

C'est la phase de "disponibilité", durant laquelle il ne se passe rien de spécial, on attend simplement que l'utilisateur se décide à démarrer un premier ou nouveau lancer. C'est aussi le moment privilégié pour régler certains paramètres, comme le mode de vue, la position de la caméra et l'affichage de l'aide logicielle. Lorsque l'utilisateur est prêt à effectuer un nouveau lancer, il presse le bouton de la boule (et le maintient enfoncé), ce qui initialise une boule virtuelle et marque le début de la phase suivante.

– **Prise :**

A partir du moment où une boule a été initialisée, on considère que l'utilisateur la tient dans la main. Tant qu'elle ne sera pas relâchée, c'est à dire tant que le joueur maintient le bouton appuyé, la boule est assujettie à la main, sa position est donc la même que celle repérée par le capteur de position. On enregistre la position de la boule à chaque frame afin de pouvoir calculer sa vitesse et sa direction initiale au moment du lâcher. La méthode employée pour cela sera décrite plus loin. Lorsque le bouton est relâché, on considère que la boule est lâchée par l'utilisateur, la vitesse et la direction calculées sont transmises à la boule, et on passe à la phase suivante.

– **Roule :**

La boule ayant été lâchée, avec une direction et une vitesse initiales, elle va maintenant rouler le long de la piste, peut-être après un ou deux rebonds. Durant cette phase, on s'occupe principalement du comportement physique de la boule et des collisions qu'elle va engendrer. En fonction des collisions qui seront détectées, des sons appropriés seront joués. Cette phase se termine lorsque la boule a atteint la fin de la piste, ou si elle est manuellement supprimée (via une touche du clavier), ce qui peut être nécessaire si la boule reste bloquée. On passe alors dans la phase FinPiste.

– **FinPiste :**

Lorsque la boule a atteint le bout de la piste (ou qu'on l'a supprimée), on entre dans cette phase. On n'y fait rien de spécial, si ce n'est attendre que les quilles se stabilisent. Pour éviter que cette phase ne dure trop longtemps, on passe automatiquement à la suivante au bout de 4 secondes. Déjà 2 secondes après le début de la phase, le balai de la piste s'affiche pour indiquer que les quilles vont être ramassées. On enregistre l'orientation des quilles avant de passer à la phase suivante.

– **Score :**

Cette phase est très simple, il s'agit juste de comparer l'orientation finale de chaque quille avec son orientation initiale. Si elle diffère de manière notable autrement que par une rotation sur l'axe vertical, cela signifie que la quille a été renversée. Si tel est le cas, on ajoute un point au score réalisée par la boule. D'autre part, si l'on joue avec deux boules

par "frame" (comme au bowling réel), et qu'il s'agissait là de la première boule, on note le numéro de la quille renversée afin de ne pas la remettre pour la boule suivante. Lorsque ce travail a été effectuée pour les 10 quilles, le calcul du score se termine et l'on passe à la phase d'après.

– **Quilles :**

Lorsque le score a été calculé, on procède à la remise en place des quilles. Elles peuvent être toutes recrées, ou seulement en partie, selon le mode de jeu sélectionné. Les quilles sont ensuite "lâchées" d'une hauteur infime à leur position initiale. Ceci est nécessaire, et dû au fait que les quilles sont des objets physiques, comme la piste, et sont soumises à la gravité. Il fallait donc s'assurer que les objets étaient positionnés au bon endroit, et ne s'inter-pénétraient pas. Lorsque les quilles sont "lâchées", on passe à la phase suivante.

– **Stabilisation :**

Le fait de lâcher les quilles, même d'une hauteur très faible, induit de légères oscillations de leur part. Pour le moteur physique, ce mouvement est toujours le même, puisque les quilles sont toujours lâchées de la même position, avec la même vitesse initiale, et sans élément extérieur susceptible d'introduire une modification extérieure du mouvement (courant d'air, particules en suspens, etc.). De ce fait, la position et l'orientation des quilles lorsque la stabilisation s'effectue sont immuables. Les orientations obtenues sont les références par rapport auxquelles on compare les orientations des quilles, enregistrées en fin de phase FinPiste, pour le calcul du score. Lorsque les quilles sont stables, on retourne en phase Ok.

La figure 5.9 récapitule le fonctionnement de la simulation.

La Main : Compte tenu de l'interface utilisée, nous avons repris l'idée de la Main Virtuelle, mais sans nous préoccuper des doigts. Elle sera donc "ouverte" ou "fermée" selon l'état du bouton situé sur la boule tenue par l'utilisateur. Si le bouton est enfoncé, la Main virtuelle est fermée, sinon elle est ouverte. L'action d'enclencher le clic ou de le relâcher fait passer la Main d'un état à un autre.

La figure 5.10 illustre cette modélisation simpliste de la Main Virtuelle.

La Boule : La boule commence par être "Inactive", ce qui correspond à la phase Ok de la simulation. Lorsque l'utilisateur ferme la main, la boule devient Saisie, et le reste tant que la main reste fermée. Lorsque la main virtuelle s'ouvre, la boule est lâchée et Roule sur la piste. Elle reste dans cet état tant qu'elle n'a pas atteint le bout de la piste. Lorsque c'est chose faite, elle redevient inactive.

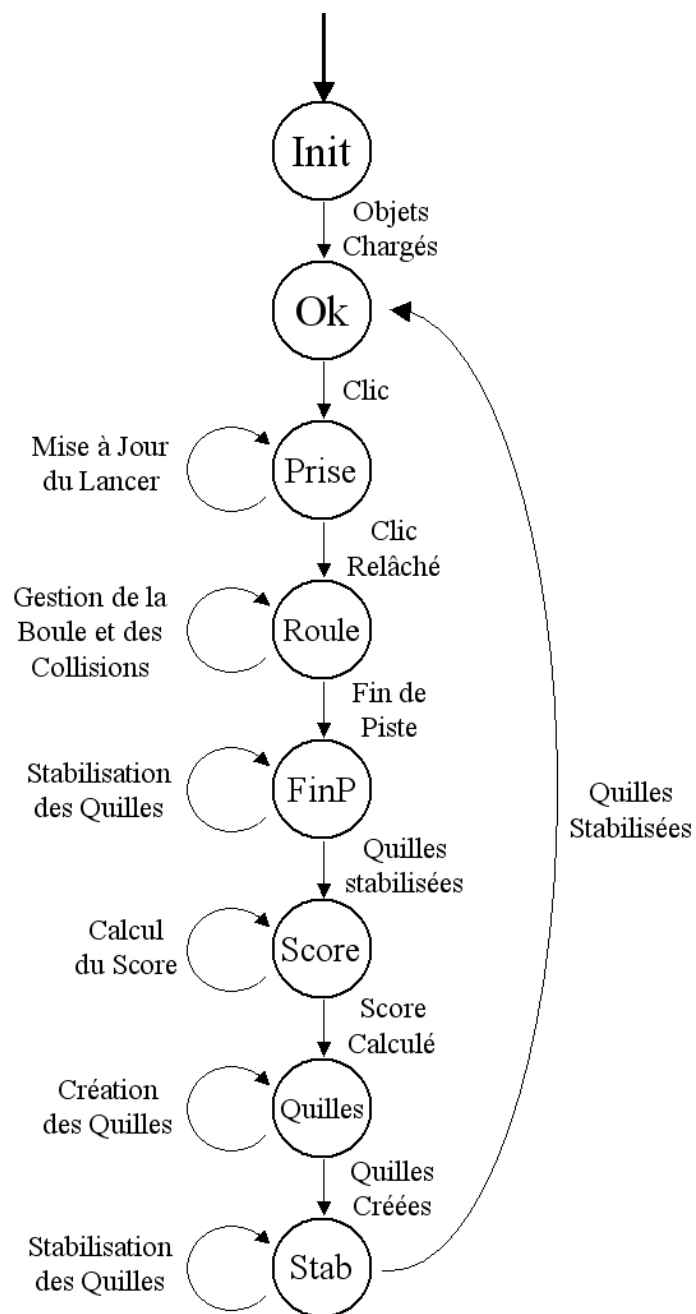


FIG. 5.9 – Diagramme d'États de la Simulation du bowling

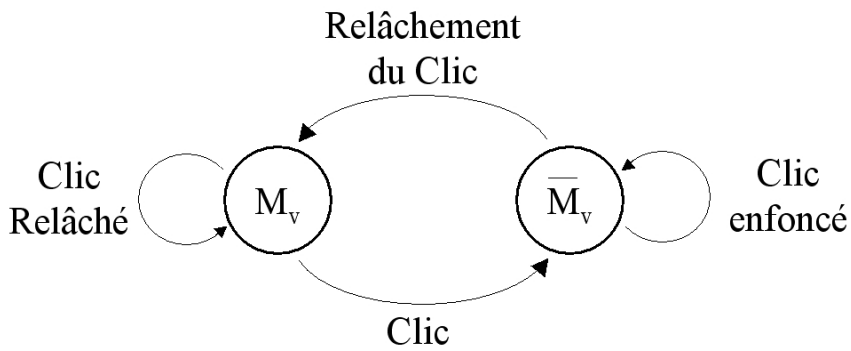


FIG. 5.10 – Diagramme d'États de la Main Virtuelle (Bowling)

La modélisation des états de la boule est illustrée par la figure 5.11.

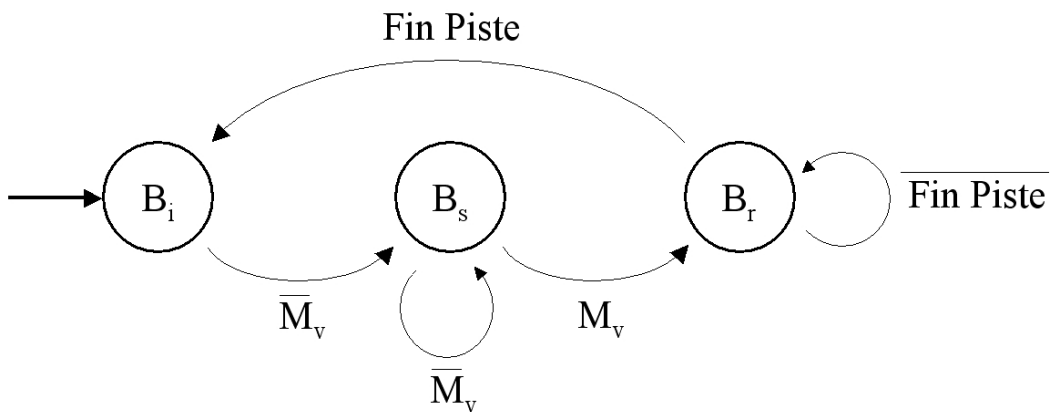


FIG. 5.11 – Diagramme d'États de la Boule virtuelle

La Quille : La Quille étant un élément complexe de la simulation, nous allons détailler son fonctionnement

– **Inactive :**

La Quille commence par être tout simplement inactive. Elle n'apparaît pas graphiquement et n'a aucun comportement physique. Sommairement, elle n'existe pas. Néanmoins, on peut requérir sa Création, lorsque les 10 quilles sont placées en début de frame de jeu, ou si elle n'a pas été couchée par la première boule. Si c'est le cas, on passe à la phase suivante.

– **Création :**

Cette phase consiste juste à créer ou recréer la Quille avec tous les paramètres physiques que cela implique (coefficient de friction, etc.). Cette phase qui ne dure pas longtemps, puisqu'aussitôt la Quille créée, il faut la positionner. On passe donc rapidement à la phase d'après.

– **Positionnement :**

En fonction de son numéro, la Quille doit être placée à une position précise. Étant donné qu'elle est un objet physique, et qu'elle interagit avec les autres objets de ce type, dont principalement la piste, il faut s'assurer que la Quille et la Piste ne s'interpénètrent pas. De ce fait, on positionne la Quille quelques millimètres au-dessus de la piste, et on la lâche. Cela entraîne de légères oscillations de la Quille. Lorsqu'elle est stabilisée, cette phase se termine.

– **Ok :**

A cet instant, la Quille est créée et positionnée, elle est donc dans une phase d'attente. Globalement dans la simulation, cela signifie qu'on attend le lancer de l'utilisateur, ou que la boule a été lancée et roule sur la piste. Si un objet entre en collision avec la Quille, elle entre dans la phase "Mouvement" juste ci-dessous. Si en revanche la simulation a atteint la phase de Calcul du Score sans qu'aucune collision n'ait été détectée concernant la Quille en question, elle est Détruite à cette phase et redevient Inactive.

– **Mouvement :**

Lorsque la Quille est en mouvement, son comportement est pris en charge par le moteur physique de notre simulation. Tant que sa vitesse est supérieure à un ϵ proche de 0, la Quille est toujours considérée en mouvement. Quand sa vitesse passe sous cette valeur seuil, la quille est Stabilisée. Si la Quille est restée debout malgré la collision (c'est le cas si elle n'a été qu'effleurée), elle passe dans l'état Stabilisée Debout. Dans le cas contraire, elle a été renversée, et devient donc Stabilisée Couchée. Cette distinction est nécessaire pour la prise en compte du score et éventuellement pour la recreation des quilles dans la frame de jeu suivante.

– **Stabilisée Debout :**

La Quille a été mise en mouvement, mais pas suffisamment pour la faire se renverser, et elle s'est stabilisée en position verticale. Néanmoins, une nouvelle collision (par une autre Quille elle-même renversée par exemple) remettra la Quille dans l'état "En Mouvement". Si cela ne se produit pas et que la simulation passe en phase de Calcul du Score, la Quille sera Détruite et redeviendra Inactive.

– **Stabilisée Couchée :**

Suite à une ou plusieurs Collisions, la Quille a été renversée et s'est stabilisée en positions

plus ou moins horizontale. Si une nouvelle collision se produit, la Quille repassera dans l'état "En Mouvement". Bien qu'il soit très fortement improbable qu'à partir de là elle retrouve une position verticale, cette possibilité a été laissée offerte à la simulation. Plus logiquement, elle devrait revenir dans cet état de Stabilisation en position couchée, et, une fois la simulation passée dans la phase de "Calcul du Score", la Quille sera détruite et redeviendra "Inactive".

La figure 5.12 illustre cette modélisation des états de la Quille.

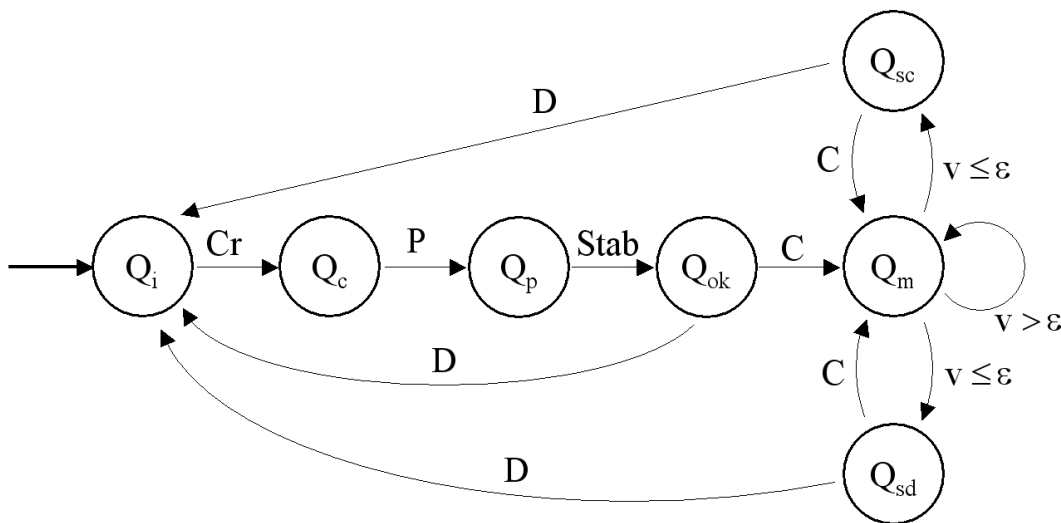


FIG. 5.12 – Diagramme d'États d'une Quille virtuelle

5.3.3.2 Modèles 3D

L'approche utilisée pour cette simulation de bowling diffère de celle employée pour le Tir à l'Arc.

Les modèles 3D des objets utilisés dans la simulation ont été réalisés avec le logiciel Discreet 3D Studio Max 5. Ils ont été ensuite importés et modifiés dans le logiciel Alias/Wavefront Maya 5. Une fois considérés comme définitifs, ces modèles ont été exportés sous le format de fichier Xsi. Grâce à un "loader xsi" (une portion de code permettant de lire et de traiter les objets au format xsi), les objets 3D étaient alors chargés dans la simulation OpenGL. Parallèlement, les objets étaient traités par le moteur physique NovodeX afin de créer les objets physiques correspondant. Nous appelons cette procédure la "Modélisation Physique" des objets.

5.3.3.3 Modélisation Physique

La Modélisation Physique des objets consiste donc à créer l'équivalent physique des objets graphiques. Les objets physiques peuvent alors être gérés indépendamment des objets graphiques, mais il est évident que pour que la simulation soit cohérente, il vaut mieux qu'il y ait une adéquation entre le comportement physique des objets, et leur comportement graphique, c'est à dire ce que l'on voit à l'écran.

Pour que la simulation soit réaliste, il est préférable d'utiliser le maillage des objets 3D pour la détection de collision plutôt qu'une forme approximative. On aurait pu par exemple assimiler les quilles à des cylindres, mais nous voulions une gestion des collisions la plus proche du réel possible, c'est pourquoi les calculs sont effectués en prenant en compte la forme réelle de la quille, et non pas un simple cylindre.

La figure 5.13 illustre cette procédure d'intégration et de gestion des objets 3D.

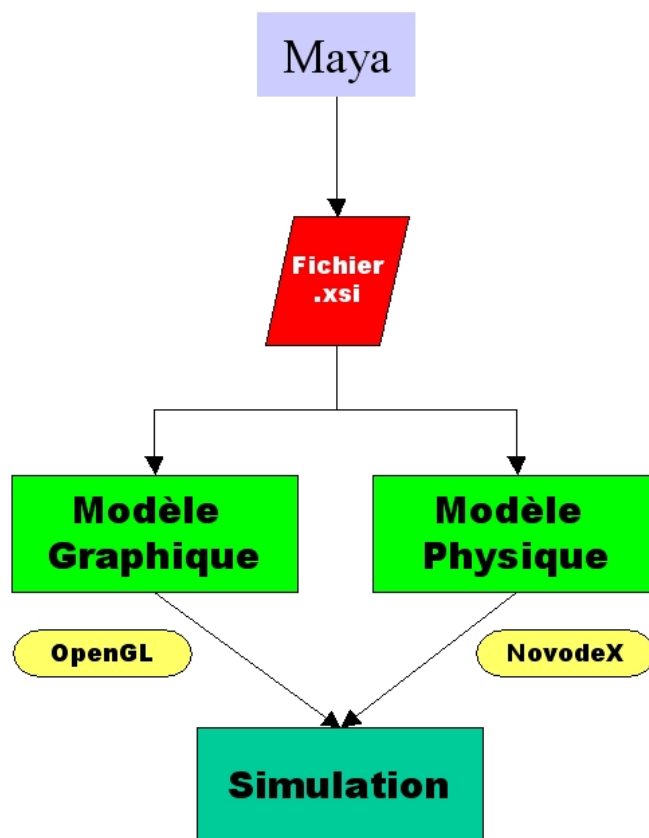


FIG. 5.13 – Schéma de la Gestion des Objets 3D

Le centre de gravité de l'objet est automatiquement déduit par NovodeX au moment de la Modélisation Physique de l'objet graphique. Cela implique donc qu'un objet ayant un point

d'équilibre assez précis, comme une quille, doit être modélisé avec précision. En cas d'erreur trop prononcée, le centre de gravité détecté ne sera pas au bon endroit, entraînant un comportement physique de l'objet plutôt indésirable. Les figures 5.14 et 5.15 illustrent le comportement physique des objets.

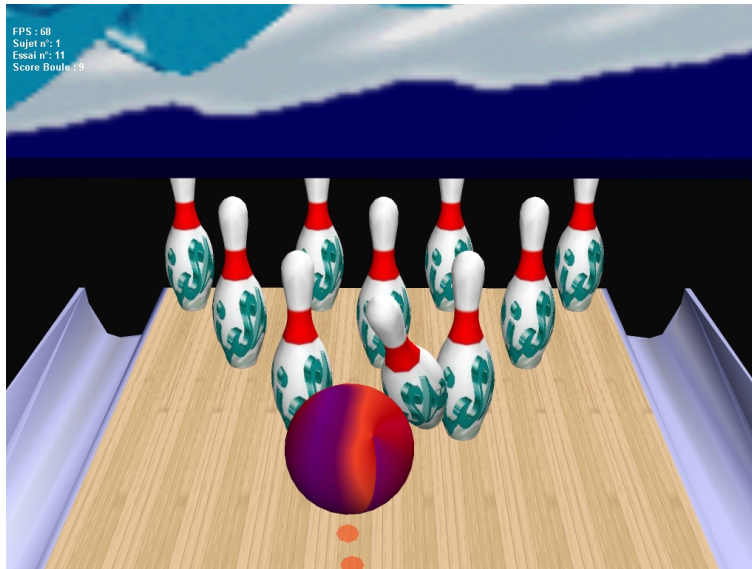


FIG. 5.14 – Capture d'écran d'une simple collision Boule-Quille

5.3.3.4 Gestion du Lancer

La méthode employée consiste à assimiler la position de la boule à celle de la main. On calcule donc la vitesse de la main, que l'on communique à la boule au moment du lâcher. La position instantanée de la main est connue grâce au capteur de position. On enregistre continuellement la position renvoyée par le capteur dans un tableau, ainsi que le Temps correspondant.

Toutes les positions enregistrées n'ont pas à être prises en compte. En fait seules les dernières, celles proches du lâcher vont nous intéresser. En effet, les mouvements intermédiaires de l'utilisateur ne nous concernent pas, il nous faut juste la direction et l'amplitude du mouvement de la main de l'utilisateur au moment de lâcher la boule.

Après analyse d'une vidéo de lancer par un professionnel, nous avons décidé que nous ne prendrions que les positions intervenant moins de 200ms avant le lâcher de la boule. Notre simulation fonctionnant à environ 85 images par seconde, soit autant de positions enregistrées dans ce laps de temps, notre échantillonnage porte sur les 17 dernières positions obtenues. Si on a P_{17} la dernière position obtenue et P_0 la 17ème position précédant P_{17} , on a :

$$\Delta P = P_{17} - P_0$$

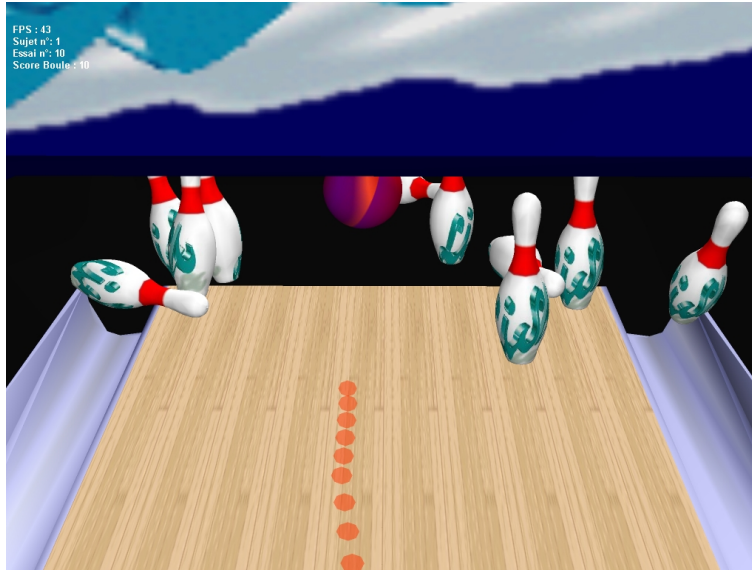


FIG. 5.15 – Capture d'écran d'un "pin action"

Et si on a T_{17} le temps de la dernière position enregistrée et T_0 celui de la 17ème position précédant P_{17} , alors :

$$\Delta t = T_{17} - T_0$$

La vitesse de la boule s'obtient donc de la manière suivante :

$$V = \frac{\Delta P}{\Delta T}$$

Des problèmes peuvent apparaître avec cette méthode, surtout si le temps d'échantillonnage est trop long. Des changements brusques ou prononcés de la trajectoire de la main par exemple ne seront pas détectés, aboutissant à un calcul de la vitesse erroné.

Prenons un exemple sur une trajectoire comportant 4 échantillons, et supposons que la main effectue la trajectoire réelle suivante à vitesse constante (figure 5.16).

les positions enregistrées dans le tableau sont les suivantes (figure 5.16) :

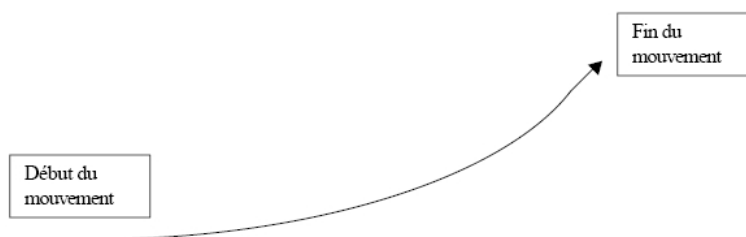


FIG. 5.16 – Trajectoire de la main

par conséquent le Vecteur vitesse calculé est le suivant (figure 5.18)

Ainsi, si la balle est lâchée après P_4 , elle partira dans la direction de ce vecteur (en rouge sur le

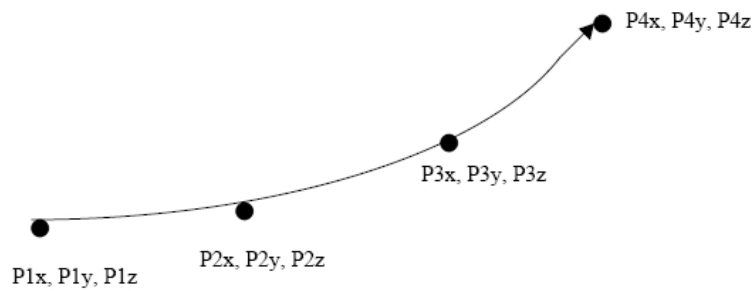


FIG. 5.17 – Échantillons de la trajectoire de la main

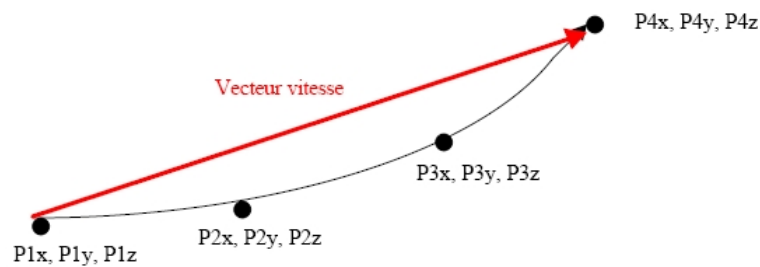


FIG. 5.18 – Vecteur Vitesse calculé à partir des échantillons de la trajectoire de la main

schéma).

En revanche, s'il y a des changements brusques ou des "temps d'arrêt" dans la trajectoire, comme c'est le cas au point "p" dans l'exemple suivant (figure 5.19) alors les positions enregistrées dans le tableau sont les suivantes (figure 5.20) :



FIG. 5.19 – Trajectoire de la main avec un temps d'arrêt "t" au point "p"

par conséquent le vecteur Vitesse calculé est le suivant (figure 5.21) :

Or ce vecteur vitesse, qui sera communiqué à la balle, est erroné. En effet, la main étant immobile au point « P » durant un temps « t », le bon vecteur Vitesse s'apparente à celui de la figure 5.22 :

En revanche, avec un temps d'échantillonnage réduit et avec un lancer normal, c'est à dire si l'utilisateur ne cherche pas à mettre en défaut le système, l'importance de ce problème diminue

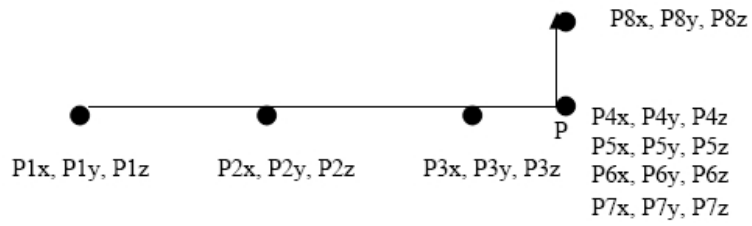


FIG. 5.20 – Échantillons de la Trajectoire comportant un temps d'arrêt

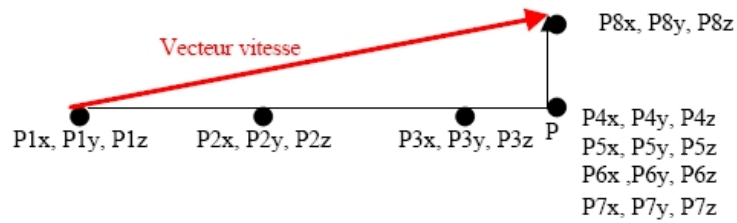


FIG. 5.21 – Vecteur Vitesse calculé à partir des échantillons de la trajectoire comportant un temps d'arrêt

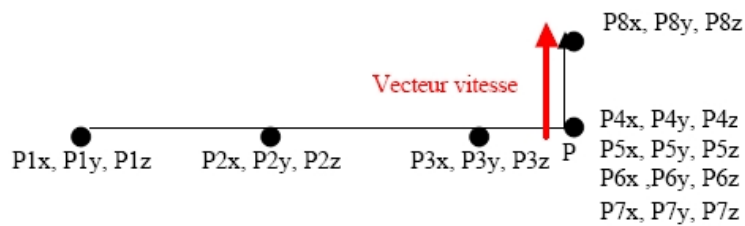


FIG. 5.22 – Vecteur Vitesse adapté à la trajectoire de la main

fortement.

Il est par contre à noter que l'émetteur du Polhemus doit être le plus orthogonal possible au système de projection visuelle, faute de quoi un lancer censé être droit pour l'utilisateur sera détecté oblique par le capteur de position, aboutissant à un décalage gênant.

5.3.3.5 Modes de Jeu

Deux modes de jeu ont été développés. Le mode normal reprend le fonctionnement classique du Bowling américain, à savoir 10 frames par partie, deux boules par frame sauf pour la dernière qui compte trois boules pour déterminer le bonus en cas de strike ou spare. S'il n'y a pas strike lors de la première boule, seules les quilles qui n'ont pas été renversées sont remises en place. Dans tous les cas, toutes les quilles seront remises à leur place après la deuxième boule (sauf cas spécial à la 10ème frame).

Dans le mode Expérimental en revanche, il n'y a qu'une seule boule par frame et toutes les quilles sont remises en place après chaque lancer. Il n'y a pas non plus de nombre limite de frames. Ce mode a été développé spécialement pour nous permettre de conduire notre expérimentation.

5.4 Expérimentation

5.4.1 Objectif de l'expérience

L'objectif de cette expérience était d'étudier l'effet d'une aide logicielle consistant en l'affichage d'une trajectoire à faire suivre à la boule sur la performance humaine lors de la réalisation de la tâche d'une part, et sur l'apprentissage du lancer d'autre part.

5.4.2 Description de la tâche

Le but pour les sujets était d'essayer de réaliser un strike avec chaque boule. Les sujets avaient droits à 5 lancers d'essais afin de se familiariser avec l'interface.

L'expérience était constituée de deux sessions consécutives, la première dite d'"Entraînement" et la seconde de "Test", chacune consistant en 20 lancers. Cette tâche étant moins ardue que celle du tir à l'arc, aucun temps de récupération obligatoire n'a été imposé.

5.4.3 Monde virtuel

Le monde virtuel est composé de 5 pistes de bowling juxtaposées, comme on peut en trouver dans des établissements réels. La piste centrale est celle utilisée par le joueur, les autres ne sont là que pour favoriser un minimum l'immersion. Pour illustrer le fait qu'elles ne servent pas, le balai est en position baissée et aucune quille n'est en place, contrairement à la piste centrale où le balai est relevé et les quilles positionnées.

Nous n'avons toutefois pas représenté un environnement complet. Cela aurait nécessité un certain temps de modélisation pour un résultat n'ayant trait qu'à l'aspect graphique et éventuellement à l'immersion, ce qui n'était pas le but de notre étude.

Le modèle de la piste n'est pas très détaillé. Il y manque les flèches qui servent de repère en conditions réelles, mais qui ne seraient pas très utiles en virtuel à cause du manque de perception de la profondeur. C'est toutefois un défaut qu'il faudra corriger pour les lancers en crochet. De plus, la piste s'arrête à la ligne de faute (représentée par une mince ligne verte), la zone d'approche n'ayant pas été modélisée.

Si l'aide logicielle est activée, comme c'est le cas sur la figure 5.23, des octogones orange semi-transparents représentant chacun une position de la boule durant une trajectoire rectiligne idéale sont affichés sur la piste.

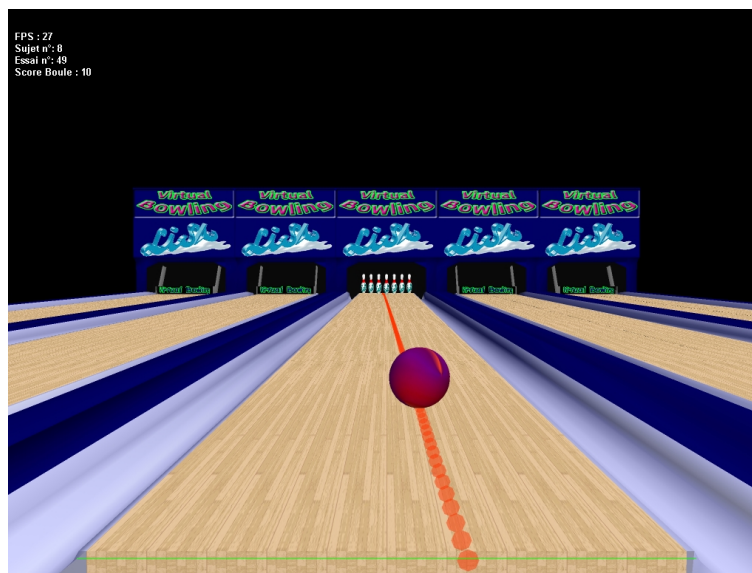


FIG. 5.23 – Monde Virtuel de la simulation de Bowling avec affichage de la Trajectoire Rectiligne

En haut à gauche de l'écran sont affichés le numéro de la boule en cours et le score réalisé

(donc 0 tant que la phase de calcul du score n'a pas été effectuée).

La caméra est fixée un peu en retrait de la piste, de manière à visualiser le tout début de la piste. Il est possible d'activer la caméra dynamique. Dans ce cas là, lorsque la boule est en mouvement, la caméra se déplace pour suivre la boule dans sa trajectoire. Le cas échéant, elle se stoppera néanmoins à environ 1 mètre de la position de la première quille, afin d'avoir une bonne visualisation du "pin action".

5.4.4 ALID : TrajectoireRectiligne

Cette Aide reprend le principe de l'ALID TrajectoireParabolique utilisée pour le lancer de balle décrit dans le chapitre 3. Ici, la trajectoire est symbolisée par des hexagones 2D plutôt que par une ligne en 3D. Cette aide pourra être déclinée pour des trajectoires en crochet, ou pour des dispositions de quilles spécifiques autre que celle de base utilisée ici (split 6-6 ou 7-10 par exemple).

Le tableau 5.1 représente la structure de l'ALID TrajectoireRectiligne.

Propriété	Valeur
Nom	TrajectoireRectiligne
But	"Apprentissage"
PID	"Lancer"
Type	"Visuel"
Description	Cette aide consiste en l'affichage d'un nombre relativement élevé d'hexagones orangés légèrement transparents. Les hexagones sont alignés, et cet alignement illustre la trajectoire idéale que nous proposons pour réaliser un strike avec une boule possédant une trajectoire rectiligne à partir de la position initiale fixée. Les hexagones sont affichés en superposition de la piste mais sont cachés lorsque la boule roule dessus. Ils sont toujours orientés vers la caméra, de manière à être visibles de l'utilisateur.
Genre	"Statique"
Pré-condition	aucune
Post-condition	aucune
État	État_TrajectoireR
Fonctionnement	aucune

TAB. 5.1 – Structure de l'ALID "TrajectoireRectiligne".

Les Paramètres de l'ALID "TrajectoireRectiligne" sont :

- **PID** → **Modèle PID** : $S_f G_c C_f$.
- **Type** → **Sous-Type** : Multiples Objets 2D.
- **État** → **État_TrajectoireR** : Soit P_{lancer} la position de la boule au moment du lâcher, pré-déterminée dans la simulation, alors on a :

$$P_{TrajectoireR} = P_{lancer}$$

L'Orientation des hexagones est telle que leur normale pointe vers la caméra, de manière à être toujours complètement visible de l'utilisateur.

5.4.5 Protocole expérimental

Nous avons fait appel à 10 sujets volontaires novices en réalité virtuelle et n'ayant pas un niveau particulier en Bowling. Ils étaient séparés en 2 groupes de 5 sujets.

Le second groupe disposait de l'affichage de la trajectoire rectiligne lors de la session d'entraînement, contrairement au premier groupe qui ne l'avait pas. Les conditions de la session de test en revanche étaient les mêmes pour les deux groupes, à savoir que l'aide logicielle n'était pas activée. Ce protocole, résumé par le tableau 5.2, est conforme à celui présenté au chapitre 3 et nous permet de nous faire une idée sur l'utilité de l'aide pour la réalisation des lancers, et d'évaluer le transfert d'apprentissage entre les deux sessions.

Groupe	1ère Série	2nde Série
Groupe 1	C1 (aucune aide)	C1 (aucune aide)
Groupe 2	C2 (TrajectoireRectiligne)	C1 (aucune aide)

TAB. 5.2 – Conditions pour chaque série de 20 boules, par groupe.

5.4.6 Analyse des résultats et discussion

Afin d'évaluer la performance des utilisateurs et l'effet de l'Aide Logicielle utilisée dans cette expérience, nous avons enregistré le score réalisé par chaque boule, et par extension le nombre de Strikes effectués par le sujet, ainsi que la trajectoire de la boule au cours du temps, à partir de l'instant où elle est lâchée.

Les données concernant les scores et le nombre de Strikes ont été soumises à la méthode ANOVA.

5.4.6.1 Scores Moyens

L'ANOVA ne révèle aucune différence significative entre les conditions ($F = 0.25$, $p = 0.8583 > 0.05$). Les scores moyens pour les membres de chaque groupe sont résumés dans le tableau 5.3. On obtient ainsi une moyenne par groupe, pour la 1ère Session, de 4.7875 pour le Groupe 1 (Std : 0.40) et de 4.075 pour le groupe 2 (Std : 1.41). Cette moyenne est de 4.3 pour le Groupe 1 (Std : 1.56) et de 4.325 pour le Groupe 2 (Std : 1.03) lors de la 2nde Session.

1ère Session		2nde Session	
Groupe 1 : C1	Groupe 2 : C2	Groupe 1 : C1	Groupe 2 : C1
4.8	4.25	5.2	4.15
4.4	5.25	4.3	3.05
4.6	4.75	2.1	5.55
5.35	2.05	5.6	4.55

TAB. 5.3 – Nombre moyen de Quilles renversées par Sujet, par Groupe et par Session

5.4.6.2 Nombre de Strikes

L'ANOVA globale ne nous permet pas de mettre en relief l'effet de l'Aide Logicielle pour l'Interaction Dynamique utilisée dans cette expérience ($F = 2.0$, $p = 0.1678 > 0.05$), mais l'analyse comparant les résultats du Groupe 2 entre la 1ère et la 2nde Session révèle une différence significative. Le nombre de Strikes réalisés par les membres de chaque groupe sont résumés dans le tableau 5.4. Ainsi, le Groupe 1 a effectué 5 Strikes contre 4 pour le Groupe 2 lors de la 1ère Session, tandis que durant la 2nde Session, le Groupe 1 en a réalisé 7 (soit une amélioration de 40% par rapport à la première Session), contre 10 pour le Groupe 2 (soit une amélioration de 150% par rapport à la Session 1).

1ère Session		2nde Session	
Groupe 1 : C1	Groupe 2 : C2	Groupe 1 : C1	Groupe 2 : C1
2	2	2	4
0	1	2	1
1	1	1	2
2	0	2	3

TAB. 5.4 – Nombre moyen de Strikes réalisés par Sujet, par Groupe et par Session

Nous pensons que ces résultats s'expliquent par le fait que, bien que le geste ne soit pas maîtrisé, les membres du Groupe ayant disposé de l'Aide Logicielle lors de la 1ère Session ont gardé en mémoire, au moins partiellement, la trajectoire proposée. Leur geste s'améliorant

avec les essais, ils ont ainsi pu effectuer des lancers possédant des trajectoires proches de celle proposée, aboutissant à la réalisation de strikes.

5.5 Conclusion

Dans ce chapitre, nous avons expliqué notre intérêt pour le bowling par le fait qu’il s’agissait d’un lancer en 2 dimensions, pour lequel les effets de rotation étaient très importants, et qui fournissait divers exercices intéressants (fermeture de split) en plus de la réalisation de la tâche de base (faire un strike).

Nous avons ensuite détaillé la conception et le développement de notre simulation de bowling, en explicitant notamment les choix que nous avons faits, tant concernant l’interface et que sur le plan logiciel. Enfin, nous avons présenté l’expérimentation que nous avons conduite à l’aide de ce simulateur. Les résultats de cette expérimentation ont ainsi montré l’effet positif de notre ALID pour l’apprentissage de lancers aboutissant à la réalisation de strikes.

Conclusion et perspectives

Conclusion

Dans ce mémoire, nous avons contribué à l'étude de l'interaction dynamique en environnement virtuel. Dans ce contexte, nous avons dans un premier temps situé la place des environnements dynamiques dans le domaine de la Réalité Virtuelle. Nous avons analysé la structure d'un système de RV, puis nous avons effectué une description critique des outils d'entrée-sortie utilisables dans le cadre d'interaction naturelle avec des objets en mouvement ou à mettre en mouvement. Nous avons ensuite présenté les interactions reconnues en réalité virtuelle et démontré qu'elles ne suffisaient pas à recouvrir les possibilités offertes par l'interaction dynamique. Nous avons mis en lumière la nécessité d'approfondir le sujet par la description de travaux de recherche portant sur ce type d'interaction, mais ne l'abordant pas dans son ensemble. Enfin, bien qu'il existât déjà plusieurs types d'aides logicielles, nous avons montré qu'elles n'étaient pas adaptées à l'interaction dynamique et qu'il était donc souhaitable de concevoir une structure d'aides logicielles spécifiques aux environnements dynamiques.

La deuxième partie de notre travail constitue une analyse des tâches d'interaction dynamique. Nous avons établi qu'il existait 3 *Primitives d'Interaction Dynamique* (PID) qui sont le *Lancer*, l'*Attraper* et le *Frapper*. Nous avons modélisé chacune de ces PID selon les composantes dont elles sont constituées parmi : l'*Émetteur*, le *Sujet*, le *Geste* et la *Cible*, et nous en avons déduit le D-Bicube, une modélisation bi-cubique globale des Interactions Dynamiques. Enfin, nous avons contribué à l'amélioration de ce type d'interaction en proposant la structure, les paramètres, les valeurs et les composantes des *Aides Logicielles pour l'Interaction Dynamique* (ALID), qui doivent permettre de favoriser la réalisation et/ou l'apprentissage de tâches impliquant un objet en mouvement ou à mettre en mouvement.

Dans la troisième partie, nous avons débuté une série d'expérimentations dont le but était l'évaluation et la validation des Aides Logicielles pour l'Interaction Dynamique. Dans cette

optique, nous nous sommes intéressés aux tâches basiques de *lancer de balle* et d'*interception d'objets en mouvement*, et nous nous en sommes justifiés. Nous avons conçu et développé une application hautement configurable et aisément paramétrable qui a servi de support à notre étude de ces deux tâches. Après deux expérimentations aux résultats peu encourageant, l'une traitant du lancer de balle sur cible fixe et l'autre portant sur l'interception d'objets en approche frontale, nous avons émis des critiques sur l'environnement virtuel et le protocole expérimental qui avaient été utilisés. Elles nous ont permis d'aboutir notamment à un nouveau protocole, adapté à l'évaluation des ALID, qui a été testé et validé durant une seconde expérimentation sur l'interception d'objets en mouvement, cette fois-ci en approche oblique. Cette dernière étude a mis en valeur l'utilité des *ALID testées* pour la Réalisation de la tâche, mais a également pointé leur inefficacité, voire leur caractère néfaste, pour l'Apprentissage de la tâche.

Puis, dans la quatrième partie, nous nous sommes intéressés au *tir à l'arc*, pour les multiples raisons que nous avons évoquées (dont le fait qu'il s'agisse d'un contrôle bi-manuel avec retour d'effort). Nous avons ensuite décrit le processus de conception et de développement du simulateur qui avait été créé dans ce contexte. Une première expérimentation nous a permis de valider la simulation et a montré qu'il était nuisible à l'utilisateur de trop le perturber dans sa perception du tir. La seconde expérience qui a été menée dans le cadre du tir à l'arc, outre le fait qu'elle ait permis de re-valider le protocole expérimental qui avait été proposé dans la partie précédente, a pour sa part mis au premier plan l'apport des Aides Logicielles pour l'Interaction Dynamique qui avaient été proposées pour cette activité. Les résultats ont surtout démontré qu'il existait bien *deux types* d'ALID : celles destinées à la *réalisation* de la tâche et celles permettant un meilleur *apprentissage* de la tâche.

Enfin, nous avons abordé dans le dernier chapitre le cas du lancer de boule de *bowling*. Parmi les aspects de ce sport qui ont retenu notre attention et que nous avons énumérés, nous avons entre autres cité le fait qu'il s'agissait d'un lancer en 2 dimensions, pour lequel les effets de rotation de la boule étaient primordiaux pour la performance des joueurs. Nous avons expliqué en détail comment l'application que nous avons développée avait été conçue, l'une des difficultés ayant trait à la *Modélisation Physique* de la scène, et notamment celle des quilles, dû à notre volonté d'avoir un système de collisions réaliste. Pour finir, nous avons présenté les résultats de l'expérimentation que nous avons conduite à l'aide de ce simulateur, dans l'objectif de le valider et d'évaluer l'apport de l'affichage d'une trajectoire rectiligne idéale. Ces résultats ont révélé l'effet positif de cette ALID pour l'apprentissage de lancers permettant la réalisation de strikes.

Perspectives

L'étude de l'interaction dynamique en environnement virtuel ne fait que commencer : peu de travaux ont déjà été réalisés, et ceux qu'il reste à faire sont considérables.

Nous avons pour notre part isolé 3 Primitives d'Interaction Dynamique, et apporté pour chacune d'elle une modélisation. Ces modélisations sont à prendre comme des outils d'aide à l'analyse d'une tâche d'interaction dynamique. Elles permettent de situer la tâche en fonction de ces critères Émetteur-Sujet-Geste-Cible et nous espérons qu'à terme il sera possible de sélectionner les Aides Logicielles pour l'Interaction Dynamique ayant été validées pour ces critères. Nous n'avons pu, par manque de temps évident, concevoir, évaluer et valider des ALID correspondant à tous les sommets des modélisations des 3 PID ou de tous les Types possibles (Visuel, Auditif, Haptique). Il serait donc très intéressant de compléter au fur et à mesure un "ensemble" d'ALID testées et fonctionnelles. On pourrait alors, au moment du développement d'une application, "piocher" dans cet ensemble la ou les ALID adaptées à la tâche d'interaction dynamique à laquelle on s'intéresse, ou s'inspirer d'une ALID dont le "Modèle PID" (Émetteur-Sujet-Geste-Cible) s'approche de celui de la tâche que l'on veut étudier.

Nous n'avons pas non plus, à l'heure actuelle, les moyens d'établir une *méthode de conception* des ALID. Celles que nous avons proposées dans ce travail ont été conçues intuitivement, ou adaptées d'aides issues du monde des jeux vidéo. Lorsque davantage d'ALID auront été créées et validées, nous pensons qu'il deviendra possible de mieux comprendre pourquoi elles fonctionnent, et dans quel cas. On pourra alors établir une méthode de conception d'ALID, permettant de facilement créer une nouvelle ALID adaptée à la tâche d'interaction dynamique sur laquelle on souhaite travailler.

L'une des grandes questions concernant nos travaux porte sur l'utilisation de ALID dans le but de faciliter l'apprentissage d'un geste sportif dans le réel. Pour le moment, les applications que nous avons développées ne sont pas assez précises, au niveau du geste à réaliser, pour permettre de s'entraîner en virtuel afin de s'améliorer en réel. Néanmoins, suite à la présentation de nos travaux à ce sujet [IR05], des membres de la section "Tir à l'Arc" de l'Institut National du Sport et de l'Education Physique se sont montrés intéressés par notre simulateur. En effet, un précédent travail de recherche les avait amené à mettre au point un dispositif matériel permettant de calculer avec précisions les paramètres du tir (force de traction, angle de tir, moment de l'échappement, etc.). Sans apport supplémentaire, ce système ne leur est d'aucune utilité, mais couplé à notre simulateur, et avec un calcul de la trajectoire de la flèche plus évolué, il deviendrait possible d'afficher sur la cible réelle le point d'impact prédictif (ALID "PointRouge"). Une telle information serait très intéressante pour les formateurs de ce sport.

En ce qui concerne le bowling, les trajectoires courbes "en crochet" sont gérées par le logiciel, mais nous n'avons pas encore trouvé d'interface ne requérant pas un système de caméras et un lancer réel nous permettant de connaître les effets de rotation imprimés à la boule au moment du lancer. Si un vrai lancer est nécessaire, nous pensons qu'il serait alors plus judicieux de faire de la réalité augmentée. Le système consisterait alors à afficher les ALID en surimpression sur le système d'affichage (généralement des lunettes semi-transparentes). Le joueur disposerait ainsi des avantages que la RV peut apporter, mais au cours d'un entraînement en réel. Un tel système nous apparaît toutefois difficile à mettre en place aujourd'hui.

Compte tenu du fait que la vue est le principal sens utilisé pour l'interaction dynamique, il pourrait être judicieux d'évaluer l'influence du niveau de détail des objets (maillage et texture) sur la perception de la profondeur et sur la performance humaine lors d'interactions dynamiques en général. Si l'on prend l'exemple d'une balle de tennis, elle nous apparaît à grande distance comme une sphère uniformément jaune. Quand elle se rapproche, on distingue des courbes blanches dessinées dessus. A une distance plus faible encore, on voit qu'elle n'est pas exactement sphérique : les courbes blanches forment de légers sillons à la surface de la balle. Enfin, à quelques dizaines de centimètres de nous, on peut distinguer les "poils" jaunes qui donnent sa couleur à la balle. Le niveau de détail d'affichage du modèle pourrait donc donner un indice sur la distance à laquelle il se trouve de l'utilisateur.

Pour finir, le concept des Aides Logicielles pour l'Interaction Dynamique pourrait être étendu afin d'être appliqué aux situations dans lesquelles l'interaction souhaitée est justement *d'éviter* la collision avec l'objet en mouvement. C'est particulièrement le cas dans les tâches de Navigation (conduite automobile, pilotage d'engins aériens ou nautiques) pour lesquelles il est souhaitable d'éviter les collisions avec les piétons ou les autres véhicules.

Bibliographie

- [AMK90] R.A. Abrams, D.E. Meyer, and S. Kornblum. Eye-hand coordination : Oculomotor control in rapid aimed limb movements. *Journal of Experimental Psychology : Human perception and Performance*, 15 :248–267, 1990.
- [BDA99] P. Bourdot, M. Dromigny, and L. Arnal. Virtual navigation fully controlled by head tracking. *Proceedings of International Scientific Workshop on Virtual Reality and Prototyping*, 1999.
- [BDHB99] D. A. Bowman, E. T. Davis, L. F. Hodges, and A. N. Badre. Maintaining spatial orientation during travel in an immersive virtual environment. *Presence : Teleoperators and Virtual Environments*, 6(8) :618–631, 1999.
- [BECH04] L. Bouguila, F. Evequoz, M. Courant, and B. Hirsbrunner. Walking-pad : a step-in-place locomotion interface for virtual environments. In *ICMI '04 : Proceedings of the 6th international conference on Multimodal interfaces*, pages 77–81. ACM Press, 2004.
- [BFM⁺03] B. Bideau, L. Fradet, F. Multon, S. Ménardais, R. Kulpa, and B. Arnaldi. Validation d'un processus de traitement allant de la capture du mouvement à l'immersion de sujets en réalité virtuelle : application au tir au handball. *Revue de CFAO et d'informatique graphique*, 18(2) :151–164, 2003.
- [BH97] D.A. Bowman and L. F. Hodges. An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. *Proceedings of the Symposium on Interactive 3D Graphics*, pages 35–38, 1997.
- [BIS00] L. Bouguila, M. Ishii, and M. Sato. New haptic device for large workspace virtual environment : Scaleable-spidar. *Proceeding of World Multiconference on Systems, Cybernetics and Informatics (SCI 2000)*, pages 415–421, 2000.
- [BKM⁺03] B. Bideau, R. Kulpa, S. Ménardais, L. Fradet, F. Multon, P. Delamarche, and B. Arnaldi. Real handball goalkeeper vs. virtual handball thrower. *Presence : Teleoperators and Virtual Environment*, 12(4) :411–421, August 2003. Fourth international workshop on presence.

- [BMH98] D. C. Brogan, R. A. Metoyer, and J. K. Hodgins. Dynamically simulated characters in virtual environments. *IEEE Comput. Graph. Appl.*, 18(5) :58–69, 1998.
- [BMK⁺04] B. Bideau, F. Multon, R. Kulpa, L. Fradet, and B. Arnaldi. Virtual reality applied to sports : do handball goalkeepers react realistically to simulated synthetic opponents ? In *VRCAI '04 : Proceedings of the 2004 ACM SIGGRAPH international conference on Virtual Reality continuum and its applications in industry*, pages 210–216. ACM Press, 2004.
- [Bux90] W. Buxton. A three-state model of graphical input. *Proceedings of the Human Computer Interaction INTERACT*, pages 449–456, 1990. Elsevier Science Publishers.
- [BW01] D.A. Bowman and C.A. Wingrave. Design and evaluation of menu systems for immersive virtual environments. *Proceedings of IEEE Virtual Reality'2001*, 2001.
- [Car81] L.G. Carlton. Processing visual feedback information for movement control. *Journal of Experimental Psychology : Human Perception and Performance*, 7 :1019–1030, 1981.
- [CBR⁺06] C.M. Craig, E. Berton, G. Rao, L. Fernandez, and R.J. Bootsma. Judging where a ball will go : The case of curved free kicks in football. *Naturwissenschaften*, 93 :97–101, 2006.
- [CNSD93] C. Cruz-Neira, D. Sandin, and T. Defanti. Surround-screen projection-based virtual reality : The design and implementation of the cave. In *SIGGRAPH 93 Conference Proceedings*, volume 27, pages 135–142, 1993.
- [Coi95] Ph. Coiffet. Virtual reality systems : a new technique deeply improving robot behavior. *Proceedings of the 3rd French-Israeli Symposium on Robotics*, pages 7–16, May 1995.
- [CS04] G. Chevrier and R. Sauvain. Virtual pétanque. Rapport de Projet de semestre, Section d’Informatique, École Polytechnique Fédérale de Lausanne, 2004. Laboratoire d’Infographie - Réalité Virtuelle, Hiver 2003-2004.
- [Div87] Polhemus Navigation Science Division. Space isotrack user’s manual, 1987.
- [DS93] R. P. Darken and J. L. Sibert. A toolset for navigation in virtual environments. *Proceedings of the ACM Symposium on User Interface Software and Technology 93*, pages 157–165, 1993.
- [EM87] D. Elliott and J. Madalena. The influence of premovement visual information on manual aiming. *Quarterly Journal of Experimental Psychology*, 39(3) :541–559, 1987.

- [FDFH92] D. Foley, A. Van Dam, S. Feiner, and J. Hughes. *Computers Graphics, Principles and Practice, Seconde édition*. Addison-Wesley Professional, 1992.
- [FMP03] P. Fuchs, G. Moreau, and J.P. Papin. *Le Traité de la Réalité Virtuelle Seconde Edition*. Les Presses de l'École des Mines, 2003.
- [FWC84] D. Foley, V. Wallace, and V. Chan. The human factors of computer graphics interaction techniques. *Proceedings of the IEEE Computer Graphics and Applications*, 4(11) :13–48, November 1984.
- [GBCR02] J. Grosjean, J.-M. Burkhardt, S. Coquillart, and P. Richard. Evaluation of the command and control cube. *icmi*, 00 :473, 2002.
- [Gib77] James J. Gibson. The theory of affordances. In R. E. Shaw and J. Bransford, editors, *Perceiving, Acting, and Knowing*. Lawrence Erlbaum, Hillsdale, 1977.
- [Gro03] J. Grosjean. *Environnements Virtuels : Contrôle d'Application et Exploration de Scènes 3D*. PhD thesis, Université de Versailles-Saint-Quentin-en-Yvelines, 2003.
- [Han97] C. Hand. A survey of 3d interaction techniques. *Computer Graphics Forum*, 16(5) :269–281, 1997.
- [HB96] C. Hendrix and W. Barfield. The sense of presence within auditory virtual environments. *Presence*, 5 :290–301, 1996.
- [Hei60] M. Heilig. Stereoscopic-television apparatus for individual use. US Patent No. 2,955,156, October 1960.
- [HPGK94] K. Hinckley, R. Pausch, J. C. Goble, and N. F. Kassell. Passive real-world interface props for neurosurgical visualization. In *Proceedings of ACM-CHI'94 Conference on Human Factors in Computing Systems*, volume 1, pages 452–458, 1994.
- [HS85] R.M. Hansen and A.A. Skavenski. Accuracy of spatial localisations near the time of of saccadic eye movements. *Vivion Research*, 25 :1077–1082, 1985.
- [IIRS05] F.-X. Inglese, S. Jeong, P. Richard, and M. Sato. A multimodal human-scale virtual environment for product design. In *Proceedings of Virtual Concept 2005, November 08-10, 2005, Biarritz, France*, 2005.
- [IKD⁺05] A. Iyoda, H. Kimura, X. Du, S. Takei, Y. Kakiuchi, S. Fujii, Y. Masuda, D. Masuno, and K. Miyata. Kyukon baseball. International collegiate Virtual Reality Contest 2005, 2005.
- [IKM⁺02] Masataka Imura, Jun Kozuka, Koichi Minami, Yoshito Tabata, Tatsuya Shuzui, and Kunihiro Chihara. Virtual horseback archery : Horse-squared-warrior-squared. *Transactions of the Virtual Reality Society of Japan*, 7(4) :481–486, 2002.

- [ILRF05] F.-X. Inglese, P. Lucidarme, P. Richard, and J.-L. Ferrier. Previs - a human-scale virtual environment with haptic feedback. In *ICINCO 2005, Proceedings of the Second International Conference on Informatics in Control, Automation and Robotics, Barcelona, Spain, September 14-17, 2005, 4 Volumes / CD*, pages 140–145, 2005.
- [INS94] Masahiro Ishii, Masanori Nakata, and Makoto Sato. Networked spider : A networked virtual environment with visual, auditory, and haptic interactions. *Presence*, 3(4) :351–359, 1994.
- [IR05] F.-X. Inglese and P. Richard. Apprentissage de gestes sportifs par les techniques de réalité virtuelle. Premier Colloque Sport et Recherche en Pays de la Loire, March 2005. Nantes, France.
- [IRFB05] F.-X. Inglese, P. Richard, J.-L. Ferrier, and L. Bouguila. Human performance evaluation in a virtual reality archery simulation. In *Proceedings of the IEEE Virtual Reality 2005 Virtuality Structure Workshop*, pages p. 36–40, 2005.
- [IS93] Masahiro Ishii and Makoto Sato. A 3d interface device with force feedback : A virtual work space for pick-and-place tasks. In *VR*, pages 331–335, 1993.
- [IS94] Masahiro Ishii and Makoto Sato. A 3d spatial interface device using tensed strings. *Presence*, 3(1) :81–86, 1994.
- [IYN01] H. Iwata, H. Yano, and F. Nakaizumi. Gait master : A versatile locomotion interface for uneven virtual terrain. *Proceedings of IEEE Virtual Reality Conference 2001 (VR'01)*, 00 :131, 2001.
- [Kru00] E. Kruijff. System control. Siggraph Course "3D User Interface Design", 2000.
- [LaV01] J. LaViola. 2d interfaces in 3d worlds. Siggraph Course "3D User Interface Design", 2001.
- [Log91] Logitech. Logitech 2d/6d mouse technical reference manual, 1991.
- [MAN⁺99] T. Molet, A. Aubel, H. Noser, T. Capin, E. Lee, I. Pandzic, D. Thalmann, N.M. Thalmann, and Sannier. Anyone for tennis ? *Presence*, 8(2) :140–156, April 1999.
- [Min95] M. R. Mine. Virtual environment interactions techniques. *Technical Report 95-018*, 1995. University of North Carolina - Chapel Hill.
- [MNS98] A. J. Madhani, G. Niemeyer, and J. K. Salisbury. The black falcon : a teleoperated surgical instrument for minimally invasive surgery. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'98)*, volume 2, pages 936–944, 1998.

- [Mor02] T. Morineau. Influence des environnements virtuels. *Techniques de l'Ingénieur*, Feb 2002.
- [OCMH00] S. Otmane, E. Colle, M. Mallem, and Ph. Hoppenot. Disabled people assistance by a semiautonomous robotic system. In *Proceedings of the World Multiconference on Systemics, Cybernetics and Informatics*, 2000.
- [OZ00] R. O'Hagan and A. Zelinsky. Visual gesture interfaces for virtual environments. *Proceedings of the First Australasian User Interface Conference*, 00 :73, 2000.
- [PAHR97] J. Paradiso, C. Abler, K.-Y. Hsiao, and M. Reynolds. The magic carpet : physical sensing for immersive environments. In *CHI '97 : CHI '97 extended abstracts on Human factors in computing systems*, pages 277–278. ACM Press, 1997.
- [PEKJ79] C. Prablanc, J.F. Echallier, E. Komilis, and M. Jeannerot. Optimal response of eye and hand motor systems in pointing. *Biological Cybernetics*, 35 :113–124, 1979.
- [PI99] I. Poupyrev and T. Ichikawa. Manipulating objects in virtual worlds : categorization and empirical evaluation of interaction techniques. *Journal of Visual Languages and Computing*, 1(10) :19–35, 1999.
- [PPG86] C. Prablanc, D. Pélisson, and M.A. Goodale. Visual control of reaching movements without vision of the limb. i. role of retinal feedback of target position in guiding the hand. *Experimental Brain Research*, 62 :293–302, 1986.
- [PWBI97] I. Poupyrev, S. Weghorst, M. Billinghurst, and T. Ichikawa. A framework and testbed for studying manipulation techniques for immersive vr. *Proceedings of the ACM Symposium on Virtual Reality Software and Technology 97*, pages 21–28, 1997.
- [PWBI98] I. Poupyrev, S. Weghorst, M. Billinghurst, and T. Ichikawa. Egocentric object manipulation in virtual environments empirical evaluation of interaction techniques. *Proceedings of the EUROGRAPHICS/SIGGRAPH Workshop on Graphics Hardware*, 3(17) :41–52, 1998.
- [PXH⁺03] Zhigeng Pan, Weiwei Xu, Jin Huang, Mingmin Zhang, and Jiaoying Shi. Easy-bowling : a small bowling machine based on virtual simulation. *Computers & Graphics*, 27(2) :231–238, 2003.
- [Ros93] L. B. Rosenberg. The use of virtual fixtures to enhance telemanipulation with time delay. In *Proceedings of ASME winter Annual Meeting on Haptic Interfaces for Virtual environments and Teleoperator Systems*, 1993.
- [RTR06] E. Richard, A. Tijou, and P. Richard. Multi-modal virtual environments for education : From illusion to immersion. In *Technologies for E-Learning and Digital En-*

ertainment, *First International Conference, Edutainment 2006, Hangzhou, China, April 16-19, 2006, Proceedings*, pages 1274–1279, 2006.

- [Sal97] G. Salvendy. *Handbook of human factors and ergonomics*, chapter 22, pages 729–771. John Wiley and Sons, g. salvendy edition, 1997. T.R. Anderson G.R. McMillan, R.G. Eggleston. "Chapter 22, Nonconventional controls".
- [Sch83] Schmandt. Spatial input/display correspondence in a stereoscopic computer graphic work station. *Proceedings of ACM SIGGRAPH'83*, 17(3) :253–261, July 1983.
- [SMWB03] I. Siegler, B. Mantel, W. Warren, and B. Bardy. Le contrôle du rebond en situation de jonglage virtuel : dynamique du comportement avant et après perturbations des paramètres environnementaux. 10ème CONGRES ACAPS - Symposium 8 - Sciences du mouvement et réalité virtuelle, Novembre 2003.
- [SP94] Craig P. Sayers and Richard P. Paul. An operator interface for teleprogramming employing synthetic fixtures. *Presence*, 3(4) :309–320, 1994.
- [Sta98] K. Stanney. Aftereffects and sense of presence in virtual environments : formulation of a research and development agenda. *International Journal of Human-Computer Interaction*, 2(10) :135–187, 1998.
- [Stu92] D. Sturman. *Whole-hand Input*. PhD thesis, MIT, School of Architecture and Planning, 1992.
- [Sut65] I.E. Sutherland. The ultimate display. *Proceedings of IFIP Congres 1965*, 1965.
- [Sut68] I. Sutherland. A head-mounted three-dimensional display. In *Proceedings of the AFIPS Fall Joint Computer Conference*, volume 33, pages 757–764, 1968.
- [TAT⁺06] Yoshihiro Takamura, Norihiro Abe, Kazuaki Tanaka, Hirokazu Taki, and Shoujie He. A virtual billiard game with visual, auditory and haptic sensation. In *Technologies for E-Learning and Digital Entertainment, First International Conference, Edutainment 2006, Hangzhou, China, April 16-19, 2006, Proceedings*, pages 700–705, 2006.
- [Tec93] Ascension Technologies. "a flock of birds". Company brochure, 1993. 2pp.
- [Tec04] Fifth Dimension Technologies. 5dt data glove ultra series user's manual, 2004.
- [Tij06] A. Tijou. Rapport sur l'expérience de persistance. Rapport interne, Laboratoire d'Ingénierie des Systèmes Automatisés, Février 2006.
- [TRR06] A. Tijou, E. Richard, and P. Richard. Using olfactive virtual environments for learning organic molecules. In *Technologies for E-Learning and Digital Entertainment, First International Conference, Edutainment 2006, Hangzhou, China, April 16-19, 2006, Proceedings*, pages 1223–1233, 2006.

- [UAW⁺99] M. Usoh, K. Arthur, M. C. Whitton, R. Bastos, A. Steed, M. Slater, and F. P. Brooks Jr. Walking > walking-in-place > flying, in virtual environments. *j-COMP-GRAPHICS*, 33(Annual Conference Series) :359–364, 1999.
- [War90] C. Ware. Using hand position for virtual object placement. *The Visual Computer*, 6(5) :245–253, 1990.
- [WM93] B. Wirtz and C. Maggioni. Imageglove : A novel way to control virtual environments. *Proceedings of Virtual Reality Systems'93 Conference*, March 1993.
- [Woo99] R. S. Woodworth. The accuracy of voluntary movement. *Psychological review, Monograph Supplement*, 3 :1–119, 1899.
- [ZHK83] H.N. Zelasnik, B. Hawkins, and L. Kisselburgh. Rapid visual feedback processing in single-aiming movement. *Journal of Motor Behavior*, 15 :217–236, 1983.
- [ZLB⁺87] T.G. Zimmerman, J. Lanier, C. Blanchard, S. Bryson, and Y. Harvil. A hand gesture interface device. *Proceeding of SIGCHI/GI conference on Human factors in computing systems and graphics interface 87*, pages 189–192, 1987.
- [ZM03] F.T.J.M. Zaal and C.F. Michaels. The information for catching fly balls : Judging and intercepting virtual balls in a cave. *Journal of Experimental Psychology : Human Perception and Performance*, 29(3) :537–555, June 2003.

Annexe A

La Méthode Statistique ANOVA

A.1 Description de la méthode ANOVA

La méthode d'analyse statistique que nous avons utilisée dans notre étude est la méthode ANOVA (Analysis Of Variance). ANOVA est une méthode pratique de séparation des effets de multiples facteurs dans un ensemble de données.

Cette méthode examine quel facteur a un effet significatif sur une variable dépendante, en comparant la variance à l'intérieur d'un facteur à la variance entre les facteurs.

A.1.1 La variable "F"

La méthode ANOVA permet de calculer deux paramètres statistiques importants pour chacun des facteurs : la valeur **F**, et sa valeur associée **p**. Dans un ensemble de résultats, **F** est une mesure de la variation causée par un facteur expérimental particulier, quand tous les autres facteurs ont été pris en compte. Nous pouvons nous la représenter comme une estimation du poids d'un facteur sur l'évolution d'une variable dépendante. Ainsi, plus **F** est élevée, plus l'effet du facteur associé est significatif. La valeur de **F** est obtenue en faisant le rapport des valeurs de **MS effect** et de **MS error** comme l'indique l'équation A.1 :

$$F = \frac{\text{MS Effect}}{\text{MS Error}} \quad (\text{A.1})$$

A.1.2 La variable "p"

La variable **p** donne la probabilité associée à une valeur **F** correspondante. Autrement dit, **p** représente la probabilité pour que la variation des résultats associés à un facteur donné soit due

au hasard.

Par exemple, une valeur de **p** de 0.1 signifie qu'il y a une chance sur 10 pour que l'effet observé soit due à une variation aléatoire des résultats. Les résultats seront d'autant plus surs que la valeur de **p** est faible. Il est généralement reconnu qu'un effet est significatif si **p** a une valeur inférieure ou égale à 0.05 ($p \leq 5\%$). Ce seuil est accepté comme critère de signification en psychologie de la perception. Ceci est signalé dans les tables STAT GENERAL ANOVA par une astérisque portée sur la probabilité.

A.1.3 Les valeurs MS

"**MS effect**" est la variance calculée pour une variabilité entre les groupes de données, "**MS error**" est la variance calculée pour une variabilité à l'intérieur d'un groupe. La variance étant calculée comme la somme des carrés des écarts à la valeur moyenne, divisée par le nombre de valeurs moins un **Equation A.2**.

$$V = \frac{\sum_{i=1}^n (X_i - \bar{X})^2}{n - 1} \quad (\text{A.2})$$

A.1.4 Les valeurs df

La variable "**df Effect**" est le nombre de degrés de liberté (ddl) entre les groupes de données : il est égal au nombre de valeurs des variables moins un (**Equation A.3**).

$$\text{dfEffect} = n - 1 \quad (\text{A.3})$$

La variable "**df Error**" est le nombre de degrés de liberté (ddl) à l'intérieur d'un groupe de N sujets (**Equation A.4**).

$$\text{dfError} = (n - 1) \times (N - 1) \quad (\text{A.4})$$

La terminologie "effect/error" vient du fait que l'on ne peut vraiment décrire une variabilité à l'intérieur d'un groupe de données, c'est une erreur de variance (Error), alors que l'on peut par la différence des moyennes entre groupes montrer les effets significatifs entre facteurs (Effect).

$$t = \frac{(\text{moyenne du groupe 1} - \text{moyenne du groupe 2})}{\sqrt{\frac{\text{variance du groupe 1}}{\text{taille du groupe 1}} + \frac{\text{variance du groupe 2}}{\text{taille du groupe 2}}}} = \frac{X_1 - X_2}{\sqrt{\frac{S_1^2}{N_1} + \frac{S_2^2}{N_2}}} \quad (\text{A.5})$$

A.2 Les effets croisés

Les facteurs simples, qui ont une importance significative sur les variables, sont appelés *facteurs principaux*. En addition aux effets principaux, ANOVA peut étudier aussi les *interactions d'ordre plus élevé* entre les facteurs. Ces interactions, appelées *effets croisés*, sont des facteurs dont les effets dépendent de l'état d'un ou plusieurs facteurs additionnaux. Une fois que ANOVA a déterminé lesquels des facteurs sont significatifs sur une variable dépendante, une technique inférentielle est utilisée pour déterminer la direction des effets. Pour les tests relatifs à l'interaction visuelle par exemple, la direction de ces effets sera la variation du temps de saisie, la variation de la distance de suivi etc....

A.2.1 Le t-test

Ce test permet de comparer deux-à-deux plusieurs valeurs moyennes. C'est la méthode inférentielle la plus simple. Elle compare les valeurs moyennes pour des tests avec et sans facteur particulier. Le signe de la valeur **t-test** dit si le facteur a un effet significatif (positif) ou non (négatif).

Le **t-test** produit également une valeur de probabilité notée (**p-level**) associée qui indique la différence de fiabilité entre les moyennes. Cette probabilité différera de la valeur **p** associée à **F**, de par le nombre d'observations indépendantes, ou degré de liberté (**df**), qui interviennent dans le calcul des **t-tests** et des **F**.

En général, la statistique **F** est la mesure la plus puissante et la plus fiable d'un effet. Notons qu'il y a deux tests possibles : le **test de Scheffe**, qui est le plus dur au niveau de l'attribution d'un effet significatif, et le **test LSD** qui est plus tolérant (composé de simples t-tests). L'équation A.5 rappelle la formule du t-test.

Annexe B

Application "PID" : Fichiers de Configuration

B.1 Fichiers de Configuration

Comme précisé dans le Chapitre 3, l'application de Lancer/Attraper d'objets est entièrement paramétrable grâce à deux fichiers d'initialisation, ce qui permet de modifier le déroulement de la simulation, en fonction de l'expérience à réaliser, sans avoir à s'introduire dans le code.

Afin de faciliter l'utilisation et la compréhension de ces fichiers, le contenu de ceux-ci est découpé en plusieurs parties distinctes, chacune d'entre elle permettant de paramétrer un élément précis de la simulation.

Détaillons pour commencer le contenu du premier fichier, divisé en 7 parties :

- **Les paramètres du projectile :**

Il est possible de changer la nature et l'apparence du projectile (texture, fil de fer, nuage de points, balle ou cylindre). Si c'est une balle, on peut en modifier le rayon, le poids, la couleur et la vitesse de rotation. Dans le cas du cylindre, on peut également modifier sa hauteur et son diamètre.

- **Les paramètres de l'enceinte parallélépipédique :**

On indique ici la couleur et les dimensions de chaque face de l'enceinte, ainsi que leur rendu (grille, texture ou simple arête).

- **Les paramètres de la cible :**

Ce paramètre permet d'ajuster les dimensions et la position de la cible.

– **Les paramètres de la main :**

Cette donnée détermine la position de départ de la main. On peut éventuellement remplacer la main par une sphère dont on assigne la couleur et le rayon. Enfin, on fixe le gain entre la main réelle et la main virtuelle.

– **Les paramètres des caméras :**

Ce paramètre nous donne accès à la position de la caméra, ainsi qu'à la direction vers laquelle elle pointe. Si une vision stéréoscopique est souhaitée, il y a deux caméras. On peut alors déterminer la distance entre ces deux caméras afin d'accentuer ou de diminuer l'effet de profondeur.

– **Les paramètres du repère :**

Il est possible grâce à ce réglage de positionner n'importe où l'origine du repère dans lequel sont fournies les coordonnées des autres éléments. On peut également en changer l'orientation.

– **Les divers autres paramètres :**

C'est dans cette dernière partie que l'on indique si l'on désire activer les capteurs de localisation, le gant, les ombres, le son, les collisions, etc. On peut y activer un vent latéral, arrière ou frontal afin d'accélérer ou ralentir le projectile selon une direction établie. Enfin, on peut brider le framerate en n'affichant qu'une image sur 2 ou 5 par exemple, et on peut ajuster le temps de latence entre le mouvement réel et le mouvement à l'écran.

Le second fichier permet de paramétrer indépendamment chaque lancement de projectile de la simulation ainsi que le nombre de projectiles qui seront lancés, et est composé de deux parties distinctes. La première est commune aux deux types de tâche implémentés (Lancer et Attraper), la deuxième est propre à chaque tâche. Regardons cela plus en détail :

– **Partie Commune :**

C'est à cet endroit que l'on affecte un numéro à chaque sujet et à chaque expérience. Ces numéros seront utilisés pour la génération des fichiers de résultats retraçant la performance de l'utilisateur durant l'expérience. On y indique également la taille et la latéralité (droitier ou gaucher) du sujet, et le nombre d'itérations de l'expérience (le nombre de projectiles à lancer ou à attraper).

Par ailleurs, c'est ici que l'on va indiquer les aides logicielles et les indices visuels qui seront activés durant l'expérience (présence des grilles, utilisation de la stéréoscopie, affichage de telle ou telle aide logicielle). Enfin, dans le cadre de l'Attraper, on peut spécifier si l'on souhaite que les projectiles soient envoyés aléatoirement ou de manière prédéterminée.

– **Partie Propre - Lancer :**

Cette partie est constituée d'un tableau dans lequel on spécifie, pour chaque tentative de lancer, la position de la cible sur le plan du sol, et la position dans l'espace à laquelle la balle apparaît. On en profite pour indiquer la valeur de la gravité.

– **Partie Propre - Attraper :**

Cette partie se présente sous la même forme que pour le Lancer. Dans le cas où le projectile à attraper est une balle, on spécifie, pour chaque envoi, sa position, sa vitesse et sa direction initiales. Dans le cas du cylindre, on peut de surcroît paramétrer son inclinaison de départ.