INSA ROUEN
À taille humaine,
à l'échelle du monde

EA

État de l'art
PIC FLOYD

version 1.02

Auteur(s)	DELPHINE SOULA, ALINA MIRON, GAUTIER BIDEAULT, ARTURO MONDRAGON CARDENAS, ET YANN BAUCHÉ
Destinataire(s)	Direction du Département ASI, Grande Fabrique.
Résumé	Ce document permet de synthétiser les recherches et études effectuées dans le cadre du Projet INSA Certifié.
Thème et mots clés	État de l'art, recherche, étude, technologie, logiciel, librairie
Nature de la dernière modification	Mise à jour du document
Référence	PICFloyd_S_EA

Exemplaire en diffusion contrôlée ☒
Exemplaire en diffusion non contrôlée ☐

Page de service

Historique des évolutions

Version	Date	Auteur(s)	Modification(s)	Parties modifiées
1.00	23/03/2010	Delphine SOULA, Alina MIRON, Gautier BIDEAULT, Arturo MONDRAGON CARDENAS, et Yann BAUCHÉ	Création	Toutes
1.01	04/06/2010	Delphine SOULA, Alina MIRON, Arturo MONDRAGON CARDENAS, Quentin SUIRE et Yann BAUCHÉ	FOC09	Toutes
1.02	16/09/2010	Gautier BIDEAULT	FOC12	Toutes

Suivi des diffusions

Version	Date	Destinataire(s)
1.00	23/03/2010	Le PIC Floyd et la Grande Fabrique
1.01	04/06/2010	Le PIC Floyd et la Grande Fabrique
1.02	16/09/2010	Le PIC Floyd et la Grande Fabrique

Toute remarque ou difficulté d'application de la présente procédure est à signaler à la Direction Qualité des PIC.

Signatures

	Fonction	Nom	Date	Visa
Vérificateur	Responsable Qualité PIC	Yann BAUCHÉ		
Validateur	Chef Adjoint PIC	Quentin SUIRE		
Approbateur	Client	client		

Table des matières

Page de service 3

I Dossier 7

1 Introduction 9

2 Responsabilités 11

3 Études et recherches effectuées 15
3.1 Interface Homme-Machine . . . 15
3.1.1 Étude de l'Interface Homme-Machine : PSMove . . . 15
3.1.2 Étude de l'interface Homme-Machine : Wiimote . . . 17
3.2 Librairies de manipulation de la Wiimote et du Nunchuk . . . 17
3.2.1 Étude de Librairie Java : Motej . . . 17
3.2.2 Étude de Librairie Java : Wiigee . . . 17
3.2.3 Étude de Librairie Java : WiiRemotJ . . . 18
3.2.4 Étude de Librairie C et Python : Gyroscope WiiMotionPlus . . . 19
3.2.5 Étude de Librairie Matlab : FWiine . . . 20
3.2.6 Étude de Librairie C++ : ArduinoForum . . . 20
3.2.7 Étude de Logiciel de reconnaissance Bluetooth : Bluesoleil . . . 21
3.3 Librairies pincipales de Max/MSP . . . 22
3.3.1 Étude de Librairie Max/MSP : MMJ CNMAT . . . 22
3.3.2 Étude de Librairie Max/MSP : FTM . . . 24
3.4 Communication entre Wiimote et Max/MSP . . . 25
3.4.1 Étude de Logiciel de captation : OSCulator . . . 25
3.4.2 Étude du Patch AKA sous Max/MSP . . . 28
3.5 Projets effectués dans le domaine de la reconnaissance de gestes . . . 28
3.5.1 Étude de l'article : A Gesture follower for performing arts . . . 28
3.5.2 Étude du Projet de l'IRCAM : Wireless sensor interface and gesture-follower for music pedagogy . . . 29
3.5.3 Étude d'un Projet d'étude : Gesture Recognition with Wii Controller . . . 30
3.5.4 Étude d'un Projet d'étude : LiveMove White Paper . . . 30
3.6 Librairie de reconnaissance de gestes . . . 31
3.6.1 Étude de Librairie Java : JaHMM . . . 31
3.6.2 Étude du patch Gesture Follower . . . 32

4 Conclusion 33

II Annexes : Documentations Techniques 35

5 Utilisation de la Nintendo Wiimote 39

5.1 Recherche de solution pour faire fonctionner la Nintendo Wiimote avec Max sous Windows 7 (Familial) . . . 2
5.2 Objets Max existants . . . 2
5.3 Création d'une acquisition bluetooth sour la forme d'un objet Max . . . 2
5.4 Conclusion . . . 3

6 HMM **5**
6.1 Étude de la Librairie Java : JaHMM . . . 5

7 Gesture Follower **9**
7.1 Présentation générale . . . 2
7.2 Explications sur le fonctionnement . . . 2
7.2.1 Global . . . 2
7.2.2 Partie apprentissage . . . 3
7.2.3 Partie reconnaissance . . . 3
7.3 Remarques . . . 3
7.4 Conclusion . . . 5

8 Wiimote **7**
8.1 Wiimote . . . 2
8.1.1 Présentation rapide . . . 2
8.1.2 Connectivité . . . 3
8.1.3 Accéléromètre . . . 3
8.1.4 Caméra infrarouge . . . 3
8.2 Nunchuk . . . 4
8.2.1 Présentation rapide . . . 4
8.2.2 Joystick . . . 6
8.2.3 Accéléromètre . . . 6
8.3 MotionPlus . . . 6
8.3.1 Présentation rapide . . . 6
8.3.2 Gyroscopes . . . 7
8.4 Conclusion . . . 8

Première partie

Dossier

Chapitre 1
Introduction

Cet état de l'art a pour but de faire état des connaissances existantes dans le large domaine de la reconnaissance de gestes et plus spécifiquement de la manipulation de la Wiimote.

En accord avec la Grande Fabrique, nous avons décidé de faire un état de l'art sur les recherches déjà effectuées dans ces domaines et de l'utiliser comme point de départ à nos connaissances pour le PIC. Ainsi une première version de ce document sera livrée à la Grande Fabrique lors de la remise du Dossier de Spécifications Externes (DSE) et du Dossier de Spécifications Internes (DSI) et une version finale lui sera remise à la fin du PIC.

Au cours de cet état de l'art, nous avons cherché à répondre aux problématiques suivantes :

- **Comment récupérer les données de la Wiimote et de ses extensions ? Comment récupérer les données du WiiMotionPlus ?** Malgré le fait que cette dernière extension offre de nouvelles données exploitables en reconnaissance gestuelle, nous avons été étonnés de constater qu'elle n'a pas été étudiée dans ce domaine. Ainsi, nous avons prêté un intérêt particulier à la recherche de moyens permettant de récupérer ses données. De plus, le Nunchuck est aujourd'hui l'outil le plus utilisé par la Grande Fabrique, ainsi la récupération de ces données a été mise en avant.

- **Quels outils sont à notre disposition sous Max/MSP pour faire de la reconnaissance gestuelle ?** La Grande Fabrique nous a donné des pistes que nous avons exploré afin de répondre à cette problématique. Le but étant de se familiariser avec le logiciel mis à notre disposition afin d'en connaître les possibilités.

- **Comment la notion de "Geste" a-t-elle été intégrée dans d'autres projets ?** Nous voulions étudier les définitions qui ont été données au "Geste" et les utilisations qui en ont été faites afin de créer notre propre définition de ce terme en accord avec celle de le Grande Fabrique.

- **Quels sont les outils pré-existants pour la manipulation des modèles de Markov cachés ?** En poussant nos recherches sur les projets existants au sujet de la reconnaissance gestuelle, nous avons vu que les modèles de Markov cachés étaient souvent utilisés actuellement, en particulier pour les reconnaissances gestuelles complexes. Nous avons donc poussé nos investigations sur le sujet afin d'ouvrir la possibilité de l'intégrer à notre projet.

Les recherches sur ces principaux points nous ont amené à construire un premier état de l'art.

Cet état de l'art va être amené à évoluer au court du projet car, d'une part, d'autres problématiques peuvent apparaître lors de la réalisation des livrables pour la Grande Fabrique. D'autre part, les recherches autour de l'utilisation de la Wiimote dans l'art ou autour de la reconnaissance gestuelle sont des un problèmatiques qui font l'objet de nombreuses recherches aujourd'hui et de nouveaux outils et algorithmes apparaissent régulièrement.

Nous avons cherché à décrire chacun des sujets de recherche de manière à expliquer leur fonctionnement tout en analysant les intérêts et inconvénients pour le PIC.

Pour le premier semestre, Delphine SOULA a été en chargée de veiller à la cohérence de ce document et des informations qu'il contient. Gautier BIDEAULT, Alina MIRON, Arturo MONDRAGON CARDENAS et Yann BAUCHÉ ont été chargés des recherches et de la rédaction des chapitres techniques.

Chapitre 2
Responsabilités

Le tableau ci-dessous décrit la répartition des recherches entre les membres du PIC ayant participés à la rédaction de l'état de l'art :

Objet de la recherche	Technologie étudiée	Date			Auteur	
		Proposition	Recherche	Edition fiche	Recherche	Validation
MoteJ	Librairie Java	22/02/10	24/02/10	5/03/10	MIRON	BIDEAULT
Wiigee	Librairie Java	22/02/10	24/02/10	2/03/10	BIDEAULT	SOULA
WiiRemotJ	Librairie Java	22/02/10	24/02/10	5/03/10	MIRON, BIDEAULT	SOULA
Gyroscope WiiMotionPlus	Librairie C, Librairie Python	22/02/10	24/02/10	3/03/10	BIDEAULT	SOULA
FWiine	Librairie Matlab	22/02/10	24/02/10	3/03/10	BIDEAULT	SOULA
Arduino forum	Librairie C++	22/02/10	24/02/10	2/03/10	BIDEAULT	SOULA
CNMAT	Librairie Max/MSP	22/02/10	22/02/10	9/03/10	MONDRAGON CARDENAS	SOULA
FTM	Librairie Max/MSP	22/02/10	22/02/10	10/03/10	MONDRAGON CARDENAS	SOULA
JaHMM	Librairie Java	11/03/10	15/03/10	19/03/10	MIRON	SOULA
A Gesture follower for performing arts	Projet d'étude	8/03/10	8/03/10	16/03/10	BAUCHÉ	SOULA
Wireless sensor interface and gesture-follower for music pedagogy	Projet de l'IR-CAM	8/03/10	8/03/10	16/03/10	BAUCHÉ	SOULA
Gesture Recognition with Wii Controller	Projet d'étude	8/03/10	8/03/10	16/03/10	BAUCHÉ	SOULA
LiveMove White Paper	Projet d'étude	8/03/10	8/03/10	16/03/10	BAUCHÉ	SOULA

Objet de la recherche	Technologie étudiée	Date Proposition	Date Recherche	Edition fiche	Auteur Recherche	Auteur Validation
Osculator	Protocole de communication	25/5/10	26/5/10	28/5/10	MONDRAGON CARDENAS	SOULA
Bluesoleil		25/5/10	26/5/10	2/6/10	MIRON	SOULA
PSMoove	Interface Homme-Machine	25/5/10	26/5/10	29/5/10	SUIRE	SOULA

L'état de l'art est complété par la documentation technique qui a été effectuée de façon plus poussée sur certaines technologies.

Objet de la recherche	Technologie étudiée	Date de création	Auteur		Numéro de DT
			Rédaction	Validation	
AKA	Protocole de communication	8/3/10	MEULEMAN		01
HMM	Algorithme de reconnaissance gestuelle	12/3/10	MIRON		03
WiiRemotJ	Interface Homme-Machine	18/3/10	SUIRE		04
GestureFollower	Librairie Max/MSP	28/5/10	BAUCHÉ		06

Le résultat de ces recherches a été résumé dans la documentation technique du PIC et a été porté en Annexe (Chapitre II) à ce document.

Chapitre 3
Études et recherches effectuées

3.1 Interface Homme-Machine

3.1.1 Étude de l'Interface Homme-Machine : PSMove

Présentation de PSMove

Le PSMove est la nouvelle manette de la PlayStation 3 de Sony. Elle sera disponible sur le marché mondial en automne 2010, et coûtera à peu près 100$. Elle se veut la concurrente directe de la Wiimote de Nintendo. Avec une annonce d'une précision de capture de mouvement 1 :1 aussi bien à vitesse lente que rapide.

Pour le moment, elle n'a été qu'annoncée par Sony, et peu de caractéristiques techniques ont été dévoilés. On sait cependant, qu'il s'agit d'une double manette :

- le **motion controller** (voir figure 3.1). Cette manette embarque :
  - un accéléromètre 3 axes (pour les déplacements),
  - un gyroscope 3 axes (pour les angles de la manette),
  - un magnétomètre (pour corriger les erreurs accumulées du gyroscope),
  - une boule colorée à son extrémité supérieure pour le tracking.

  En effet, cette boule permet la localisation de la manette dans la pièce à l'aide de l'EyeToy (la caméra de la PlayStation 3) obligatoire pour utiliser la manette. De plus, cette manette possède les boutons croix, rond, carré et triangle habituels chez Sony.
- le **navigation controller** (voir figure 3.2). Cette deuxième manette possède un joystick analogique ainsi que des flèches directionnelles et encore les boutons croix et rond (nécessaire à la navigation dans les menus de la PlayStation 3). Elle ne possède aucun matériel permettant sa localisation.

FIG. 3.1 – Motion controller

Avantages de PSMove

Les avantages de PSMove sont :

- elle possède à la fois une **boule colorée** pour le tracking par caméra, mais aussi un **accéléromètre 3 axes**, un **gyrosocope** également **3 axes**, et un **magnétomètre** pour recalibrer le gyroscope.

FIG. 3.2 – Navigation controller

- elle utilise le **bluetooth** 2.0.
- elle fonctionne avec une **batterie li-ion** rechargeable par mini-USB.
- elle possède les boutons habituels d'une manette de PlayStation, et un **joystick analogique**.

Inconvénients de PSMove

Les inconvénients de PSMove sont :
- elle est plutôt chère (**environ 100$**).
- elle ne sort qu'en **automne 2010**.
- Sony ne dévoilera pas son fonctionnement, il faudra donc que de la **rétro-ingénierie** soit faite, ce qui rajoute encore un délai avant une utilisation sur ordinateur.

Liens utiles pour PSMove

Pour plus de détails :
- le site officiel du PSMove : `http://us.playstation.com/ps3/playstation-move`
- l'article Wikipédia américain : `http://en.wikipedia.org/wiki/PlayStation_Move`

3.1.2 Étude de l'interface Homme-Machine : Wiimote

Voir la documentation technique en Annexe, Chapitre 8.

3.2 Librairies de manipulation de la Wiimote et du Nunchuk

3.2.1 Étude de Librairie Java : Motej

Présentation de Motej

Motej est une librairie Java qui permet de récupérer les données d'une Wiimote. Elle se compose de deux packages principaux : *Motej library* and *Motej extras*. La librairie Motej fournit un accès de base à la Wiimote et nécessite une JSR 82 (Bluetooth) mise en œuvre et slf4j.

Avantages de Motej

Les avantages de Motej sont :
- Offrir une solution Open Source,
- Communiquer avec la Wiimote et ses extensions : balance board, contrôleur classique, nunchuck ...,
- Permettre d'ajouter de nouveaux modules.

Inconvénients de Motej

Les inconvénients de Motej sont :
- Implémenter un nouveau module pour WiiMotionPlus.

Liens utiles pour Motej

`http://sourceforge.net/projects/motej/files/`

3.2.2 Étude de Librairie Java : Wiigee

Présentation de Wiigee

La librairie Wiigee est une librairie de reconnaissance gestuelle basée sur les accéléromètres de la Wiimote de Nintendo. Elle est implémentée en Java et s'appuie sur la technologie bluetooth de la Wiimote. La reconnaissance de gestes se fait en quatre temps :

1. Un filtre qui supprime toutes les données supérieures à un seuil fixé,
2. Un clustering de type K-Means, de façon à regrouper les données reçues en fonction de leurs caractéristiques,

3. L'application d'un HMM (left-to-right) sur les clusters générés précédemment afin de dégager des modèles de ces derniers,
4. Finalement, l'application de l'algorithme de Bayes, de façon à classifier les modèles obtenus au-dessus.

Applications testées

Un programme gérant le passage de diapositives est fourni avec la librairie. Il fonctionne relativement bien malgré des problèmes au niveau de la reconnaissance de gestes car il ne tolère pas d'erreurs au niveau de la réalisation du geste. Cependant, nous n'avons pas eu d'aperçu du mode apprentissage, les gestes permettant de passer d'une diapositive à une autre étant déjà prédéfinis.

Avantages de Wiigee

Les avantages de Wiigee sont :
– Mode d'apprentissage et mode de reconnaissance,
– Récupération des données issues des accéléromètres et des boutons de la Wiimote.

Inconvénients de Wiigee

Les inconvénients de Wiigee sont :
– Pas de prise en compte de la WiiMotionPlus ou du Nunchuck,
– La détection des gestes est très peu tolérante aux erreurs.

Liens utiles pour Wiigee

`http://www.wiigee.org/`

3.2.3 Étude de Librairie Java : WiiRemotJ

Présentation de WiiRemotJ

Librairie Java permettant de récupérer diverses informations de la Wiimote de Nintendo.

Tests effectués

Lors d'un Élément Constitutif d'Approfondissement ou d'Ouverture proposé au semestre précédent, nous avons mis en place la création d'aléas sur les données récupérées. Nous avons créé une interface, utilisant un pattern Modèle Vue Contrôleur, qui applique des fonctions aux valeurs des accéléromètres de la Wiimote. Une interface Homme-machine a été réalisée ainsi qu'un module d'affichage graphique des valeurs des accéléromètres.

Nous avons également réalisé un test pour WiiMotionPlus, en créant un module qui permettra d'ajouter cette fonctionnalité à la bibliothèque. Cependant les méthodes de gestion de la WiiMotionPlus n'étant pas encore intégrés à WiiRemotJ ces tests n'ont pas pu aboutir.

Avantages de WiiRemotJ

Les avantages de WiiRemotJ sont :
- Récupération des valeurs des accéléromètres de la Wiimote et des boutons,
- Filtre débruiteur réalisé,
- Possibilité d'appliquer des fonctions aux valeurs reçues,
- Interface graphique,
- Pattern Modèle Vue Contrôleur permettant l'application rapide de nouvelles fonctions aux données captées.

Inconvénients de WiiRemotJ

Les inconvénients de WiiRemotJ sont :
- Programme pas assez segmenté,
- Pas de prise en charge du module WiiMotionPlus,
- Sources non disponibles rendant difficile l'ajout de nouveaux modules.

Liens utiles pour WiiRemotJ

`http://www.world-of-cha0s.hostrocket.com/WiiRemoteJ/`
`http://wiibrew.org/wiki/Wiimote/Extension_Controllers#Wii_Motion_Plus`

3.2.4 Étude de Librairie C et Python : Gyroscope WiiMotionPlus

Présentation de Gyroscope WiiMotionPlus

Ce programme permet de récupérer les données mesurées par les gyroscopes du module WiiMotionPlus. Il est implémenté en C et en Python.

Avantages de Gyroscope WiiMotionPlus

Les avantages de Gyroscope WiiMotionPlus sont :
- Récupération efficace de l'intégralité des données du module WiiMotionPlus.

Inconvénients de Gyroscope WiiMotionPlus

Les inconvénients de Gyroscope WiiMotionPlus sont :
- Prise en charge exclusive du module WiiMotionPlus (pas d'interface Wiimote), nécessité d'une interface matérielle d'acquisition supplémentaire.

Liens utiles pour Gyroscope WiiMotionPlus

`http://www.pobot.org/Gyroscope-Wii-Motion-Plus.html`

3.2.5 Étude de Librairie Matlab : FWiine

Présentation de FWiine

Cette librairie est une librairie Matlab pour Windows, et Scilab pour Linux qui permet de récupérer l'intégralité des données du module WiiMotionPlus.

Avantages de FWiine

Les avantages de FWiine sont :
– Récupération de l'intégralité des données du module WiiMotionPlus.

Inconvénients de FWiine

Les inconvénients de FWiine sont :
– Utilisation de Scilab sous Linux.

Liens utiles pour FWiine

`http://fwiineur.blogspot.com/2008/05/fwiine-release-02-is-available-english.html`

3.2.6 Étude de Librairie C++ : ArduinoForum

Présentation de ArduinoForum

Il s'agit d'un programme permettant de récupérer les données du module WiiMotionPlus. Le langage utilisé est le C++.

Avantages de ArduinoForum

Les avantages de ArduinoForum sont :
– Récupération des données du module WiiMotionPlus.

Inconvénients de ArduinoForum

Les inconvénients de ArduinoForum sont :
– Implémentation en Java nécessaire pour le portage sous Max/MSP,
– Utilisation d'une interface matérielle Arduino et non prise en charge de la WiiMote.

Liens utiles pour ArduinoForum

`http://randomhacksofboredom.blogspot.com/2009/06/wii-motion-plus-arduino-love.html`
`http://www.arduino.cc/cgi-bin/yabb2/YaBB.pl?num=1245723011`

3.2.7 Étude de Logiciel de reconnaissance Bluetooth : Bluesoleil

Présentation de Bluesoleil

Ce Logiciel de reconnaissance Bluetooth permet de connecter la manette Wii à un PC grâce à une clef bluetooth (sauf si votre ordinateur a le bluetooth intégré). Bluesoleil est utilisé pour récupérer et traiter les données de la Wiimote grâce au logiciel pour contrôler la wiimote : glovepie.

Avantages de Bluesoleil

Les avantages de Bluesoleil sont :
- Il est l'un des logiciels PC Bluetooth les plus populaires dans l'industrie,
- Il permet de connecter à votre PC une large variété de périphériques Bluetooth numériques, tels que les téléphones mobiles, stéréo / mono casques, claviers et souris, appareils photo, imprimantes, appareils GPS, PDA et plus encore.
- Il est compatible avec Windows et Linux (Distribution Ubuntu et Debian).

Inconvénients de Bluesoleil

Les inconvénients de Bluesoleil sont :
- Il n'est pas compatible avec MAC OS.

Conclusion sur la possibilité de remplacer OSculator par Bluesoleil

Bluesoleil n'est pas un logiciel pour remplacer OSculator sur le système d'exploitation Windows, mais il pourrait être utilisé comme un logiciel fiable pour la connexion avec la télécommande Wii. Il faudrait le coupler à une autre interface qui fait la liaison entre la télécommande Wii et Max MSP environnement (comme *tk Wii*), cela pourrait être une alternative.

Les avantages par rapport à OSculator de l'utilisation de Bluesoleil et tk Wii sont :
- tk Wii est disponible pour Windows,
- il est open source,
- il peut se connecter à plusieurs télécommandes Wii.

Les inconvenients par rapport à OSculator de l'utilisation de Bluesoleil et tk Wii sont :
- la dernière version de tk Wii a été publiée en 2007, il ne prend donc pas en charge les nouvelles fonctionnalités de la télécommande Wi.

Liens utiles pour Bluesoleil

`http://ether.jp/blog/?page_id=5` `http://lowfrequency.org/interactivity/wiki/index.php?title=Using_Nintendo_Wiiremotes_in_Max/MSP/Jitter` `http://www.logic-sunrise.com/dossiers-et-tutoriaux-77237-utiliser-sa-wiimote-sur-son-pc-via-bluetooth.html`

3.3 Librairies pincipales de Max/MSP

3.3.1 Étude de Librairie Max/MSP : MMJ CNMAT

Résumé

Max/MSP est un logiciel de programmation visuelle permettant de réaliser, traiter et diffuser du son et de la vidéo en temps réel. La CNMAT, *The Center for New Music and Audio Technologies*, est un centre de recherche de musique contemporaine qui réalise des bibliothèques et des applications Max/MSP. Leurs recherches ont abouti à l'élaboration d'une librairie, le dépôt MMJ (Max/MSP/Jitter).

Présentation de MMJ CNMAT

Le CNMAT a montré un intérêt croissant pour la pédagogie assistée par ordinateur dans le domaine de la musique. Les recherches dans ce domaine soulèvent un problème lorsque différents organismes développent en parallèle des modules similaires.

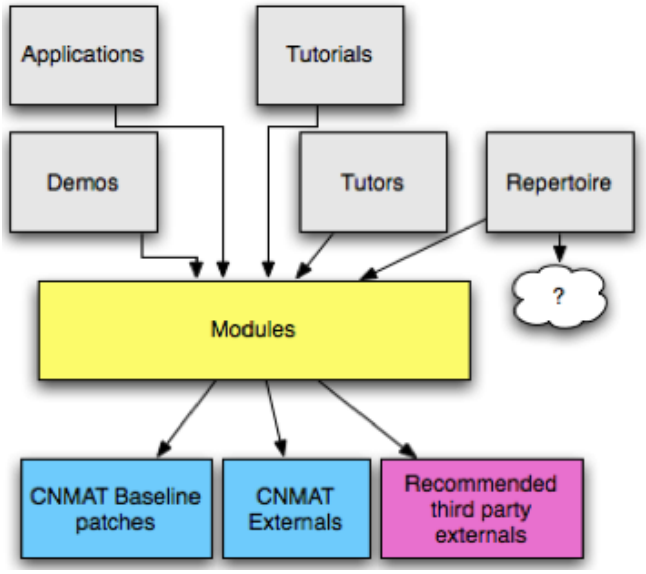
Les avancements faits dans ce domaine sont importants mais ils sont souvent dévalorisés par cette décentralisation des connaissances. La mise en commun de ces connaissances représente pour les musiciens un gain de temps car de nos jours ils redéveloppent leurs outils au lieu de travailler l'esthétique et la technique de leurs instruments. Dans le but de revaloriser les travaux effectués dans ce domaine, la CNMAT a mis à disposition sa libraire MMJ.

Structure MMJ CNMAT

MMJ contient 3 types des fichiers principaux :

1. Fichiers au format Max/MSP (`.pat` ou `.maxpat`) appelés "Patch".
2. Fichiers d'aide au format Max/MSP (`.maxhelp`)
3. Fichiers compilés pour Max/MSP (`.mxo`)

Ces fichiers sont rangés dans des packages avec la structure suivante :



Les flèches montrent les dépendances.

Exemple d'application

Prenons l'exemple de *Tutorials*. Il a besoin de modules tels que : *Data_Structure, Communication, Controllers*. Notons que le module *Controllers* met à disposition un "Patch" qui permet la connexion avec la tablette graphique Wacom. Nous avons besoin des "Patches" du package *CNMAT Baseline Patches* qui implémentent le protocole OSC afin de finaliser la communication avec cette tablette graphique.

Cet exemple montre l'interdépendance qu'il y a entre les modules, interdépendance également décrite sur le schéma précédent.

Des informations plus détaillées concernant chacun de ces modules et leur répartition sont disponibles sur le lien suivant : `http://cnmat.berkeley.edu/library/max_msp_jitter_depot`.

Avantages de MMJ CNMAT

Pour la première prise en main de cette librairie, il existe le "Patch" `+CNMAT_MMJD_Overview`, qui permet d'avoir un aperçu global de la librairie avec une description de ces principales fonctionnalités.

Cette librairie présente les avantages suivants :

- **Contrôle** : Il existe de nombreux "Patches" qui permettent le contrôle et l'adressage des données, comme par exemple le protocole de communication responsable de la communication entre les différents périphériques.
- **Audio** : Plusieurs "Patches", en ce qui concerne l'audio, ont la possibilité d'augmenter leurs fonctionnalités au moment de la construction d'un moteur audio. Par exemple, l'objet `multibuf` qui nous permet de *bufferizer* des sons de manière dynamique.

Inconvénients de MMJ CNMAT

Les "Patches" les plus généralistes, notamment ceux dont nous aurons besoin dans notre projet, se trouvent dans le package *CNMAT Baseline*. Dans ce "Patch" se trouve le savoir faire de CNMAT sous forme de fichiers compilés pour Max/MSP, ce qui a pour conséquence :

- Différentes librairies en fonction des systèmes d'exploitation,
- Code inaccessible car il est sous forme compilé,
- Code pas toujours propre : il est souvent difficile de comprendre le code sur les "Patches" auxquels nous avons accès.

Liens utiles pour MMJ CNMAT

`http://cnmat.berkeley.edu/library/max_msp_jitter_depot`

3.3.2 Étude de Librairie Max/MSP : FTM

Présentation de FTM

FTM est une librairie créée à l'IRCAM (*Institut de Recherche et Coordination Acoustique/Musique*) afin de faciliter le traitement de structures complexes (matrices, tuples, dictionnaires, séquences). FTM veut dire *Faster Than Musique* ainsi que *Funner Than Messages*. Elle regroupe un ensemble de packages externes à Max (de la même façon que le moteur vidéo *Jitter*) qui fournissent aux artistes des outils supplémentaires de traitement du son, de la musique et des signaux en temps réel.

Structure Générale de FTM

FTM a été développé dans le langage de programmation C et ensuite porté sur Max grâce à l'API de développement de Max/MSP. Il est composé de trois dossiers qui sont :

FTM

Ce package contient le noyau de la librairie, il permet de :

- Définir des structures complexes ainsi que des opérations réalisables sur ces structures,
- Créer des interfaces d'échange et d'intégration, comme par exemple un éditeur de classes complexes ou la lecture de fichiers dans le format SDIF.

Gabor

Gabor est une application qui a été établie grâce aux outils proposés par la librairie FTM. Elle met à disposition des fonctions qui permettent notamment de faire de la synthèse granulaire ou d'utiliser la transformée de Fourier avec une approche matricielle.

MnM

MnM (*Max is the New Matlab*) est une boite à outils mathématiques pour Max/MSP qui utilise les objets FTM. Il permet de réaliser de l'algèbre linéaire basique et propose des algorithmes comme l'ACP, GMM et HMM.

Une des applications de cette toolbox est le `Gesture Follower` qui réalise de la reconnaissance de gestes basiques en temps réel.

Avantages de FTM

- Création dynamique et statique de structures complexes des données,
- Évaluation des opérations et expressions sur données complexes,
- Importation/Exportation des formats de musique comme le MIDI et le SDIF,
- Sérialisation/Désérialisation des objets,
- Toolbox mathématique.

Inconvénients de FTM

- Interface entre les objets Max/MSP et ceux de FTM parfois source d'erreurs,
- Documentation peu détaillée et mal structurée.

Liens utiles pour FTM

`http://ftm.ircam.fr/index.php/Main_Page`

3.4 Communication entre Wiimote et Max/MSP

3.4.1 Étude de Logiciel de captation : OSCulator

Dans le cadre de notre projet nous avons besoin de faire la captation des données des différentes IHM low-cost. Nous avons donc fait l'analyse du logiciel OSCulator .

Présentation de OSCulator

OSCulator est un logiciel qui réalise la récupération des différents périphériques :

- Lemur : Contrôleur de musique multitouch pour : synthétiseur, séquenceur, etc...
- iPhone : Smartphone,
- Wiimote : Manette pour la console Wii de Nintendo,
- Wacom : Tablettes graphiques Wacom,
- SpaceNavigator : Souris 3D professionnel.

Ainsi que des logiciels comme : Kyma, reacTIVision. Afin de pouvoir les traiter dans une autre application comme Max/MSP, Ableton Live par exemple.

Fonctionnement de OSCulator

Le mode de fonctionnement de OSCulator est simple. Voici le comportement usuel que nous avons trouvé de façon empirique :

1. Connexion aux périphériques souhaités par usb, bluetooth, socket TCP/IP.
2. Remise en échelle et lissage des données sortant du périphérique.
3. Génération du message contenant les données après pré-traitement grâce au protocole MIDI ou OSC.
4. Envoi du message généré par socket TCP/IP dans le protocole UDP.
5. Si cela est supporté par le périphérique, il est possible d'envoyer des messages à l'aide d'un socket TCP/IP.

La Wiimote et OSCulator

Il est en théorie possible de capter jusqu'à 8 Wiimotes sur OSCulator . La communication avec la Wiimote se fait par bluetooth. OSCulator est capable de récupérer et traiter toutes les données de la Wiimote (buttons, accéléromètres, IR) ainsi que des extensions : Nunchuk (joy, buttons, accéléromètres) et Wiimotion Plus (gyroscopes).

Connexion Wiimote : Quand on exécute OSCulator , une nouvelle fenêtre de travail est ouverte. Dans cette fenêtre le port d'écoute d'OSCulator apparaît. Par défaut il s'agit du port 8000. Pour brancher la Wiimote sur OSCulator , nous avons le bouton *Wiimote Drawer*. Un nouveau module s'affiche sur la fenêtre de travail, On lance la recherche de Wiimote, et on met la Wiimote en mode *Discovery* (en appuyant les boutons 1 et 2 de la Wiimote).

Paramétrage Wiimote et ses extensions : OSCulator est paramétrable pour que les sorties des données soit en OSC ou MIDI. Si on est en OSC on peut choisir la plage de valeurs en sortie. OSCulator propose aussi le lissage des données pour :

- Wiimote : accéléromètres (filtre passe-bas), IR.
- Nunchuk : accéléromètres(filtre passe-bas) et joystick
- Wiimotion Plus : gyroscopes(filtre kalman).

OSCulator envoi les valeurs issues de la Wiimote dès qu'il les reçoit pour faire du temps réel.

Couplage avec Max/MSP : Pour le couplage avec Max/MSP il faut avoir l'abstraction *OSC-route* qui se trouve dans la librairie Max/MSP réalisée par la CNMAT. Cette abstraction gère les messages OSC.

Avantages de OSCulator

Les avantages de OSCulator sont :

- Prise en main rapide.
- Logiciel stable.
- Bien documenté.
- Grande communauté.
- Support de plusieurs extensions Wiimote : Nunchuk et Wiimotion Plus.

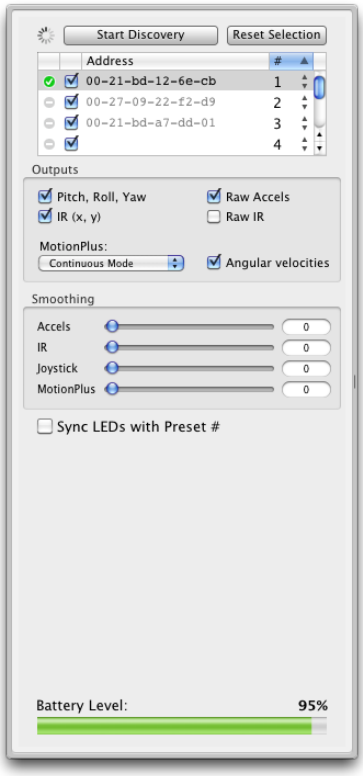


FIG. 3.3 – Module pour la Wiimote en OSCulator .

Inconvénients de OSCulator

Les inconvénients de OSCulator sont :
- Logiciel payant, Non open-source.
- Il n'est pas multiplateforme, et ne fonctionne que sous Mac Os X.
- Lissage des données en sortie médiocre.
- Changements des échelles non évidentes.

Liens utiles pour OSCulator

- `http://www.osculator.net/`

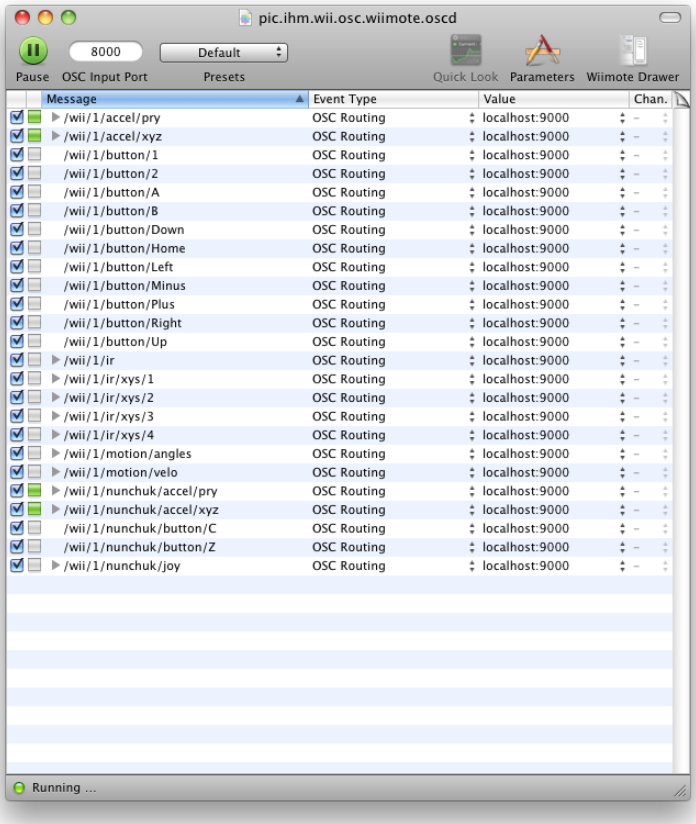


FIG. 3.4 – Fenêtre de travail de OSCulator configuré avec une Wiimote pour envoyer des données en OSC sur le port 9000 de la machine locale.

3.4.2 Étude du Patch AKA sous Max/MSP

Voir la documentation technique en Annexe , Chapitre 5.

3.5 Projets effectués dans le domaine de la reconnaissance de gestes

3.5.1 Étude de l'article : A Gesture follower for performing arts

Présentation du cadre d'utilisation de la reconnaissance de mouvements

Dans cet article, qui présente le résultat d'une recherche de l'IRCAM, la reconnaissance de mouvements est utilisée dans le cadre des arts scéniques, plus particulièrement dans le cadre de la danse. Une série de mouvements est faite par un premier danseur et le programme de reconnaissance gestuelle apprend et enregistre des données. Ensuite le programme est mis en mode de reconnaissance et un deuxième danseur se met à effectuer la même série de mouvements. Le programme détecte si oui ou non les mouvements ont été bien reproduits.

Avantages de ce type de reconnaissance gestuelle

Les avantages de cette reconnaissance gestuelle sont :
– Utilisation de HMM,
– Analyse en temps réel,
– Comparaison avec une base de données,
– Reconnaissance effectuée.

Inconvénients de l'étude

Les inconvénients de cette étude par rapport au cadre du PIC Floyd sont :
– Acquisition effectuée par caméra,
– Acquisition faite sur des danseurs et non un mouvement de bras.

Liens utiles pour cet article

`articles.ircam.fr/textes/Bevilacqua05b/`
`http://en.wikipedia.org/wiki/Gesture_recognition`

3.5.2 Étude du Projet de l'IRCAM : Wireless sensor interface and gesture-follower for music pedagogy

Présentation de ce projet

Ce projet de l'IRCAM consiste à élaborer un système d'acquisition ainsi qu'un programme permettant de faire de la reconnaissance gestuelle. Le travail effectué ici concerne à la fois le côté hardware et software. Cette reconnaissance gestuelle est utilisée ici dans le cadre d'une leçon de musique entre un professeur (phase d'apprentissage) et un élève (phase de reconnaissance).

Avantages de ce projet

Les avantages de ce projet sont :
– Reconnaissance d'un mouvement de bras,
– Utilisation de Max/MSP.

Inconvénients de ce projet

Les inconvénients de ce projet sont :
– Utilisation d'un système d'acquisition difficile à reproduire au sein du PIC Floyd,
– Utilisation de HMM sur une échelle discrète et non continue.

Liens utiles pour ce projet

`articles.ircam.fr/textes/Bevilacqua07a/`
`http://en.wikipedia.org/wiki/Gesture_recognition`

3.5.3 Étude d'un Projet d'étude : Gesture Recognition with Wii Controller

Présentation du Projet d'étude

Ce projet a été réalisé par l'*Oldenburg University* et l'*OFFIS Institute for Information Technology*. Le but ici est de reconnaître certains mouvements choisis au préalable. L'apprentissage se fait sur plusieurs dizaines de mouvements pour chacun des mouvements que l'on souhaite reconnaître. Une fois la phase d'apprentissage terminée, l'algorithme est testé en demandant à plusieurs personnes d'effectuer ces mouvements. Les résultats obtenus sont fournis sous forme de probabilités de reconnaissance. L'acquisition des données du mouvement est faite à partir d'une manette de console Wii. Les données sont envoyées aux ordinateurs par bluetooth et récupérées grâce à une API Java. Elles sont ensuite traitées par un programme en Java correspondant à une implémentation de HMM.

Avantages de Gesture Recognition with Wii Controller

Les avantages de Gesture Recognition with Wii Controller sont :
- Utilisation d'une Wiimote,
- Possibilité de porter ce produit sous Max/MSP grâce à l'utilisation de la technologie Java.

Inconvénients de Gesture Recognition with Wii Controller

Les inconvénients de Gesture Recognition with Wii Controller sont :
- Reconnaissance sur une échelle de temps discrete et non continue,
- Taux d'échec de reconnaissance d'environ 10%.

Liens utiles pour Gesture Recognition with Wii Controller

`http://wiigee.org/download_files/gesture_recognition_with_a_wii_controller-schloemer_poppinga_henze_boll.pdf`

`http://en.wikipedia.org/wiki/Gesture_recognition`

3.5.4 Étude d'un Projet d'étude : LiveMove White Paper

Présentation de LiveMove White Paper

LiveMove White Paper est une application mettant à disposition de l'utilisateur des outils lui permettant de faire de la reconnaissance de gestes sans avoir à implémenter de code. C'est une solution toute faite qui comprend une interface d'apprentissage, afin de sélectionner et de créer la base de données que l'on souhaite avoir à disposition, dans ce cas, elle contient les mouvements que l'interface doit pouvoir reconnaître par la suite. LiveMove White Paper utilise des séparateurs non-linéaires afin d'effectuer un choix pertinent au moment de reconnaître un geste.

Une partie particulièrement intéressante de cet article concerne la notion de *slack*. En résumé, un slack élevé fera en sorte que même si l'utilisateur fait un mouvement non reconnu celui-ci sera reconnu en fonction de sa probabilité d'être un des gestes de la base et un résultat sera obtenu (mieux vaut quelque chose de faux que rien du tout). En revanche, un slack faible sera beaucoup moins permissif et ne donnera aucun résultat si l'incertitude concernant le geste effectué par l'utilisateur est élevée. Un bon exemple permettant d'illustrer ces deux idées est le jeu vidéo. Dans un jeu de combat, même si l'on ne reconnaît pas le mouvement, on préférera que le personnage effectue une attaque sur son opposant (même mauvaise), plutôt que de rester inactif. À l'inverse, dans un jeu où l'on doit être discret, on préférera que le personnage ne fasse rien. Cette notion de slack peut être un paramètre particulièrement intéressant pour la réalisation d'un instrument.

Avantages de LiveMove White Paper

Les avantages de LiveMove White Paper sont :
- Bonne reconnaissance,
- Gestion du slack et autres paramètres pertinents.

Inconvénients de LiveMove White Paper

Les inconvénients de LiveMove White Paper sont :
- Impossibilité d'avoir accès au code et à la méthode de séparation utilisée,
- Projet trop fermé et trop simplifié pour le "bien de l'utilisateur" (notamment l'interface).

Liens utiles pour LiveMove White Paper

`http://www.ailive.net/papers/LiveMoveWhitePaper_en.pdf`
`http://en.wikipedia.org/wiki/Gesture_recognition`

3.6 Librairie de reconnaissance de gestes

3.6.1 Étude de Librairie Java : JaHMM

Présentation de JaHMM

JaHMM est une Librairie Java contenant des algorithmes de HMM (Hidden Markov Model) connexes.

Algorithmes de Modèle de Markov Caché de base

1. **Forward-Backward** (estime la probabilité d'une séquence d'observation obtenue par un HMM) avec mise à l'échelle (ce qui signifie que ce calcul ne génère pas de dépassement de la mémoire avec des séquences longues),

2. **Viterbi** (trouve la séquence la plus probable d'état qui correspond à une séquence donnée d'observation par un HMM),
3. **Séquence d'observation** correspondant à un générateur de HMM.

Algorithmes d'apprentissage de Modèle de Markov Caché

L'apprentissage est fondé sur un ensemble de séquences d'observation (et non une unique séquence).

1. **K-Means** apprentissage par défaut ;
2. **Baum-Welch** apprentissage avec mise à l'échelle (ce qui signifie qu'un apprentissage basé sur des séquences longues observations ne génèrent pas de dépassement de mémoire).

Autres Algorithmes liés au Modèle de Markov Caché

1. **Kullback-Leibler** est utilisé pour estimer la distance entre deux HMM.
2. **K-Means** (en utilisant l'algorithme standard, une version optimisée de la version classique Loyd's).

Avantages de JaHMM

Les avantages de JaHMM sont :
– Bibliothèque Open Source,
– Nombreux algorithmes de base de HMM mis en œuvre,
– Facilité de développement.

Inconvénients de JaHMM

Les inconvénients de JaHMM sont :
– Possibilité d'améliorer la complexité de la mise en œuvre des algorithmes impliqués par la théorie.

Liens utiles pour JaHMM

`http://code.google.com/p/jahmm/`
Remarque : L'algorithme général de Modèle Caché de Markov a été étudié et décrit dans la documentation technique en Annexe , Chapitre 6.

3.6.2 Étude du patch Gesture Follower

Voir la documentation technique en Annexe , Chapitre 7.

Chapitre 4
Conclusion

Nous avons essayé, dans cet état de l'art, d'étudier le plus de technologies utilisées dans la reconnaissance gestuelle et d'étudier de nombreux projets effectués dans ce domaine. Nous avons aussi fait le lien avec notre projet afin de voir quelles informations pourraient lui être enrichissant.

Les informations qui peuvent être utiles pour le PIC sont les suivantes :

- Lors d'un choix de librairie pour manipuler la Wiimote et ses extensions, la Librairie Java la mieux adaptée est la *WiiRemoteJ* car elle permet de manipuler les extensions de la Wiimote comme le Nunchuck très utilisé par la Grande Fabrique. Malgré le fait que la Librairie Java *WiiGee* comprend des modules de reconnaissance de gestes, elle ne contient pas de module de manipulation des extensions, et la gestion des erreurs lors de la reconnaissance de gestes n'est pas assez performante. Lors de notre PIC, nous projetons la création d'un module en Java afin de récupérer les données de la WiiMotionPlus. Les librairies *MoteJ* et *WiiRemoteJ* permettent l'ajout d'extension mais la librairie *MoteJ* est plus structurée et mieux documentée que la librairie *WiiRemoteJ*. Les autres librairies que nous avons étudiées sont spécifiques aux problèmes de la récupération des données de la WiiMotionPlus.

- Nous voulions étudier les librairies les plus couramment utilisées sous le logiciel de développement de la Grande Fabrique afin d'avoir une large vue sur les possibilités de ce logiciel. Les librairies Max/MSP (*FTM* et *MMJ CNMAT*) que nous avons étudié, nous ont permis de prendre connaissance de la capacité du logiciel Max/MSP.

- Afin de mieux comprendre la définition d'un geste, nous avons recherché des projets déjà effectués sur le sujet. L'étude de ces projets a mis en évidence les points suivants qui vont être utiles pour la réalisation du PIC :

  - **Suivi de danseurs** : Cette étude est intéressante car elle utilise de la reconnaissance en temps réel à partir d'une base de données.
  - **Interface sans fil de l'IRCAM** : l'interface mise en place par l'IRCAM permet de faire de la reconnaissance de gestes grâce au logiciel Max/MSP.
  - **Contrôleur Wii** : la reconnaissance de gestes se fait en Java grâce à une Wiimote ce qui permet de le porter facilement sous Max/MSP. Cependant, il serait intéressant d'améliorer ses performances et de faire de la reconnaissance de manière continue.
  - **LiveMore** : Ce projet nous a permis de réfléchir au problème de l'indécision face à un geste non reconnu. Nous avons aussi eu le regret de ne pas avoir accès au source du projet.

  Tous ces projets ont utilisé un HMM pour leur reconnaissance de gestes.

– La Librairie Java *JaHMM* nous a paru intéressante lors des études des projets car les Modèles de Markov Caché est l'une des méthodes de reconnaissance gestuelle les plus utilisées actuellement. Nous avons étudié plus longuement cette librairie car son code est clair et Open Source. De plus, elle peut être facilement adaptée à nos besoins du fait de la large palette d'algorithmes qu'elle contient.

Les points qui ont été jugés intéressants pour le PIC sont étudiés dans la documentation technique du PIC.

Deuxième partie

Annexes : Documentations Techniques

La documentation technique produite en complément de cet état de l'art a été ajoutée dans cette annexe.

Chapitre 5
Utilisation de la Nintendo Wiimote

Résumé

L'utilisation de la wiimote sous MAC OS X se fait grâce à Osculator et/ou Aka, outils permettant de recevoir les données de la Wiimote. Ces programmes ne fonctionnent pas sous Windows (et plus particulièrement Windows Seven Familial édition 64 bits). Ce document présente brièvement les différents outils rencontrés permettant d'effectuer cette communication, les résultats obtenus et les solutions envisageables.

5.1 Recherche de solution pour faire fonctionner la Nintendo Wiimote avec Max sous Windows 7 (Familial)

Le but de cette recherche est de pouvoir receuillir les différents paramètres de la Nintendo Wiimote avec Max sous Microsoft Windows 7 Familial (64 bits) c'est-à-dire la valeur des trois accéléromètres de la Wiimote, ceux du Nunchuk, si les boutons sont préssés ou non, etc. En effet, lors de l'utilisation de Max sous Mac OS X, nous utilisons Osculator et/ou Aka pour recevoir les données de la Wiimote, les programmes ne pouvant pas être utilisés sous windows. La recherche s'est axée sur deux points : chercher un objet Max pouvant répondre à nos besoins, et créer un nouvel objet Max (en Java, C, C++).

5.2 Objets Max existants

L'objet reconnu par la communauté Max pour recevoir les données de la Nitendo Wiimote sous Mac OS X est Aka. L'objet le plus intéressant rencontré lors de nos recherches fut "tk.wii", qui permet selon leur créateur d'acquérir ses données sous Windows, mais les résultats ne furent pas à la hauteur de nos espérances. En effet, ce dernier semble ne pas fonctionner correctement sur notre machine. Il découvre la Wiimote portant le DeviceId "0" par exemple, mais refuse de s'y connecter et est donc inutilisable. Si l'on croit les retours des autres utilisateurs, ces objets doivent marcher (incompatibilité avec Windows 7 ou le 64 bit ?) mais sont très limités du point de vue des fonctionnalités (Toutes les entrées de la Wiimote ne sont pas acquises à priori). Un autre objet, présent nativement dans Max détecte la Wiimote, et s'y connecte, lui, parfaitement : "hi", objets permettant l'acquisition de signal venant d'un HID (Périférique d'Interface Homme-Machine). Toutefois, une fois connecté, cet objet ne permet pas d'acquérir les signaux de la Wiimote. Il semble donc, pour le moment, impossible d'utiliser la Nintendo Wiimote correctement dans Max 5 sous Microsoft Windows 7 avec les objets existants. Il est cependant probable que des programmes d'interfaçage de qualité voient bientôt le jour, il faut donc maintenir une veille technologique sur ce sujet.

5.3 Création d'une acquisition bluetooth sour la forme d'un objet Max

La solution envisageable est de développer un objet Max pour windows permettant de faire l'acquisition de la wiimote directement. Cette solution semble être à court terme la seule solution possible. Plusieurs obstacles se sont toutefois présentés :

- Comprendre comment créer un objet Max.
- Choisir un language (C++,Java,Python) et une librairie permettant de faire l'acquisition bluetooth.
- Coder cette acquisition et son interfaçage avec Max, en obtenant de préférence toutes les entrées de la wiimote et ce de la manière la plus stable possible.
- Cet objet doit de préférence pouvoir être utilisable aussi bien sous Windows que sous Mac.

Développer un objet Max est donc tout à fait faisable, toutefois c'est une activité qui sera couteuse en temps (analyse, développement et tests), et qui est prévue dans le lot 4.

5.4 Conclusion

Après une journée de recherche peu fructueuse, l'utilisation de la Wiimote sur le poste Windows 7 semble compromise. De nouvelles recherches ponctuelles sont à envisager, toutefois, il ne faut pas oublier qu'il n'est pas nécessaire que toutes la machines puissent faire fonctionner la wiimote avec Max. En effet, la majeure partie du travail pourra être effectuée sans Wiimote, prétesté sous Windows (potentiomètres sous Max, périphériques autres) avant d'être finalement testée sous Mac avec des Wiimotes.

Chapitre 6
HMM

6.1 Étude de la Librairie Java : JaHMM

Un modèle de Markov caché (HMM) est un modèle statistique dans lequel le système modélisé est supposé être un processus de Markov [1] avec des états non observés. Un HMM peut être considéré comme la plus simple et dynamique Bayésien Network. [2]. Les HMM sont particulièrement connus pour leur application en reconnaissance de formes temporelles comme la parole, l'écriture, la reconnaissance de gestes, le tag de partie de discours, le suivi de partition de musique et la bioinformatique.

Introduction

Les processus réels sont caractérisés par des signaux observables et un problème fondamental résidant dans la caractérisation de ces signaux en terme de modélisation de signaux. Pour caractériser les propriétés d'un signal donné, il peut utiliser des modèles déterministes (ceux-ci exploitant certaines propriétés connues sur le signal), ou des modèles statistiques (qui tentent de caractériser uniquement les propriétés statistiques du signal)[3], et cette deuxième catégorie est le lieu de Hidden Markov Model.

Chaîne de Markov

Une Chaîne de Markov est une suite de variables aléatoires X1, X2, X3, ... qui vérifient la propriété de Markov, à savoire que, étant donné l'état actuel, les états futurs et passés sont indépendants. Formellement :

$$P(X_{n+1} = x | X_1 = x_1, X_2 = x_2, ..., X_n = x_n) = P(X_{n+1} = x | X_n = x_n)$$

Étant donné que le système change de manière aléatoire, il est généralement impossible de prédire l'état exact du système à l'avenir. Cependant, les propriétés statistiques du système à des mesures très nombreuses dans le futur peuvent souvent être décrites. Dans de nombreuses applications, ce sont ces propriétés statistiques qui sont importantes.

Les changements d'état du système sont appelés des transitions et les probabilités associées aux différents changements d'état sont appelées probabilités de transition.

[1] Un Processus de Markov est une variable aléatoire fonction du temps ayant une propriété spécifique (la propriété de Markov) vérifiée. Cela peut se produire de deux façons : sous la forme d'un processus stochastique qui doit être démontré (mathématiquement) pour avoir la propriété de Markov et les propriétés peuvent être déduites de cela pour tous les processus de Markov, et l'autre forme est basée sur l'hypothèse que la propriété de Markov équivaut à un processus aléatoire qui permet de construire un modèle stochastique pour ce processus (`http://mathworld.wolfram.com/MarkovProcess.html`)

[2] Bayésien Network : `http://www.niedermayer.ca/papers/bayesian/bayes.html#abstract`

[3] Lawrence R. Rabiner, A Tutorial on Hidden Markov Models and Selected applications in Speech Recognition – Proceeding of the IEEE

Un jeu de Monopoly ou tout autre jeu dont les déplacements sont entièrement déterminés par des dés est une chaîne de Markov. Ceci est en contraste avec les jeux de cartes comme le Blackjack, où les cartes représentant une mémoire des coups précédents. Pour voir la différence, considérons la probabilité pour un événement donné dans la partie. Dans les jeux de dés mentionnés ci-dessus, la seule chose qui importe est l'état actuel du plateau. L'étape suivante dépend de l'état actuel et du prochain lancer de dés. Elle ne dépend pas de comment on a obtenu l'état actuel. Dans un jeu comme le Blackjack, un joueur peut obtenir un avantage en se souvenant de quelles cartes ont déjà été sorties du jeu (et donc de quelles cartes ne sont plus dans le jeu), donc le prochain état du jeu (ou la prochaine main) n'est pas indépendante du dernier état. [4]

Les Chaînes de Markov sont également utilisées dans les algorithmes permettant de composer de la musique, en particulier grâce à des logiciels tels que CSound[5] ou Max. Dans une chaîne du premier ordre, les états du système sont des notes ou des valeurs de hauteur (Pitch), et un vecteur de probabilité pour chaque note est construit, remplissant une matrice de probabilité de transition. Un algorithme est construit pour produire et créer des notes dont les valeurs sont fondées sur la pondération matrice de transition, qui pourraient être des valeurs de notes MIDI, fréquence (Hz), ou toute autre métrique souhaitable.[6]

Spécification des HMM

Dans un modèle de Markov régulier, l'état est directement visible pour l'observateur, et donc les probabilités de transition d'état sont les seuls paramètres. Dans un modèle de Markov cachés, l'état n'est pas directement visible, mais la production dépendant de l'état est visible. Chaque état a une distribution de probabilité sur les jetons de sortie possibles. Par conséquent, la séquence de jetons générés par un HMM donne quelques informations sur la succession d'états. Notons que l'adjectif *caché* se réfère à la séquence des états par laquelle le modèle passe, et non pas aux paramètres du modèle ; car même si les paramètres du modèle sont connus exactement, le modèle est toujours *caché*.

Un HMM peut être décrit de la manière suivante :

$$N - \text{nombre d'états}$$

$$Q = q_1; q_2; ...; q_t - \text{ensemble d'états}$$

$$M - \text{le nombre de symboles (observables)}$$

$$O = o_1; o_2; ...; o_t - \text{ensemble de symboles}$$

$$A - \text{matrice de la probabilité de transition d'état ou : } a_{ij} = P(q_{t+q} = j|qt = i)$$

$$B - \text{distribution de probabilité d'observation ou : } b_j(k) = P(o_t = k|q_t = j), i \leq k \leq M$$

$$\pi - \text{la distribution état initial}$$

Ainsi, un HMM complet peut être décrit par :

$$\lambda = (A, B, \pi)$$

[4] http ://en.wikipedia.org/wiki/Examples_of_Markov_chains
[5] http ://www.csounds.com/downloads
[6] Curtis Roads (ed.) (1996). The Computer Music Tutorial. MIT Press

HMM problèmes

1. *P(O|λ)* : La probabilité d'occurrence d'une séquence d'observation particulière, $O = (o_1, ..., o_k)$, étant donné le modèle - ce qui est utile dans la séquence de qualification
2. *Décodage :* Séquence optimale d'état de produire des observations données, $O = (o_1, ..., o_k)$, ayant le modèle donné - utile dans les problèmes de reconnaissance.
3. *Apprentissage :* Trouver λ tel que P(O|λ) est maximale (c'est à dire la détermination du modèle optimal compte tenu de l'ensemble de formation d'observations)

Pour trouver les paramètres cachés, les algorithmes les plus utilisés sont Viterbi et l'algorithme de Baum-Welch.

Bien que la formation de Viterbi donne une bonne performance dans la plupart des cas, elle conduit parfois à des modèles sous-optimaux, spécialement pour l'utilisation de HMM discrets. Dans ces cas, Baum-Welch se montre plus robuste que la formation de Viterbi et l'approche combinée compense son coût de calcul élevé. Comme indiqué dans une étude comparative [7], la formation de Viterbi utilise beaucoup moins d'effort de calcul que Baum-Welch, en assurant toujours la même (ou légèrement plus mauvaise) performance, c'est pourquoi c'est le choix le plus communément utilisé.

Baum-Welch algorithme montre des propriétés très intéressantes :

1. dans le cas des HMM discrets, il n'a pas besoin de l'initialisation de modèles, mais juste une valeur aléatoire non nulle vérifiant les contraintes stochastiques ;
2. dans le cas des HMM continus, une initialisation convenable peut être faite avec les paramètres distribués en sortie des HMM discrets, d'une part, et les moyens et les écarts de prototypes contenus dans le vecteur de quantification d'autre part.
3. il utilise de façon exhaustive toutes les données disponibles pour produire des estimations fiables et optimales.

En matière de formation Viterbi :

1. il est démontré que, dans le cas des HMM discrets, elle nécessite une initialisation raisonnable, soit en utilisant les modèles obtenus pour d'autres bases de données, soit par des modèles d'apprentissage initialisés à partir d'un étiquetage manuel en sous-ensembles de la base d'apprentissage ;
2. il fait une utilisation limitée de données d'apprentissage car seules les observations appartenant aux segments correspondant à un état donné du HMM sont utilisées pour ré-estimer les paramètres de cet état. Le résultat des modèles sont plus nettes mais moins robuste. Sa performance dépendra de la quantité des données disponibles pour que la formation de Viterbi produise des estimations assez robustes.

[7] Comparative Study of the Baum-Welch and Viterbi Training Algorithms Applied to Read and Spontaneous Speech Recognition , Luis Javier Rodriguez and Ines Torres , Universidad del Pai Vasco

Chapitre 7
Gesture Follower

Résumé

Ce document décrit brièvement le fonctionnement du programme `mnm.follower-example-Wii.03`

7.1 Présentation générale

Le `mnm.follower-example-Wii.03` (ou plus communément Gesture-Follower) est un programme permettant de faire de la reconnaissance gestuelle à l'aide de HMM. Il a été réalisé par une équipe de l'IRCAM dans les années 2006/2007. Il utilise de nombreux patches appartenant aux librairies FTM et MnM.

Son utilisation est relativement simple. Il suffit de commencer par démarrer la Wiimote et de la connecter de la manière décrite dans le help pour ensuite soit enregistrer un nouveau geste dans la base, soit reconnaitre un geste. La sortie est le geste identifié avec la meilleure probabilité de reconnaissance.

7.2 Explications sur le fonctionnement

7.2.1 Global

Voilà comment se présente le Gesture-Follower :

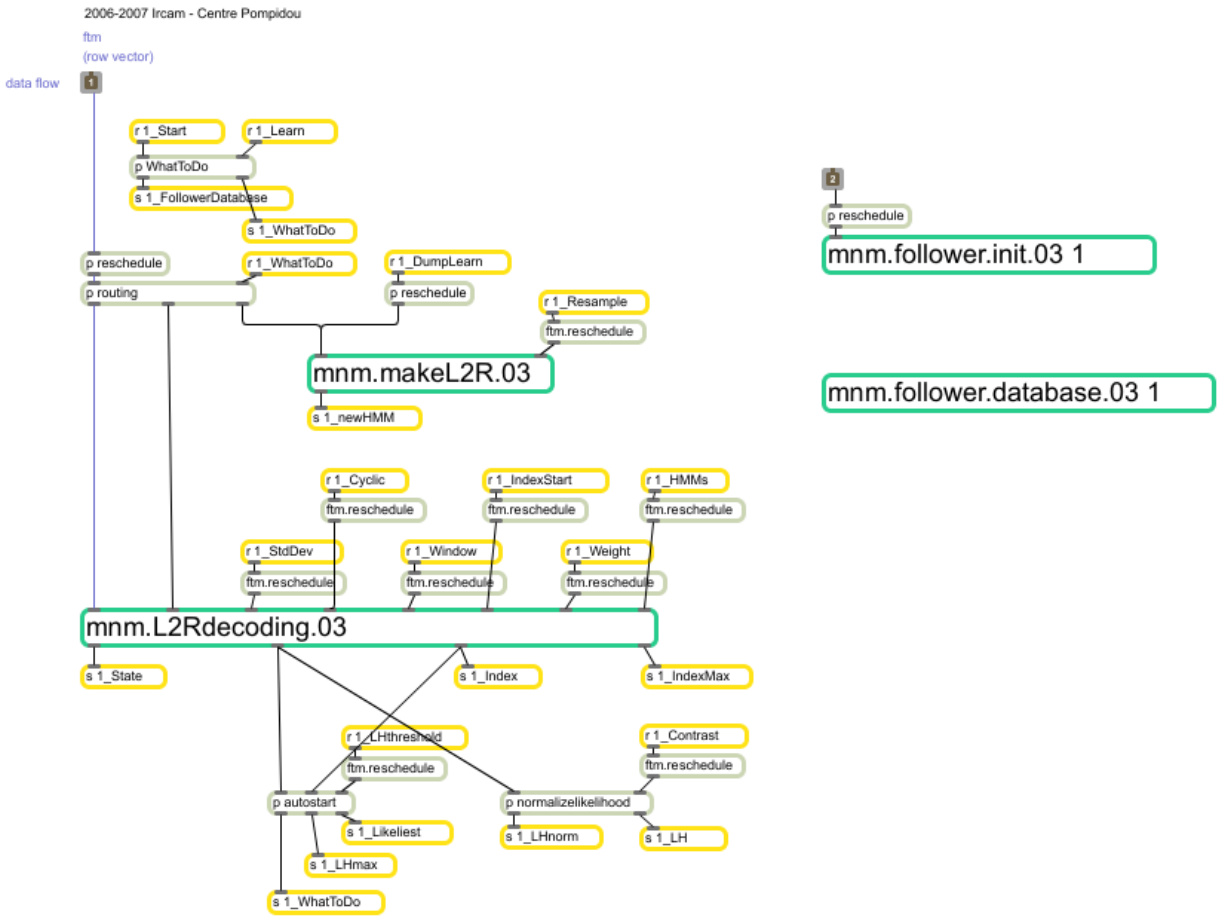


FIG. 7.1 – Structure globale du Gesture-Follower

Le programme se sert tout d'abord d'un patch appelé `WhatToDo` qui, comme son nom l'indique, sert à déterminer l'action choisie par l'utilisateur (apprentissage ou reconnais-

sance). Ensuite une orientation des flux est faite selon le cas détecté grâce à au patch `routing` : `mnm.makeL2R.03` ou `mnm.L2Rdecoding.03`. Le premier sert pour l'apprentissage et le deuxième pour la reconnaissance.

7.2.2 Partie apprentissage

Regardons d'un peu plus prêt comment a été construite cette partie.

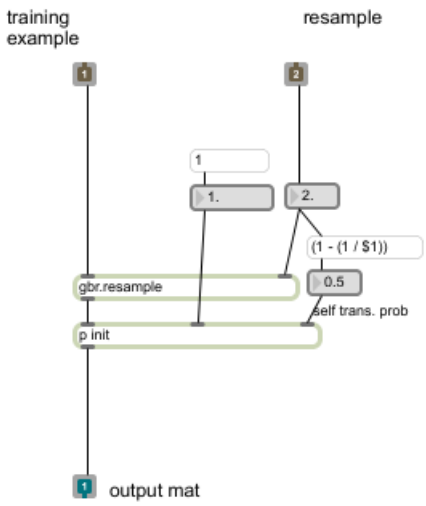


FIG. 7.2 – Structure du patch permettant l'apprentissage

Ici est en fait réalisée une initialisation de paramètres permettant la construction d'un HMM. Ces paramètres sont ensuite passés au patch `mnm.follower.database.03` qui, après avoir créé un nouvel HMM, l'envoie à l'aide d'un patch send `1_HMMs` à tous les patches en ayant besoin pour fonctionner correctement :

7.2.3 Partie reconnaissance

La partie reconnaissance se trouve dans le patch appelé `mnm.L2Rdecoding.03` :
Ce patch fait appel au patch suivant :
Ce dernier se décompose en plusieurs parties permettant d'utiliser les HMM créés dans la base, de normaliser les informations obtenues et de lancer une reconnaissance afin d'obtenir l'un des gestes pré-enregistrés (le plus vraisemblable par rapport au geste réalisé par l'utilisateur est choisi).

7.3 Remarques

Le plus gros problème que pose le Gesture-Follower c'est qu'il ne possède pas de documentation complète associée. De ce fait il est difficile de comprendre l'utilisation qui est faite des outils fournis par les librairies FTM et MnM. Il semblerait qu'il existe des équivalents

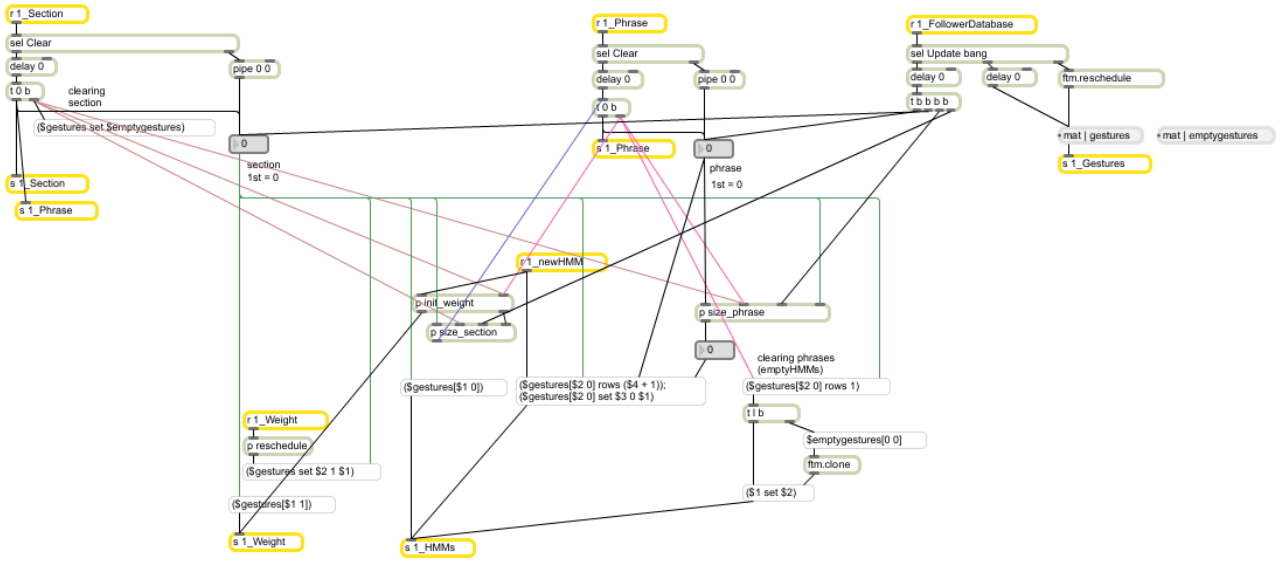


FIG. 7.3 – Structure du patch correspondant à la base de données

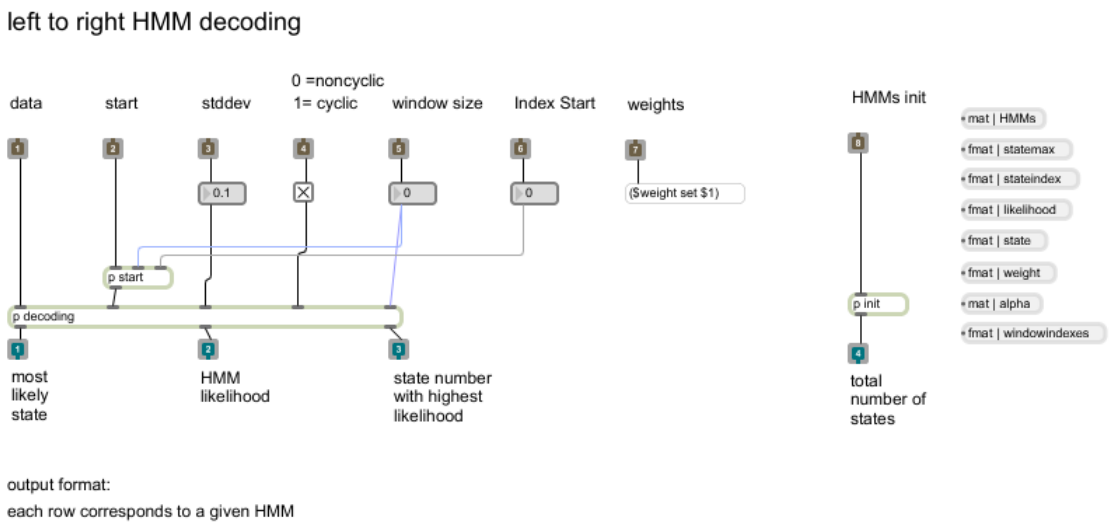


FIG. 7.4 – Structure de la partie reconnaissance

de variables globales, variables que n'importe quel patch peut modifier, cela permettant de jouer sur les HMM ou le contenu de la base de données.

De plus la complexité et la taille du patch fait qu'il est difficile d'avoir en tête au même moment toutes les variables utilisées et donc de visualiser l'agencement complet des différentes briques de l'édifice.

En ce qui concerne les HMM il n'est pas possible de connaitre l'algorithme utilisé puisque celui-ci n'est pas présent à un endroit donné mais éclaté en plusieurs morceaux dans plusieurs patches différents...rendant très difficile une analyse de celui-ci.

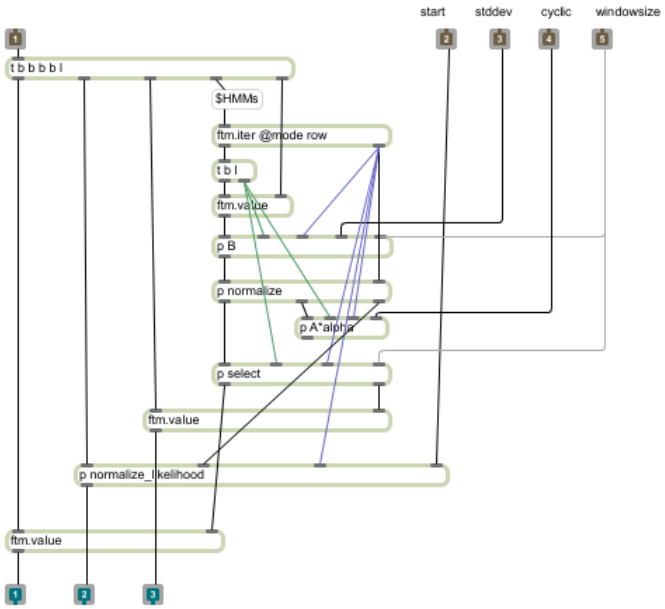


FIG. 7.5 – Structure du patch `decoding`

7.4 Conclusion

En conclusion cette analyse apporte une compréhension générale du Gesture-Follower mais ne présente pas d'intérêt particulier pour la partie HMM/FTM/MnM parce que le tout reste trop compliqué et surtout non-documenté. Les deux liens suivants sont une description générale du projet Gesture Follower de l'IRCAM :

`http://edite-de-paris.fr/spip/spip.php?page=phdproposal&id=10045224`
`http://fitg10.lille.inria.fr/workshop-data/proposals/bevilacqua.pdf`

Chapitre 8
Wiimote

Résumé

Ce document a pour but la description technique de la Wii Remote (Wiimote), du Nunchuk et de la Wii Motion Plus. Elle ne s'intéresse pas à la transmission de données, mais simplement aux entrées disponibles et aux composants qui font l'acquisition.

8.1 Wiimote

La Wiimote (RVL-003) est la manette de jeu conçue pour la console de jeux Wii qui s'est distinguée lors de sa sortie pour sa nouvelle manière d'acquérir les commandes des joueurs. À une époque où les consoles de salons sont de plus en plus puissantes, Nintendo a décidé de plus se pencher sur la jouabilité et la conception d'une nouvelle manette : la *Wiimote*.

La Wiimote est la première manette de jeux grand public à posséder un **accéléromètre** lui permettant de capter les mouvements du joueur. Il s'agit d'une manette utilisable par une seule main (gauche ou droite). Les parties suivantes décrivent ses composants principaux. Cependant, il est important de noter que toutes les caractéristiques suivantes ne viennent pas de Nintendo, mais de recherches de rétro-ingénierie.

8.1.1 Présentation rapide

FIG. 8.1 – Vue de face de la Wiimote

La Wiimote, représentée sur la figure 8.1, possède principalement :
- une **caméra infrarouge**
- un **bouton *Power***
- des **flèches directionnelles** , bouton à membrane pour chaque direction (↑, →, ↓, ←)
- un **bouton *B***, bouton à membrane (situé à l'arrière de la manette)
- un **bouton *A*,** bouton à membrane
- un **bouton -**, bouton à switch
- un **bouton *Home***, bouton à switch
- un **bouton +**, bouton à switch
- un **haut-parleur** piézo-éléctrique basse qualité. Les sons qu'il transmet sont envoyés en direct par l'hôte.
- un **moteur de vibration**
- un **bouton *1***, bouton à membrane
- un **bouton *2***, bouton à membrane
- **4 diodes** électroluminescentes bleues
- un **compartiment à 2 piles AA**
- un **bouton *Sync*** (dans le compartiment à piles)
- un **port d'extension**
- une **dragonne** de sécurité

8.1.2 Connectivité

La Wiimote est sans fil et fonctionne avec 2 piles AA. La transmission de données avec la Wii se fait en Bluetooth selon le protocole HID[1], utilisé également pour les souris, les manettes de PlayStation 3, ou encore les claviers. Ce protocole a été conçu pour offrir une liaison à faible latence. Cependant, même si Nintendo suit les grandes lignes du protocole, la firme a implanté ses propres descripteurs [1].

Le composant assurant la transmission et la réception Bluetooth est le BCM2042 de la marque Broadcom [1]. Les boutons *Power* et *Sync* ont des comportements particuliers directement liés à la connectivité. Le *Power* permet d'allumer une Wii à proximité et de lancer la procédure de synchronisation pour s'y connecter. Lors d'une connexion avec un hôte (Wii, PC, ou autre), une pression de quelques secondes déconnecte la Wiimote. *Sync* coupe toute connexion, met la Wiimote en mode visible (en anglais *Discoverable*) et relance une procédure de synchronisation pour une durée de 20 secondes.

La Wiimote possède dans sa partie inférieure un port d'extension qui lui permet par exemple de brancher un Nunchuk (voir 8.2), une Motion Plus (voir 8.3, une guitare ou même une Wii Balance Board.

8.1.3 Accéléromètre

La Wiimote possède un accéléromètre 3 axes, le ADXL330. Il permet la mesure d'accélération allant de -3 à $+3g$ [2] à $\pm 10\%$. Il supporte un choc jusqu'à $10000g$, fonctionne entre -25 et $+70°C$ et mesure $4mm \times 4mm \times 1.45mm$ [3]. En utilisation courante, il s'avère que l'accéléromètre capte très mal (voir pas du tout) les mouvements lents. Il ne détecte que les mouvements plutôt brusque, rendant impossible le passage de l'accélération à la vitesse par intégration.

Les mesures d'accélération sont dépendantes de l'orientation du composant et donc de la manette. Il est fixé à la carte principale (voir figure 8.2) à l'emplacement visible sur la figure 8.3. La figure 8.4 montre les valeurs de l'accéléromètre en fonction de l'orientation de la manette.

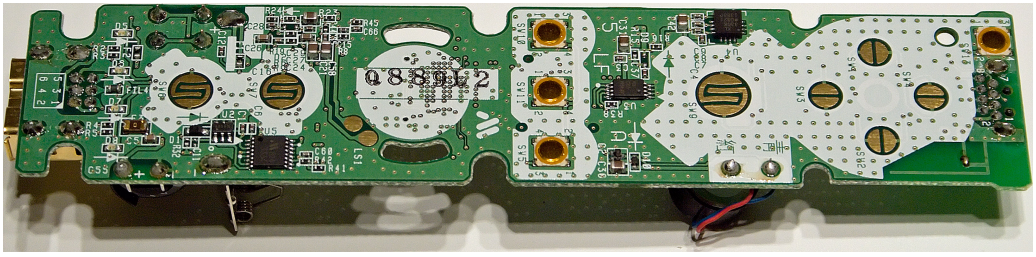


FIG. 8.2 – Vue de dessus de la carte électronique de la Wiimote [1]

8.1.4 Caméra infrarouge

La caméra infrarouge de la Wiimote est faite pour fonctionner avec la SensorBar de Nintendo (barre d'une trentaine de centimètre, émettant de l'infrarouge à ses deux extrémités).

[1] Human Interface Device Profile

[2] 1 g = 9.80665 m/s^2, soit 35.30394 km/h par seconde

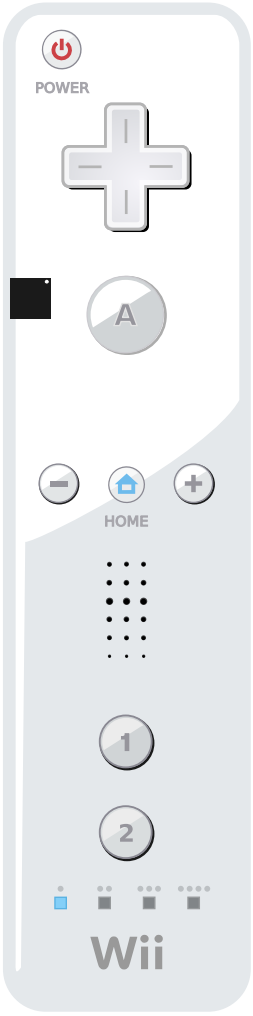


FIG. 8.3 – Position de l'accéléromètre sur la Wiimote (en noir)

Il s'agit d'une caméra monochrome de 128x96 sensible aux infrarouge. Le module de prétraitement est intégré à la Wiimote et empêche la lecture directe de l'image captée (trop grosses à transférer). Il permet le suivi de 4 points et l'obtenir de leurs cordonnées (en abscisse et ordonnée). Pour cela, il utilise une analyse de 8 sous pixels permettant de fournir une résolution de 1024x768 pour le positionnement final des points. Il faut cependant être attentif à la pollution visuelle pouvant totalement perdre le positionnement d'un point.

8.2 Nunchuk

8.2.1 Présentation rapide

Le Nunchuk (voir figure 8.5) est l'accessoire de base de la Wiimote. Il se branche avec un câble au port d'extension en dessous de la Wiimote, et possède également un accéléromètre sur sa carte électronique. Ce capteur est décrit dans la partie 8.2.3.

Cette manette secondaire possède principalement :

– un **stick analogique**. Il s'agit d'un mini-joystick analogique fait pour être utilisé par le pouce de la main tenant le Nunchuk. Il est décrit plus précisément dans la partie 8.2.2.

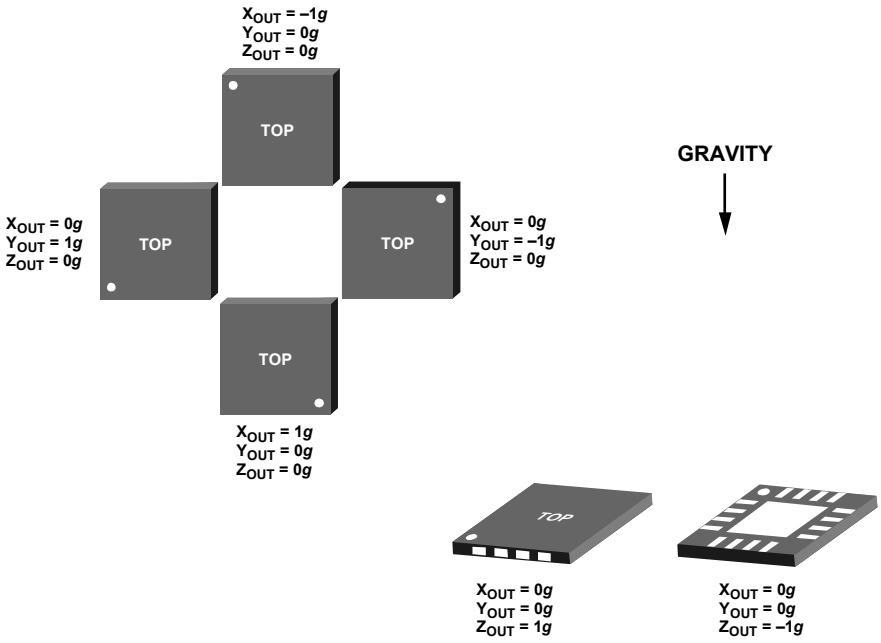


FIG. 8.4 – Valeurs de l'accéléromètre [3]

FIG. 8.5 – Photographie du Nunchuk [6]

- un **bouton *C*** sous l'index, bouton à membrane.
- un **bouton *Z*** sous le majeur, bouton à membrane.
- un **câble** lui permettant d'être relié à la Wiimote.

8.2.2 Joystick

Le mini-joystick présent sur le Nunchuk était déjà présent sur la manette de la Gamecube de Nintendo. Il est caractérisé par sa base en forme d'octogone (voir figure 8.6). Cela permet à l'utilisateur de facilement retrouver les positions habituelles des flèches directionnelles (↑, →, ↓, ←) et d'y ajouter les diagonales (↗, ↘, ↙, ↖).

Électroniquement, il s'agit de deux potentiomètres axiaux de $30K\Omega$ ramenés au centre par des ressorts. Du fait du montage en série de la manette et des composants du joystick, des problèmes de calibrages sont courants :

- impossibilité d'atteindre les extremums à cause de la base octogonale.
- retour au centre approximatif.
- décalage du centre lié à la pose du doigt sur le joystick.

FIG. 8.6 – Photographie de l'octogone du Nunchuk

8.2.3 Accéléromètre

L'accéléromètre du Nunchuk est globalement similaire à celui de la Wiimote. Il capte l'accélération sur 3 axes, mais avec une plage de mesures plus petite : de $-2g$ à $+2g$. Il s'agit du modèle *ST8XRJ3L02AE820MLT* de la marque STMicroelectronics dont la documentation n'est pas disponible. Le composant de la même marque se rapprochant le plus est le *LIS3L02AL* [2] [4].

L'accéléromètre est soumis au même problème que celui de la Wiimote (voir partie 8.1.3), il ne détecte que les mouvements plutôt brusques.

8.3 MotionPlus

8.3.1 Présentation rapide

Le Wii MotionPlus est une extension de la Wiimote sortie en juin 2009. Il est présenté comme suit par Nintendo :

> The Wii MotionPlus™ accessory takes the motion-sensing controls of the Wii console to new levels of precision, sensing gameplay movements with greater accuracy than ever before. [5]

Il est décrit comme une extension qui permettrait de faire de l'acquisition de mouvement 1 :1. Il s'avère que cette annonce n'est vraie que pour les angles. Le Wii MotionPlus détecte

parfaitement les positions angulaires de la Wiimote, mais ne capture absolument pas ses déplacements.

8.3.2 Gyroscopes

Le MotionPlus est simplement un bloc s'ajoutant en bas de la Wiimote (il la rallonge alors de 4cm, voir la figure 8.7). Il contient deux composants très importants :

- un gyroscope 2 axes, le IDG-600 de la marque InvenSense (pour le pitch (tangage) et le roll (roulis)).
- une gyroscope 1 axe, le X3500W de la marque EPSON TOYOCOM (pour le yaw (lacet)).

FIG. 8.7 – Photographie du Wii MotionPlus sur une Wiimote [5]

Les sens des sorties du MotionPlus sont celles représentées sur la figure 8.8.

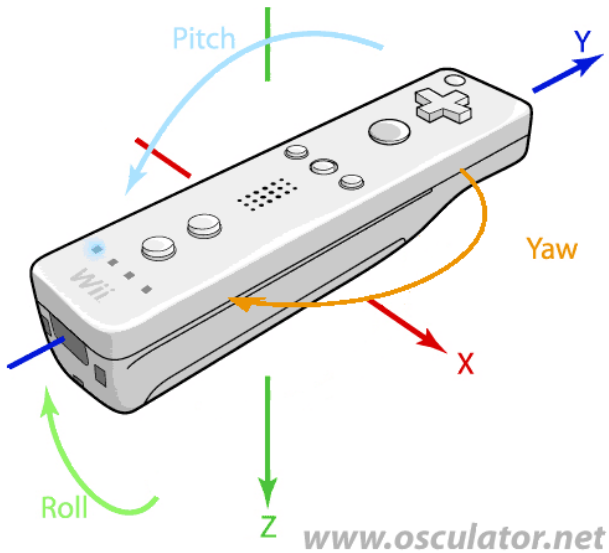


FIG. 8.8 – Repères de la Wiimote dans l'espace [7]

8.4 Conclusion

La Wiimote et ses accessoires sortis par Nintendo présente un très bon instrument de mesure innovant low-cost. Cependant, si son utilisation est aujourd'hui largement possible, c'est parce qu'une communauté l'a étudiée dès sa sortie pour pouvoir communiquer avec elle, et profiter de ses capteurs. Cependant, il apparaît clair que les possibilités offertes par les accéléromètres ne permettent pas de localiser la Wiimote dans l'espace. Cependant, l'apport du MotionPlus permet désormais de connaître précisément l'orientation de la manette dans l'espace. Finalement, la caméra infrarouge de la Wiimote n'est pas à négliger, puisqu'elle permet de situer assez précisément des émissions infrarouge, et avec un peu de calcul la manette elle-même.

Bibliographie

[1] Article *Wiimote*, sur Wiibrew,
`http://wiibrew.org/wiki/Wiimote`

[2] Article *Extension_Controllers*, sur Wiibrew,
`http://wiibrew.org/wiki/Wiimote/Extension_Controllers`

[3] *Documentation technique ADXL330*, ©Analog Devices, Inc 2007,
`http://www.analog.com/en/sensors/inertial-sensors/adxl330/products/product.html`

[4] *Documentation technique LIS3L02AL*, ©2010 STMicroelectronics,
`http://www.st.com/stonline/products/literature/ds/11668/lis3l02al.htm`

[5] *Page web Wii MotionPlus*, ©2010 Nintendo,
`http://www.nintendo.com/wii/console/accessories/wiimotionplus`

[6] *Page web des contrôleurs Wii*, ©2010 Nintendo,
`http://www.nintendo.com/wii/console/controllers`

[7] *Osculator.net*,
`http://www.osculator.net`