PACT 2015-2016

Rapport d'avancement du groupe "3.1"
"Fituring"

Date de mise à jour : 09/03/2016

Camille Demason, Khadija El Attar, Stanislas Hannebelle, Karim Khandid, Thibaud Lemaire, Alexandre Poinso, Robin Shin, Jamal Skouri

Tuteur : Michel Roux

Encadrant génie logiciel : Soumia Dermouche

Sommaire

1 Résumé du sujet choisi

Lorsqu'une personne danse, elle est habituée à bouger en fonction du rythme de la musique, selon son style et l'humeur qu'elle dégage. On a là un procédé où un certain son va provoquer une façon de danser spécifique. Pour notre projet PACT, nous voulons essayer d'inverser ce processus : le danseur peut laisser s'exprimer son corps sans contrainte, et notre appareil, appelé Fituring, sera capable de générer une musique originale qui saura s'adapter à chaque mouvement de l'utilisateur et à son style. Fituring aura la forme d'un boîtier contenant une kinect et un petit ordinateur capable de traiter les données de mouvement en sortie de la kinect et de les convertir en un son adéquat. Le but de ce projet est d'offrir à l'utilisateur un nouveau mode de création artistique passant par le corps, avec la possibilité d'un effet d'apprentissage de manière qu'il peut prévoir et moduler la musique selon son envie, et non pas aléatoirement. Nous vivons dans une société où la communauté et les réseaux sociaux ont une grande importance : Fituring ne laissera pas ces domaines de côté car il sera accompagné d'une application où l'utilisateur pourra retrouver ses anciennes créations et les partager sur son réseau social préféré.

2 Abstract

Fituring is a whole new device that will revolutionize the way you dance. People are used to adapt their moves to the music they are listening to. Now, they can control the music, its rhythm and its style according to their moves : your body is a new extremely complex instrument. Your legs will command the tempo, your arms and the top of your body will be in charge of the sound. For more and more requirements, you can chose a few options on the smartphone application, like a precise style of music or the volume. When you will get tired of dancing this much, the application will allow you to live those incredible moments again : Fituring has captured your moves and registered the music created so that you can keep videos of your performance and share it with your friend on social networks.

3 Étude d'antériorité et justification de la proposition

3.1 Description de la proposition

Lors de nos premières recherches de sujet pour le projet PACT, une idée s'est démarquée des autres, rentrant dans le thème : « conversion de gestes en son ». Il s'agissait de la réalisation d'un instrument de musique corporel et donc sans objet à manipuler. Le principe de ce projet était d'assigner chaque mouvement possible du corps à une certaine note ou un certain rythme. Par exemple, la position des bras aurait contrôlé la hauteur de la note et la vitesse des mouvements aurait défini le rythme. Apprendre à jouer à notre instrument de musique revenait alors à maîtriser des mouvements, ce qui aurait permis de jouer n'importe quelle partition de musique. Toutefois, nous nous sommes rendu compte que ce type de fonctionnement nécessiterait un apprentissage difficile de la part de l'utilisateur, de la même manière que pour un instrument déjà existant. Ce point aurait rendu l'utilisation de notre appareil difficile, entraînant donc probablement une faible demande de la part des consommateurs : la réalisation de notre instrument n'était donc pas justifié. Il nous a donc fallu repenser le fonctionnement de l'appareil pour le rendre plus intuitif : d'où l'idée d'inventer plutôt un dispositif demandant simplement à l'utilisateur de danser librement ; la création de la musique correspondante sera entièrement gérée par l'appareil.

Cette nouvelle idée nous convenant à tous, répondant parfaitement au thème PACT et promettant un résultat qui plaira à un large public, nous avons alors commencé à définir les moyens dont nous aurons besoin pour réaliser un projet si complexe. Il nous est tout de suite paru évident que l'acquisition de l'image et des mouvements de l'utilisateur se ferait grâce à une kinect qui fournirait en sortie ces données sous forme de squelette. La première difficulté est alors apparue au sujet du traitement de ce squelette qui n'est pas faisable par le biais d'un programme en java : nous avons donc déterminé le besoin d'un module pour nous apprendre à utiliser la kinect et convertir ses données grâce à un programme en langage C pour pouvoir les traiter par la suite.

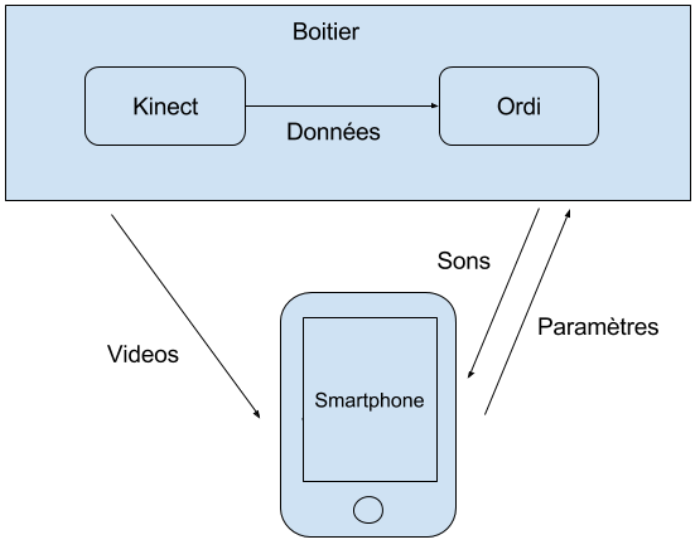
Par la suite, la difficulté majeure que nous avons mise en évidence est la façon de convertir des gestes en une musique qui conviendrait : sur quel critères pouvons-nous associer un mouvement à une caractéristique musicale telle que la hauteur des sons ou le rythme ? Comment choisir les sons adéquats dans une grande bibliothèque ? Comment reconnaître des mouvements précis ? Et comment générer une musique qui plaira ?

En élaborant le scénario d'usage, nous nous sommes rendu compte que notre système serait intéressant à utiliser en soirée, surtout pour relancer la danse quand personne ne semble vouloir se proposer. Cependant la kinect ne permet de détecter qu'une à quatre personnes. Sans oublier qu'en classant un ensemble de gestes prédéfinis, ils seront effectués de manières différentes par les consommateurs ce qui ne nous facilitera pas la tâche d'acquisition et pourrait même engendrer une musique qui ne convient pas au style de danse de l'utilisateur.

Ce qui nous semble le plus dur à réaliser est le fait de gérer les transitions au cas où la personne décide soudainement de changer son style, richesse, ou vitesse de danse. Il nous faudrait réaliser une continuité dans la musique quel que soit le changement.

Pour que notre appareil réalise toutes les fonctionnalités voulues et puisse nous permettre de trouver des solutions adéquates à nos problèmes, nous allons concevoir notre dispositif comme un boitier composé d'une caméra kinect et d'un mini-ordinateur contrôlés par un smartphone Androïd. L'utilisateur choisit ses paramètres à travers l'application Android qui les transmet à l'ordinateur, la kinect permet d'analyser la danse de l'utilisateur. Le système génère automatiquement la musique qui correspond à ses mouvements. Il diffuse la musique ainsi générée, et permet de l'enregistrer pour la diffuser plus tard. Enfin, des fonctionnalités sociales seront incluse pour pouvoir partager les musiques créées avec la vidéo du danseur correspondante.

Ci-dessous un schéma montrant l'architecture de notre système :



3.2 Description de l'état de l'art

Notre projet PACT se propose de développer un instrument qui crée une piste musicale à partir de la danse de l'utilisateur.

L'idée de créer de la musique à partir de mouvement et sans contact avec l'instrument, bien qu'originale, n'est pas nouvelle. En effet, en 1919, le russe Léon Thérémine invente l'un des tout premiers instruments de musique électroniques : le thérémine (cf http://www.sonelec-musique.com/electronique_realisations_th…). Cet instrument est composé de deux antennes.

Chacune d'elle permet de relever la position d'une main. De façon générale, la hauteur de la main gauche donne le volume tandis que celle de la main droite donne la note à jouer par l'instrument. Cet instrument est révolutionnaire pour l'époque cependant, il ne permet pas de jouer une piste musicale riche mais uniquement un son également, à l'inverse de notre projet, il se base sur des mouvements précis que réalise l'utilisateur et non sur sa danse.

L'idée d'un instrument de musique virtuel a également été développée. Par exemple, en utilisant la technologie Kinect pour capturer des mouvements, des étudiants de Georgia Tech ont créé un instrument de musique virtuel à dessiner (https://www.youtube.com/watch?v=CEd4dS_pOVM) . L'utilisateur doit dessiner des formes sur une feuille de papier, ensuite lorsqu'il touchera l'une de ses formes un son particulier sera joué. Il peut alors composée en touchant les différentes notes. Cependant, là aussi ce n'est pas une danse qui crée la musique mais uniquement des gestes.

Le compositeur Chris Vik a développé un système qui capte les mouvements d'un danseur et qui crée de la musique à partir de cette danse (https://www.youtube.com/watch?v=qXnLxi2nzrY). Dans ce système c'est également la technologie Kinect qui permet la capture des mouvements du danseur. Cependant, l'utilisateur de cet instrument est un danseur professionnel. A l'inverse, notre projet se veut accessible à tous et notre système devra être capable de produire une piste musicale de qualité même si les pas de danse de l'utilisateur ne sont pas dignes d'un grand danseur. Enfin, le système Kinect Bomba, créé par Stephanie Gil et Sam Prentice permet la création d'une piste de musique à partir des mouvements de l'utilisateur (http://people.csail.mit.edu/prentice/bomba/). Il est même capable de prendre en compte les mouvements de deux danseurs en même temps. Ce système se rapproche de notre projet puisque son utilisation est assez intuitive pour l'utilisateur. Cependant, il ne permet uniquement la production d'un style musical latin alors que dans notre projet nous souhaitons que l'utilisateur puisse choisir lui-même le style musical à jouer par l'utilisateur. Ainsi, il existe déjà des systèmes qui créent des sons à partir de mouvement cependant notre projet souhaite apporter plus de liberté à l'utilisateur par rapport au choix de style musical ou par rapport aux gestes qu'il peut réaliser.

4 Scénario d'usage

17h00 : Paola, étudiante, rentre de cours après une dure journée d'examens. Elle a besoin d'un moment de détente et active sa playlist. Sa musique préférée se lance alors : "Happy" de Pharrell Williams. Immédiatement, une poussée d'adrénaline l'enivre et lui donne envie de danser. Après 4'07'' de pur plaisir, la transition est brutale : la chanson suivante est "Quelqu'un m'a dit" de Carla Bruni Sarkozy. Paola est alors coupée dans son élan ; elle l'avait déjà bien trop écoutée cette musique. Et puis l'extrême douceur et lenteur de la chanson ne correspondait plus du tout à son humeur. Impossible de continuer à danser. La déprime des examens remonte à la surface, rien ne va plus. Paola est lassée et en a assez de ces playlists où les mêmes chansons tournent continuellement en boucle et ne veut plus dépendre du rythme de la musique : elle voudrait renverser le processus afin que ce soit la musique qui s'adapte à son humeur, à son rythme.

21h00 : Paola se rend chez son ami Paolo qui veut festoyer. En arrivant sur les lieux, elle est déçue, la fête ne semble pas encore commencée : aucune musique pour danser car par timidité, personne n'a ôsé commencer à danser. Paola, de plus en plus frustrée par sa journée, s'étonne auprès de son ami : "c'est quoi cette soirée qui bouge pas?" Paolo lui lance alors son plus grand sourire et lui rétorque : "Je fais une expérience. A toi de mettre l'ambiance ! Danse et tu verras bien ce qui se passera". Mise au défi, elle esquisse un mouvement timide d'épaule qui entraîne un beat sonore. Surprise, elle tente cette fois un enchaînement de pas qui lui plaît. Immédiatement, un son de batterie parfaitement en rythme et synchronisé avec ses pieds retentit. Progressivement, c'est tout le corps de Paola qui se met à se mouvoir, et déferle une véritable symphonie électrique : tel un maestro son corps prend les rênes de toute la musique qui envahit la salle. Étonnés par sa performance, ses amis la rejoignent et se câlent à son rythme et progressivement, la musique s'enrichit. Une fois l'ambiance installée et la soirée démarrée, la musique est redirigée vers une playlist de musiques connues.

Sans le savoir, Paolo a réalisé le souhait émis par son amie un peu plus tôt dans la journée : il lui explique donc que tout cela est possible grâce au nouvel appareil *Fituring*.

Dès le lendemain, Paola s'empresse d'acquérir ce petit bijou de technologie et de l'essayer chez elle. Elle découvre donc plus en détail *Fituring*, un séquenceur musical autonome et hautement intéractif. Il crée et diffuse de la musique qui s'adapte à la volée aux pas de danse de ses utilisateurs. Le dispositif est muni d'une caméra Kinect qui analyse les mouvements, relié à un ordinateur qui permet de traiter l'information des mouvements en son. Ceux-ci sont discriminés par un algorithme de classification liée à une base de donnée. Le système pioche ensuite dans une large bibliothèque de sons élémentaires pour créer une musique harmonieuse grace à un procédé de boucle ("loop") et parfaitement adaptée aux pas des danseurs.

Pour choisir les sons les plus adaptés, *Fituring* analyse plusieurs caractéristiques de la danse tels que le rythme, l'amplitude et la richesse des mouvements. Il y a une première phase d'initialisation du rythme pendant laquelle l'utilisateur doit uniquement bouger ses jambes

afin de calculer une moyenne de tempo, qui peut évoluer si le danseur change de rythme. Ainsi, la musique produite est parfaitement synchronisée aux mouvements de l'utilisateur et elle s'enrichit sur la base de structures musicales prédéfinie au fur et mesure qu'il ajoute des mouvements.

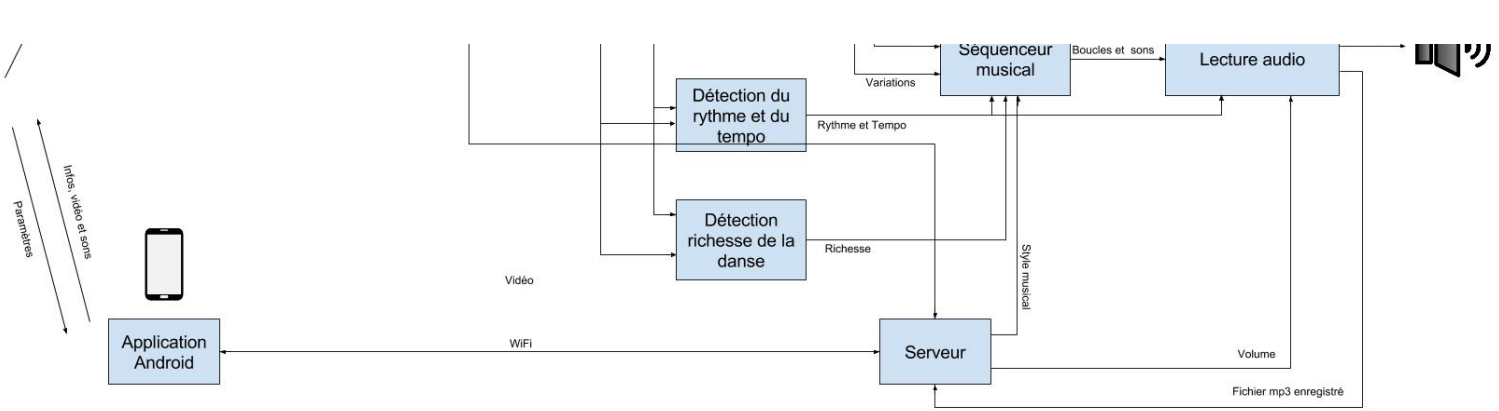
Paola prend également connaissance de l'application smartphone liée à l'outil, qui lui apprend via un tutoriel interactif comment utiliser *Fituring* au mieux. Cette application permet également de régler différentes options pour des types de musique différents, elle peut enregistrer les musiques créées.

Paola se découvre alors un réel talent de création artistique lié à la musique et à la danse, elle utilise *Fituring* lorsqu'elle a envie de danser, lorsqu'elle veut découvrir une nouvelle musique originale et même lorsqu'elle fait du sport dans son salon. *Fituring* devient alors le meilleur allié du fitness.

5 Architecture du projet

5.1 Schéma d'architecture

Le schéma ci-dessous décrit le fonctionnement de notre appareil ainsi que la façon dont chaque composant communique avec un autre.


Séquenceur musical
Boucles et sons
Lecture audio
Variations
Détection du rythme et du tempo
Rythme et Tempo
Détection richesse de la danse
Richesse
Style musical
Vidéo
Infos, vidéo et sons
Paramètres
Application Android
WiFi
Serveur
Volume
Fichier mp3 enregistré

5.2 Description des blocs

5.2.1 Kinect

La Kinect fournit des informations de profondeur et les images issues de sa caméra RGB, en temps réel (environ 30Hz). Les données transitent via un bus USB.

5.2.2 API Kinect Serveur – Client Acquisition squelette

Acquisition et traitement des données en C, grâce à **l'API Kinect de Microsoft**. Transfert de ces données via une architecture Client/Serveur à notre programme Java. Pour cela, coté java, nous utiliserons la bibliothèque **java.net**.

5.2.3 Base de données

Cette base de données contient deux tables : la première contient les mouvements usuels acquis à travers le module SES observation et caractérisés par leur style, les membres mis en mouvements, etc… La deuxième table contient les sons élémentaires disponibles, c'est-à-dire les briques de base qui serviront à construire la musique. Nous utiliserons le bibliothèque **java.sql** pour accéder à cette BDD.

5.2.4 Détection des mouvements (DTW) :

La fonction du module Dynamic Time Warping (qu'on abrègera par DTW) est d'identifier les mouvements effectués par l'utilisateur en se basant sur les mouvements préenregistrés avec l'approche probabiliste nécessaire puisque tous les utilisateurs ne font pas le même mouvement de la même manière et avec la même précision.

5.2.5 Détection du rythme et du tempo

La détection du tempo se fera à partir des mouvements des pieds et celle du rythme à partir des mains.

5.2.6 Détection de richesse de la danse

Ce bloc évalue la variation de position des membres du danseur. Cette « richesse » de la danse sera représentée par un curseur, plus la variation est élevée plus la musique contient des sons élémentaires différents.

5.2.7 Séquenceur musical

Le séquenceur musical détermine les sons à placer dans la musique générée à partir des informations sur la dans qui lui parviennent, des sons déjà utilisés dans la musique, et des exigences de l'utilisateur. Il exprime un besoin au bloc de choix des sons (par exemple « il me faut une ligne de basses profonde et électronique »).

5.2.8 Synthèse audio

Ce bloc construit la musique à partir des boucles et sons choisis par le séquenceur. Il réalise la lecture, la diffusion et l'enregistrement du son. Nous utiliserons la bibliothèque **java.sound** pour la manipulation des sons et **java.io** pour la manipulation de fichiers.

5.2.9 Application Android

L'application Android permet à l'utilisateur de communiquer le volume et style musical souhaités, et de parcourir les anciennes musiques précédemment créées.

5.2.10 Serveur

Le serveur sert de passerelle entre l'application mobile et le système. Il met à disposition les sons et les images sauvegardées et permet d'effectuer des réglages du son. Nous utiliserons la bibliothèque **java.net** pour la gestion des sockets.

5.3 Description des interfaces

5.3.1 InterfaceBloc Utilisateur – Kinect

La Kinect filme et mesure les mouvements de l'utilisateur grâce à une caméra RGB classique et deux capteurs de profondeur IR.

5.3.2 InterfaceBloc Kinect - Module Kinect

Ces deux blocs communiquent grâce à l'API Kinect et à la biblitohèque J4KSDK. La Kinect transmet des informations à son API qui exploite et met en forme les squelette, des informations de profondeur, des flux videos et audio, etc… La bibliothèque J4K interface l'API Kinect et le code Java.

5.3.3 InterfaceBloc Client Java – Détection des mouvements

Ces blocs échange un objet « squelette » contenant la dernière localisation spatiale de l'ensemble des articulations du squelette de(s) (l')utilisateur(s) dans le référentiel de la Kinect. Cet objet contient également le nombre de danseurs actuellement détectés.

5.3.4 InterfaceBloc Client Java – Détection du rythme

Ces blocs échange un objet « squelette » contenant la dernière localisation spatiale de l'ensemble des articulations du squelette de(s) (l')utilisateur(s) dans le référentiel de la Kinect. Cet objet contient également le nombre de danseurs actuellement détectés.

5.3.5 InterfaceBloc Client Java – Détection richesse de la danse

Ces blocs échange un objet « squelette » contenant la dernière localisation spatiale de l'ensemble des articulations du squelette de(s) (l')utilisateur(s) dans le référentiel de la Kinect. Cet objet contient également le nombre de danseurs actuellement détectés.

5.3.6 InterfaceBloc Détection des mouvements – BDD

Le bloc « Détection des mouvements » interroge la table « mouvements » de la base de données afin d'identifier le mouvement actuellement réalisé par l'utilisateur. L'interrogation de la base de données d'effectue par requêtes SQL, en réponse, le serveur de BDD renvoie l'ensemble des mouvements probable ainsi que leurs attributs (quelle partie du corps, amplitude du mouvement, saccades, type de danse, etc…)

5.3.7 InterfaceBloc Détection des mouvements – Choix des sons

Ces blocs s'échangent des informations sur les attributs des mouvements détectés. Ces attributs prennent la forme de « curseurs » (un pourcentage) évaluant, par exemple, l'utilisation du bras gauche, la fluidité de la danse, si le mouvement est plutôt vertical, etc… Le bloc « Detection des mouvements » en détermine les valeurs à partir des attributs préenregistrés dans la BDD mouvement, pondérés par la probabilité de détection dudit mouvement. Tous ces curseurs sont stockés dans un objet « mouvements » mis à jour à une fréquence proche de 20Hz et récupéré par le bloc « Choix des sons ».

5.3.8 InterfaceBloc Détection des mouvements – Séquenceur musical

Le bloc « Détection des mouvements » fournit deux types d'informations au séquenceur musical. D'une part il permet de synchroniser la danse et le son en indiquant les « temps forts de la danse » (typiquement un saut, particulièrement dans le cas d'une danse saccadée). Cette synchronisation se fait par l'appel d'une méthode du bloc « Séquenceur musical ».

En plus de la synchronisation, le bloc « Détection des mouvements » fournit une estimation des changements de mouvement, permettant au séquenceur de passer à une autre phase musicale.

5.3.9 InterfaceBloc Choix des sons – BDD

Le bloc « Choix des sons » interroge la table « sons » de la base de données afin de filtrer les sons correspondant à la danse et à la demande du séquenceur musical. L'interrogation de la base de données d'effectue par requêtes SQL, en réponse, le serveur de BDD renvoie l'ensemble des sons utilisables, leurs caractéristiques et leur chemin d'accès.

5.3.10 InterfaceBloc Séquenceur musical – Synthèse audio

Le bloc « Séquenceur musical » fournit le chemin d'accès des sons à jouer ainsi que les informations permettant le positionnement temporel (dans des boucles) de ces sons. Concrètement, chaque boucle prend la forme d'un objet, avec comme attributs le chemin du son, la fréquence de ceux-ci au sein d'une mesure, etc…

5.3.11 InterfaceBloc Détection du rythme et du tempo – Séquenceur musical/Synthèse audio

Le bloc « Détection du rythme et du tempo » fournit aux deux blocs audio les informations détectées dans la danse. Concrètement, un objet « tempo » est créé et est mis à jour en temps réel. En plus du rythme et du tempo, il contient une indication sur leurs variations afin de détecter des changements de phases musicales.

5.3.12 InterfaceBloc Détection de la richesse de la danse – Séquenceur musical

Le bloc « Détection de la richesse de la danse » fournit une estimation de la richesse de la danse sous la forme d'un « curseur » (entre 0 et 100) encapsulé dans un objet « richesse ». Cette information permet au séquenceur musical de choisir combien de sons jouer en même temps.

5.3.13 InterfaceBloc Fichier sons – Synthèse audio

Le bloc « Synthèse audio » récupère les fichiers sons élémentaires à jouer à partir de leur chemin d'accès récupéré dans la BDD.

5.3.14 InterfaceBloc Synthèse audio – Haut-parleurs

Le bloc « Synthèse audio » utilise la carte son du PC pour restituer en direct la musique générée sur des haut-parleurs.

5.3.15 InterfaceBloc Utilisateur – Application Androïd

Via l'application Androïd, l'utilisateur peut régler à l'avance le style musical qu'il souhaite, le volume, etc… Il peut également récupérer et jouer un musique qu'il a préalablement enregistré grâce à Fituring.

5.3.16 InterfaceBloc Application Androïd – Serveur

L'application Androïd communique avec le serveur via le protocole IP sur une liaison WiFi AdHoc. Les échanges sont bidirectionnels.

5.3.17 InterfaceBloc API Kinect – Serveur

Dès le début de la danse, le serveur enregistre l'image de la caméra RGB de la Kinect afin de pouvoir la superposer à la musique créée lors de la restitution.

5.3.18 InterfaceBloc Serveur – Séquenceur musical

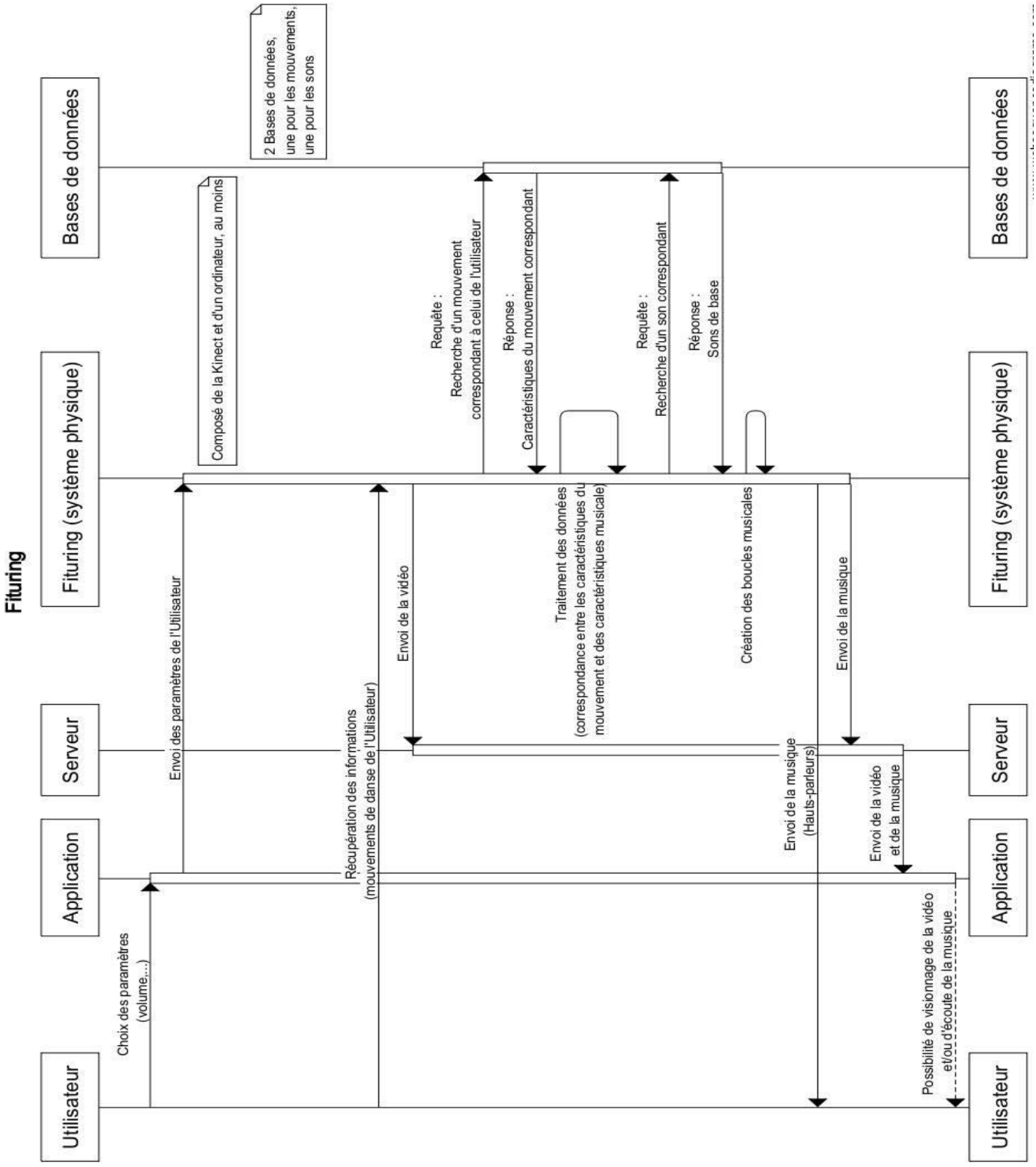
Conformément aux souhaits de l'utilisateur, le serveur fournit au séquenceur musical le style de musique à générer. Cela se fait par l'appel d'une méthode de l'objet « musique ».

5.3.19 InterfaceBloc Serveur – Synthèse audio

Conformément aux souhaits de l'utilisateur, le serveur règle le volume de la restitution audio par l'appel d'une méthode. A la fin de la musque, le bloc « synthèse audio » fournit au serveur le fichier audio généré pour pouvoir le réécouter.

5.4 Diagramme de séquence

Ce diagramme décrit les interactions entre les différentes composantes du système lors d'un cycle normal d'utilisation :



5.5 Interface utilisateur graphique

Ces schémas correspondent aux attentes graphiques de l'application android avec les transitions entre ses différentes activités.

5.6 Tableau détaillé des tâches

Tâche	Description	Démontrée au PAN3	Intégrée au PAN4

T1	**Lecture Audio (Jamal.Skouri , Alexandre Poinso)**		
T1.1	Programme qui lit un fichier son	X	X
T1.2	Programme qui joue un fichier son en boucle et en rythme	X	X
T1.3	Programme qui enregistre la musique dans un fichier	X	
T1.4	Comparaison des caractéristiques		
T2	**Détection de rythme (Stan Hannebelle, Khadija El Attar)**		
T2.1	Découpage signal sonore en bande fréquentielle	X	X
T2.2	Détection du tempo	X	
T2.3	Changement de tempo		
T3	**Kinect Squelette (Thibaud Lemaire, Robin Shin)**		
T3.1	Application test de suivi de l'ensemble du corps	X	X
T3.2	~~Définition du format d'échange des données entre C et Java~~	X	X
T3.3	~~Application client-serveur entre Java et C pour l'échange de données via sockets TCP~~	X	X
T3.4	Suivi du corps complet intégré dans le prototype allégé	X	
T3.5	Caractérisation de l'utilisation de la Kinect pour notre projet (Bilan)	X	
T4	**Androïd (Camille Demasson , Khadija El Attar)**		
T4.1	Réaliser une application spécifique (à définir en fonction des besoins du projet)	X	X
T4.2	la communication via le réseau, par exemple selon le protocole HTTP.	X	X
T4.3	Décrire le fonctionnement et l'intégration du module dans le prototype allégé	X	X
T4.4	Analyser comment le module est intégré dans le prototype	X	
T5	**Intégration & test (Thibault Lemaire, Camille Demasson)**		
T5.1	Rapport sur l'état de l'intégration des modules	X	X
T5.2	Description des interfaces java pour chaque	X	X

	module		
T5.3	Résultat des tests	X	X

Tableau 1 - Description des tâches

6 Organisation du projet

6.1 Diagramme de planification temporel des tâches

Cf. annexe

6.2 Répartition des élèves par module

Nom Module	Module Kinect	Module DTW	Module Détection du rythme	Module Lecture audio	Module Androïd	Module SES	Test & Intégration
Nom Expert	Jean Lefeuvre	Michel Roux	Bertrand David	Jean-Claude Dufourd	Jean-Claude Dufourd	Julien Morel	Soumia Dermouche
Eleve1	Thibaud Lemaire	Alexandre Poinso	Khadija El Attar	Jamal Skouri	Camille Demasson	Jamal Skouri	Thibaud Lemaire
Eleve2	Robin Shin	Karim Khandid	Stan Hannebelle	Alexandre Poinso	Khadija El Attar	Karim Khandid	Camille Demasson
Eleve3		Robin Shin				Stan Hannebelle	

Tableau 2 - Répartition élèves-modules

6.3 Plans de test

Voici les tests par module que nous prévoyons :

Le détails de ces tests est disponible en annexe, sous le nom “Tests programmés”

Module Kinect :

La kinect détecte bien les mouvements.
Test compatibilité avec plusieurs utilisateurs (pas trop de mélange de points des squelettes).

Module android :

L'application réalise correctement chacune de ses activités.
Création de la vidéo (image + son) grâce aux images et sons contenus dans la base de données du serveur.
La connexion avec le serveur se fait bien : les paramètres rentrés dans le smartphone sont bien communiqués à l'ordinateur de Fituring ; le smartphone reçoit les données de l'ordinateur et de la kinect par le serveur.

Module classification :

Vérifier qu'on a un tableau de points cohérent en entrée par rapport aux mouvements (cas pour chaque membre du corps, cas s'il y a plusieurs personnes)
Vérifier qu'on reconnaît un mouvement donné à partir du tableau de points.

Vérifier que le son choisi après analyse du mouvement est adapté à celui-ci.
Reconnaître la fin d'une danse pour entraîner la fin du morceau.

Module détection du rythme :

Tester si le programme détecte le tempo et le rythme.
Tester si l'on a bien associé une musique riche à une danse riche.

Module synthèse audio / séquenceur :

Vérifier qu'on ait bien la musique en sortie.
vérifier qu'on aie les bons fichiers son en entrée.
Les boucles musicales s'enchaînent bien, il n'y a pas de coupures dans la musique.
Le séquenceur peut enregistrer la musique générée.
La musique générée peut être lue en temps réel.

Nous avons créé un modèle à compléter pour chaque test que nous souhaitons réaliser pour les différents modules concernés :

- Entrée
- Sorties
- Support et conditions du test
- Procédure
- Résultats attendus
- Tolérance

6.4 Tâches

Module	Avancement Prévu (%)	Avancement Réel (%)	Temps passé (h)	Description brève du travail effectué, analyse des écarts constatés
Kinect	50% 80%	40% 80%	10 6	Nous avons modifié l'architecture du projet. Nous utilisons une bibliothèque JAVA qui interface la Kinect. Le code d'acquisition des données compile, mais nous n'avons pas pu le tester sur PC pour un problème de DLL manquante. Ce module est quasiment terminé. Il ne nous reste qu'a implémenter un enregistreur / lecteur de session Kinect qui servira pour les tests.
Android	20% 50%	10% 40%	10	L'application tourne sur un téléphone. L'application peut lire un contenu media, communiquer avec un

				ordinateur grâce au réseau.
Détection de rythme	20% 60%	10% 50%	8	Une première version de l'algorithme a été implémentée en Java - un rapport a été transmis à l'expert - 3 réunions ont eu lieu Elaboration de deux programmes java et d'une méthode réussie en Matlab,on est arrivé à obtenir le BPM de la danse à partir du mouvement des mains ou pieds
Classification par DTW	40% 75 %	20% 60 %	4 10	Nous avons été prévenus assez recemment que la classification par kPPV n'était pas adaptée pour notre projet. Nous avons donc opté pour la classification par DTW, et donc changé d'expert ainsi que la fiche module. L'algorithme DTW est implémenté et fonctionnel. Il a été testé sur des exemples simples , sans lien avec notre projet ; ces tests sont concluants.
SES	?? %	0%	1h	Nous sommes en attente d'une réponse de l'expert
Lecture Audio	30% 70 %	20% 50 %	3h 8h	En cours d'assimilation de la documentation. Les trois livrables demandés ont été codés, puis testés : ces tests sont concluants.

Tableau 3 - Avancement des modules

7 Bibliographie

7.1 Module Android

[1] https://openclassrooms.com/courses/creez-des-applications-pour-android
[2] http://developer.android.com/index.html
[3] https://openclassrooms.com/forum/sujet/comment-passer-d-une-activite-a-l-autre

7.2 Module détection de rythme

[4] http://introcs.cs.princeton.edu/java/97data/FFT.java.html
[5] http://introcs.cs.princeton.edu/java/97data/Complex.java.html
[6] http://www.developpez.net/forums/d552298/environnements-developpement/matlab/signal/enveloppe-d-signal-type-sinusoidal-amorti/
[7] http://www.ee.columbia.edu/~dpwe/ismir2004/CRFILES/paper191.pdf
[8] http://www.ee.columbia.edu/~dpwe/papers/Schei98-beats.pdf
[9] https://bedavid.wp.mines-telecom.fr/enseignement/atiam/fpa-signal/

7.3 Module DTW

[10] https://en.wikipedia.org/wiki/Dynamic_time_warping
[11] https://fr.wikipedia.org/wiki/Distance_de_Levenshtein
[12] https://fr.wikipedia.org/wiki/Distance_(math%C3%A9matiques)
[13] https://fr.wikipedia.org/wiki/Algorithme_de_Wagner-Fischer

7.4 Module Lecture Audio

[14] https://docs.oracle.com/javase/7/docs/api/javax/sound/sampled/package-summary.html
[15] http://www.labbookpages.co.uk/audio/javaWavFiles.html
[16] http://www.dynamic-mess.com/java/lire-un-son-en-java-8-21/

[17] http://www.blogduprogrammeur.com/lire-fichier-audio-java/
[18] http://stackoverflow.com/questions/3297749/java-reading-manipulating-and-writing-wav-files
[19] http://www.developpez.net/forums/d750988/java/general-java/debuter/lire-wav/
[20] http://www.fobec.com/java/1043/jouer-son-musique-format-wav.html
[21] http://www.java-tips.org/java-se-tips-100019/120-javax-sound/917-capturing-audio-with-java-sound-api.html
[22] http://www.codejava.net/coding/capture-and-record-sound-into-wav-file-with-java-sound-api
[23] http://docs.oracle.com/javase/tutorial/sound/capturing.html

7.5 Module Kinect

[24] https://www.microsoft.com/france/msdn/sdk-kinect/tutoriel.aspx

[25] http://www.developpez.com/actu/34969/5-minutes-pour-comprendre-le-developpement-avec-Kinect-Microsoft-publie-un-tuto-et-des-demos-pour-prendre-en-main-son-capteur-de-mouvements/
[26] http://research.dwi.ufl.edu/ufdw/j4k/J4KSDK.php
[27] https://en.wikipedia.org/wiki/Relative_luminance

7.6 Module Intégration et test

[28] http://rom.developpez.com/java-listeners/

Annexe A. Lexique

- **DLL :** Dynamic Link Library
- **DTW :** Dynamic Time Warping
- **kPPV :** k Plus Proches Voisins
- **BDD :** Base De Données
- **FFT** : Fast Fourier Transform (Algorithme de transformation de Fourier discrète)
- **SDK :** Software Development Kit

Annexe B. Modifications

A compléter à partir du PAN2

B.1. Modifications de fond

Tableau des modifications de fond apportées au projet avec validation des experts et encadrant informatique

libellé / date	Description brève	Validé par :
26/12/2015	Plus de code en C, utilisation d'une bibliothèque native en Java	Jean Le Feuvre
15/01/2016	Modification du module Classification : ce n'est plus par kPPV mais par DTW	Michel Roux
18/01/2016	Modification répartition module	Experts
10/03/2016	Ajout d'un lexique Ajout de l'avancée des modules au PAN3	

B.2. Modifications du rapport

Vous noterez dans cette section les modifications apportées au rapport depuis le PAN précédent. Si votre planification temporelle a été modifiée, vous laisserez l'ancienne planification dans cette annexe.

B.2.1. Modifications du rapport au PAN2

Pour le PAN2, nous avons effectué quelques changements mineurs dans l'architecture du projet :

D'une part, le module Kinect sera codé intégralement en Java alors que nous devions initialement réaliser une interface en C avec communication par sockets avec le code en Java.

D'autre part, le module classification devient module DTW. Ce module sera finalement encadré par Michel Roux alors que la fiche module initiale prévoyait que Slim Essid l'encadre.

B.2.2. Modifications du rapport au PAN3

Nous avons tout d'abord amélioré la mise en forme du rapport en ajoutant la numérotation des pages et en ajoutant un lexique.

Nous avons indiqué l'avancement de chaque module en annexe. La description des procédures de test a été modifiée et les résultats de certains tests ont été consignés.

Sur l'architecture du projet, il n'y a pas de changement majeur. La partie BDD est toujours en suspens. Peut être que les différents mouvements et musiques disponible seront stockés dans des bibliothèques d'objets serializables.

B.2.3. Modifications du rapport au PAN4

Annexe C. Comptes Rendus de réunions

C.1. CR du 5/10/2015

Stan : Projet langue des signes

Bonne idée car utile et très ambitieux : beaucoup de mouvements, différences subtiles

Peut-être qu'on n'a pas le niveau… Coté synthèse de la voix intéressant. On peut dépasser facilement les 8 modules.

Grosse difficulté du sujet : détection des mouvements → Comment ? A distance ou avec un équipement ?

Grosse partie de programmation : traitement des données, synthèse de la voix, etc…

Question de Camille : la difficulté ne vient-elle pas plutôt de la connaissance du langage des signes ? Personne ne connaît cette langue parmi nous.

Khadija explique comment elle envisage la détection des mouvements par « pixellisation »

Robin pense que c'est impossible à faire à notre niveau

Thibaud : Le jeu des signes/musique de Camille

Ludique est très modulaire, on peut à peu près tout imaginer du moment que c'est un jeu

Robin : Skate streetview

Skate statique qui se déplace sur streetview grâce à l'inclinaison du skate projet PACT déjà réalisé avec un vélo

Problème de la fluidité des images

Thibaud trouve ça trop facile. Camille trouve que la finalité et l'utilité du projet n'est pas super.

Jamal : Invention d'un instrument qu'on n'a pas besoin de toucher

C'est intéressant car d'habitude musique → danse alors que là c'est inversé

C'est difficile car il faut des gens qui maitrise la danse et la musique : il faut que ce soit cohérent entre danse et musique

Peut être réalisable avec des lasers qui sortent du sol

Stan pense qu'on peut mettre en place des palettes pour choisir un style musical afin de limiter le nombre et la précision des mouvements

Robin n'est pas emballé

Karim : Pas d'idée à défendre

Alexandre : Langage des signes

Camille : Lunettes GPS

Afficher le trajet sur les lunettes, ambitieux

Relier les lunettes à un Smartphone pour permettre de calculer le trajet

Difficulté : positionnement du trajet sur les lunettes

Khadija : Maillot de sport qui capte les mouvements

Mesure de l'intensité de l'effort du sportif

Capture de geste sportif précis pour les optimiser (proche des technologies de film 3D)

Existe déjà

Amélioration possible : augmenter le nombre de puce (pour quoi faire ?)

Difficulté du minimiser la taille de l'outil

Jamal : Chargeur de batterie par l'effort physique

Utile mais pas trop de rapport avec PACT

3 Projets Retenus :

- Langue des signes
- Corps instrument / Jeu mouvement-son
- Stylo réseau social permettant d'envoyer des dessins aux autres utilisateurs
-

C.2. CR du 29/09/2015

Membres du groupe

Le groupe 3.1 est composé des étudiants suivants : Jamal SKOURI, Robin SHIN, Alexandre POINSO, Thibaud LEMAIRE, Karim KHANDID, Stanislas HANNEBELLE, Khadija EL ATTAR SOFI et Camille DEMASSON. Seuls Karim et Jamal n'ont pas pu être présents.

Découverte des trois thèmes PACT de cette année

Nous avons la possibilité de réaliser notre projet PACT en respectant l'un des trois thèmes suivants :

- Du geste au son ;
- "L'invasion" virtuelle de la ville ;
- Le projet inédit.

Premières propositions de sujets

Chacun des membres présents de notre groupe a énoncé ses premières idées :

Robin :

- Skate connecté à street View permettant de se déplacer dans les rues virtuelles grâce à l'inclinaison et l'orientation du skate (1).

Alexandre :

- Voiture miniaturisée commandée par les gestes (2) ;
- Maillot sportif équipé de capteurs permettant de suivre les mouvements (rythme cardiaque, etc.) du sportif qui le porte (3).

Thibaud :

- Appareil (type mp3) qui adapte la musique écoutée selon son rythme par rapport à l'intensité d'un effort physique (4) ;
- Invention d'un instrument gyroscopique : le son produit dépend de l'orientation de l'instrument (5) ;
- Livraison de colis sur chariots qui s'orientent eux-mêmes grâce aux données cartographiques en ligne (6).

Stanislas :

- Tapis roulant entouré de la projection de décors (sur des écrans ou dans des lunettes) venant des données en ligne et permettant de s'orienter dans ces décors (7) ;
- Appareil qui règle à distance le style de musique à écouter selon les gestes effectués par l'utilisateur (8).

Khadija :

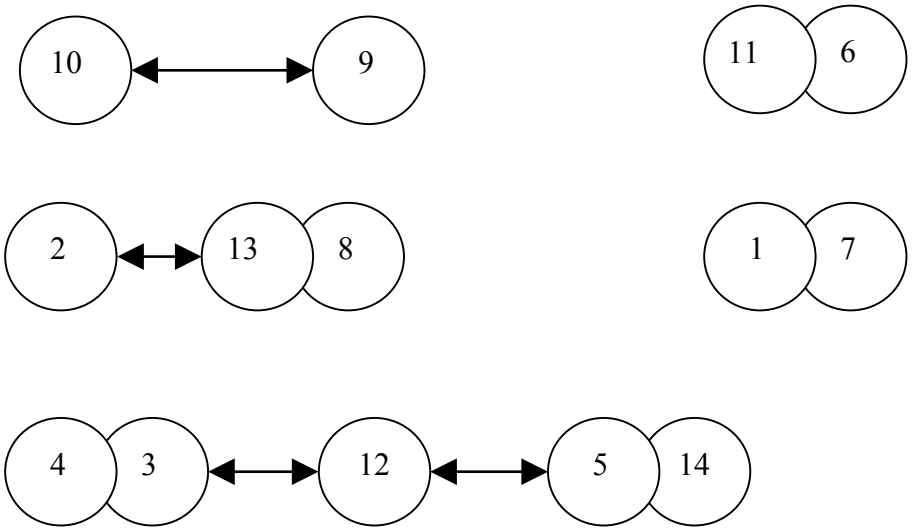
- Traduction de la langue des signes en paroles (idée de solution : capter et pixeliser le champ autour de la personne qui fait les gestes) (9) ;
- Traduction en paroles/son des besoins des plantes et des animaux en fonctions de leurs gestes (10) ;
- Lunettes GPS : lunettes portées pendant un trajet qui permet d'afficher l'itinéraire à suivre directement sur la route devant l'utilisateur (11).

Camille :

- Jeu : reproduire certains gestes particuliers pour jouer correctement une musique (12) ;
- Enceinte qui répond aux gestes (13) ;
- Invention d'un instrument que l'on n'a pas besoin de toucher : traduction d'une danse en musique (14).

Organisation des sujets proposés

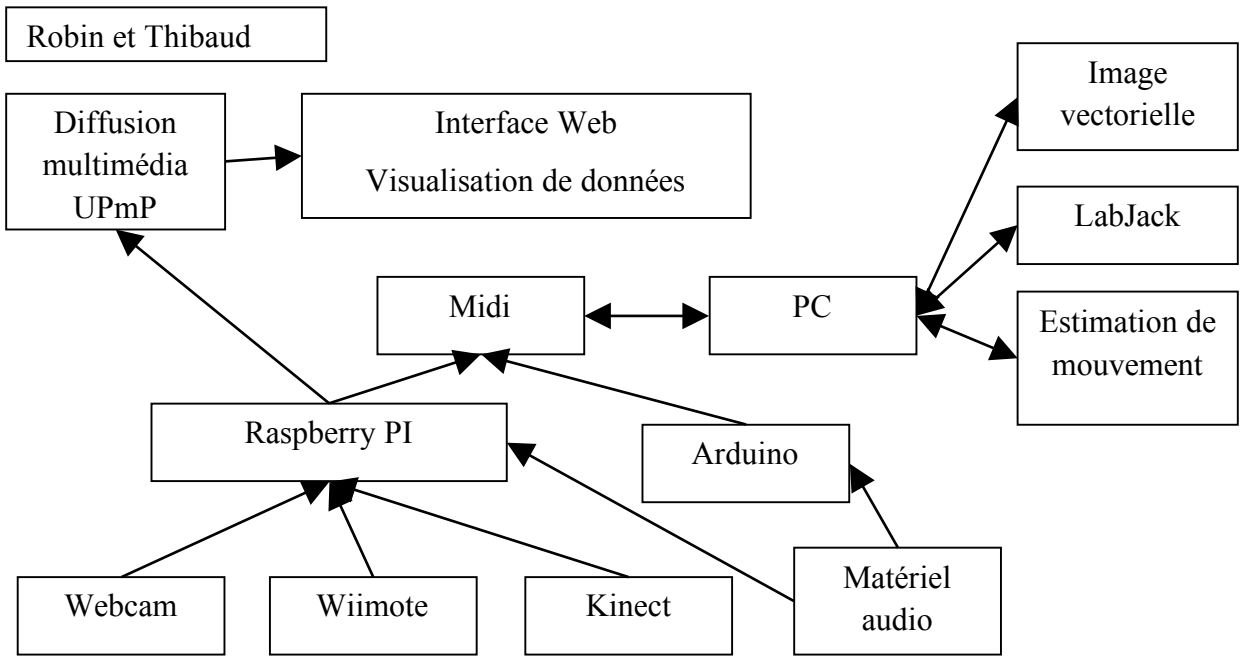
Après avoir mis en commun toutes nos idées, nous les avons regroupées par proximité de la façon suivante :

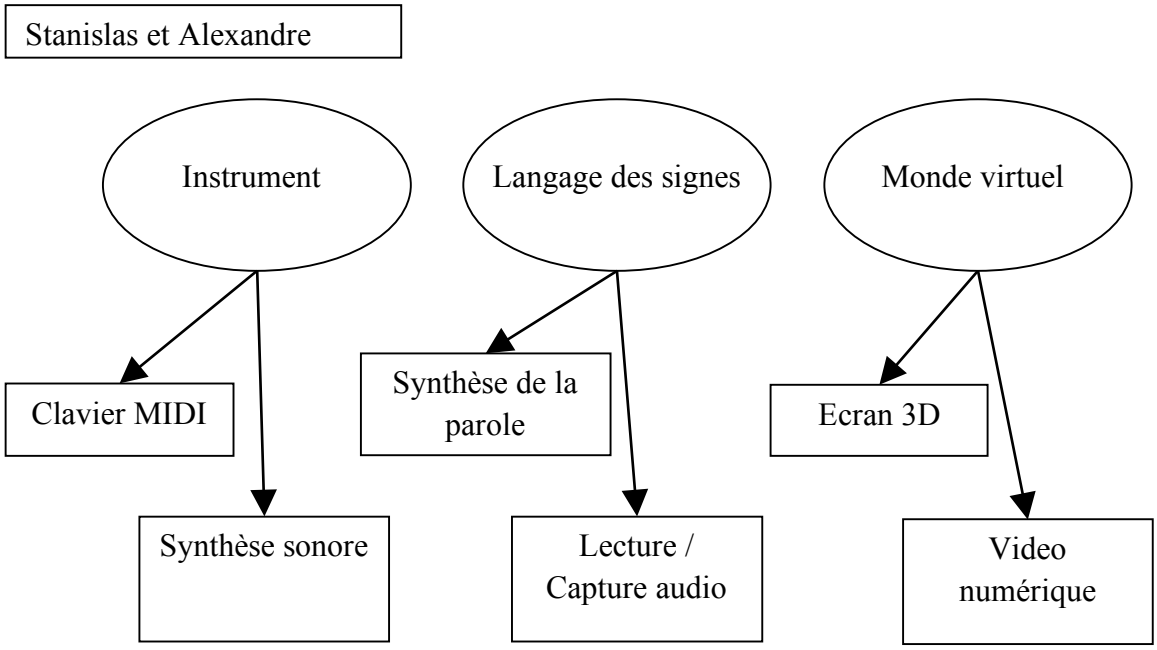


Répartition de technologies connues

Un ensemble de cartes décrivant des méthodes et des technologies nous ont été distribuées. Parmi celles que nous connaissions, nous avons dû trier les cartes correspondantes selon la proximité des domaines d'application de ces techniques, afin de les rapprocher des problèmes technologiques représentés par le projet PACT que nous allons devoir traiter.

Trois regroupements différents ont donc été réalisés :



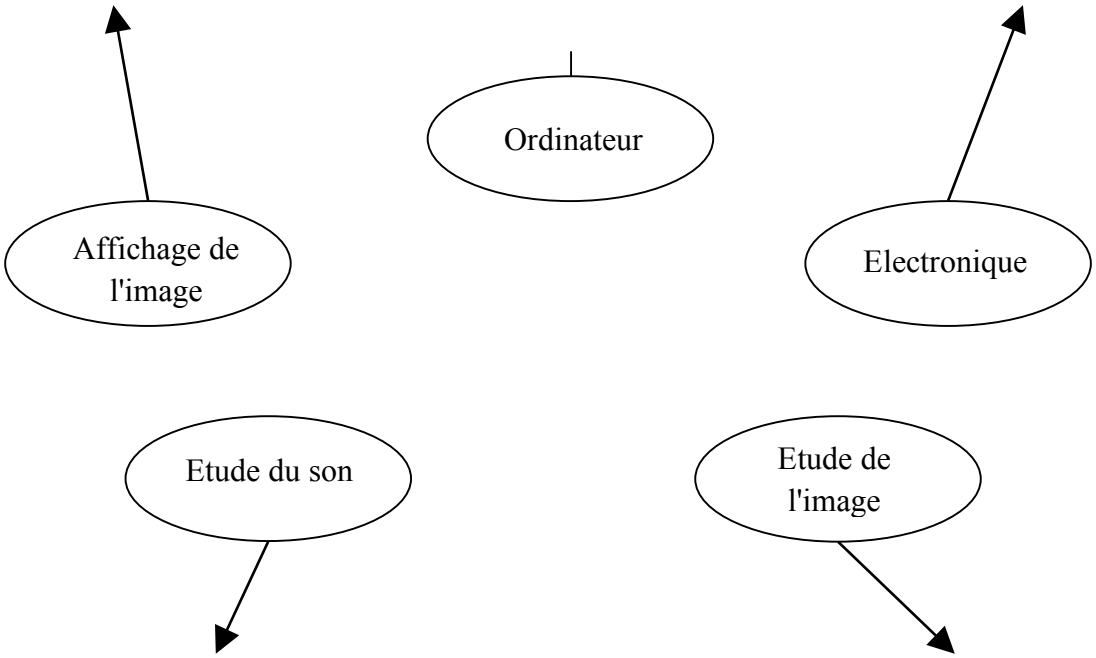

Stanislas et Alexandre
Instrument
Langage des signes
Monde virtuel
Clavier MIDI
Synthèse de la parole
Ecran 3D
Synthèse sonore
Lecture / Capture audio
Video numérique

Khadija et Camille

Interface graphique
Projecteurs et pico-projecteurs
Smartphone Tablette
Lunettes 3D
Programmation concurrente
Base de données
Applications Distribuées

Banc de filtres
Conception électronique

Synthèse d'images 3D
Vidéo 3D
Formats d'images



Estimation de hauteur / périodicité
Reconnaissance vocale

Annexe D. Module Kinect

Module Kinect Squelette – Groupe 3.1

Encadrant: Jean Le Feuvre (jean.lefeuvre@telecom-paristech.fr, Poste 7169)

réalisé par : Robin SHIN et Thibaud LEMAIRE

Volume estimé : 20 TH PACT + 10 TH perso – complexité normale

Descriptif

L'objectif de ce module est l'utilisation de la Kinect sous Windows en Java. Le module exploite la bibliothèque J4KSDK, qui permet d'exploiter la Kinect directement en Java

Ressources dont le module dépend

Une machine récente sous Windows 7 ou 10 (PAS DE MACHINES VIRTUELLES)
SDK Kinect pour Windows (site),
La documentation Microsoft (site)
La description du fonctionnement de l'algorithme de suivi du corps (lien)

Objectifs d'apprentissage

- Principes de bases du fonctionnement de la Kinect
- Modélisation par squelette: notion de joint, de pose, d'invariance. Principes du fonctionnement de l'algorithme Microsoft.

Résultats attendus

PAN1
- expliquez simplement la Kinect: capteurs utilisés, données produites
- Quelles données sont disponibles pour le suivi de corps ? Comment ces données sont exploitées par votre prototype ?
- Avoir installé les logiciels sur une machine Windows.

PAN2
- Application test de suivi de l'ensemble du corps (les données du squelette peuvent être consultées via console ou dans un fichier) – ***code C#1***
- Application qui récupère le squelette via la bibliothèque J4K
- Documentation de la bibliothèque et de son installation

PAN3
- Possibilité de rejouer une séquence enregistrée
- ***Documentation***
- Caractérisation de l'utilisation de la Kinect pour votre projet : les mouvements/gestes sont-ils tous détectable comme prévu ? Quelles sont les conditions limites de détection dans le cadre de votre application (luminosité, distance au capteurs, types de gestes) ? Quels

problèmes avez-vous pu contourner ? Quelles seraient vos pistes pour les autres ?

Annexe E. Module DTW

Module DTW – Groupe 3.1

Encadrant: Michel Roux (michel.roux@telecom-paristech.fr)

Réalisé par : Alexandre Poinso, Robin Shin et Karim Khandid

Volume estimé : ??? TH PACT + ? TH perso – complexité ????????????

Descriptif

L'objectif de ce module est de déterminer par une approche probabiliste les mouvements effectués par l'utilisateur. Le module vise à comparer les données provenant de la caméra Kinect (les points du squelette) à des mouvements préenregistrés dans une base de données.

Ressources dont le module dépend

- Aucune dépendance a priori mais possibilité d'exploiter une librairie JAVA (à choisir avec l'expert) pour développer le code de la classification.
- Définition mathématique d'une mesure, d'une distance, d'une similarité
- Une base de données (vectorielles) définie dans le projet du groupe

Objectifs d'apprentissage

- Notion en apprentissage automatique : distances, similarités, apprentissage, évaluation, validation croisée
- Compétence à développer : à l'issue de ce module vous aurez défini et programmé un classificateur kPPV en Java et vous saurez l'évaluer sur une base de données

Résultats attendus

PAN1 :

- **Savoir définir** les termes suivants : une distance, une similarité, apprentissage automatique, classification.
- **Savoir expliquer** en quoi cet outil de classification est a priori intéressant dans le projet du groupe.

PAN2 :

- *Livrable : **Pseudo code** implémentant la classification par DTW, en Java.*

PAN3 :

- *Livrable : **code Java et premières mesures** de performance sur des données standards (par forcément en lien avec votre projet).*

PAN4 :

- *Livrable : **rapport** de performance, en situation d'usage.*

Annexe F. Détection du rythme

Module Détection du rythme – Groupe 3.1

Encadrant: Bertrand David (bertrand.david@telecom-paristech.fr)

Réalisé par : Stan Hannebelle, Khadija El attar

Volume estimé : 20 TH PACT + 10 TH perso – complexité normale

Descriptif

L'objectif de ce module est de détecter le rythme et le tempo (à partir des pas) dans la danse de l'utilisateur ainsi que la richesse de sa danse reliée aux variations des membres de l'utilisateur tel que plus la danse est variée plus la musique contient des sons élémentaires différents.

Ressources dont le module dépend

- wikipédia : filterbank, spectrogram, filter, cours oasis (SI101) filtrage
- bibliothèques JAVA à mettre en oeuvre : math common api
- compléments : un tp de 2007 en signal, détecteur de mélodie (contient de la détection d'enveloppe et de la détection de hauteur notamment).

biblio et slides : sur cette page (intranet)

Objectifs d'apprentissage

- Maitriser les principes de filtrage, transformée de Fourrier, échantillonnage
- Savoir expliquer le principe de fonctionnement d'un détecteur d'attaque
- Savoir choisir les filtres (longueur, type)

Être capable de mettre un oeuvre un filtre numérique simple

Résultats attendus

- PAN 2 :
  -Notice descriptive précise : choix de paramètres permettant la détection avec justifications des choix, description mathématique et description informatique)
  -Élaboration du pseudo-code à mettre en oeuvre ou première version de code java
- PAN3 :
  - code java commenté et structuré de manière à être lisible.
  - tests de la fonction d'attaque :
    - sur un signal avec des sons bien séparé de batterie
    - sur des signaux plus complexes
- PAN 4 (éventuel) :

- pour des signaux plus complexes : rajouter un banc de filtre (module banc de filtre)
- calculer une fonction de détection dans chaque bande et fusionner l'ensemble (somme pondérée des fonctions dans chaque bande)

tester les améliorations par rapport au modèle plus simple précédent sur des cas plus complexes que vous définirez vous même.

Annexe G. Module Synthèse audio

Module Lecture, capture et traitement de fichiers audio – Groupe 3.1

Encadrant: Jean-Claude Dufourd (jean-claude.dufourd@telecom-paristech.fr)

Réalisé par : Khadija El Attar, Stan Hannebelle, Jamal Skouri

Volume estimé : ???? TH PACT + ???? TH perso – complexité ??????????

Descriptif

Ce module permet la lecture des fichiers audio en fonction des ordres du séquenceur musical. Le module génère et lit des boucles audio, gère le réglage du volume, et permet l'enregistrement de la musique générée dans un fichier

Ressources dont le module dépend

Bibliographie:

- http://www.labbookpages.co.uk/audio/javaWavFiles.html
- http://commons.apache.org/math/ (la classe à utiliser pour la transformée de Fourier est org.apache.commons.math3.transform.FastFourierTransformer)
- https://code.google.com/p/jmathplot/ ou http://www.jfree.org/jfreechart/
- http://docs.oracle.com/javase/tutorial/sound/capturing.html

Objectifs d'apprentissage

- Principe de la lecture audio en Java
- Création de fichiers sons
- Calage des boucles en rythme

Résultats attendus

- Connaissances : maitrise du son en version informatique, pour des traitements courants
- Compétence : *à l'issue de ce module vous aurez écrit en Java un programme qui lit du son dans un fichier et le joue, capture du son et le stocke dans un fichier, calcule un son synthétique et le joue, visualise un son et son spectre.*

Livrable 1: programme qui lit un fichier son, le joue en boucle et en rythme

Livrable 2: programme qui enregistre dans un fichier

Annexe H. Module Androïd

Module Androïd – Groupe 3.1

Encadrant: Jean-Claude Dufourd (poste 7733), bureau E509, jean-claude.dufourd@telecom-paristech.fr

réalisé par : Camille Demasson ,Khadija El Attar

Volume estimé : TH PACT + TH perso – complexité

Annexe I.

Descriptif

L'utilisation de smartphones ou de tablettes est devenue depuis quelques années presque incontournable. Parmi les différents systèmes d'exploitation permettant de contrôler ces smartphones et tablettes, le système Android représente près de la moitié des périphériques. La programmation des smartphones ou tablettes Android fait appel au langage Java, vu lors du cours INF103. Elle ne devrait pas poser de difficulté majeure, néanmoins une des difficultés souvent rencontrées est la prise en main de l'environnement de développement et des spécificités des applications Android. Ce module a pour vocation d'aider les élèves à surmonter cette difficulté.

Annexe J.

Ressources dont le module dépend

Le site de référence pour l'environnement de développement Android (bibliothèques, simulateur, environnement de développement, etc.) et pour l'auto-formation est le site http://developer.android.com.

Transparents du mini-cours:

PACT-2014-Android-Mini-cours

Une bonne introduction (en français) se trouve sur:

http://openclassrooms.com/courses/creez-des-applications-pour-android

Annexe K.

Objectifs d'apprentissage

Informatique : prendre en main l'environnement de développement Android, comprendre le

fonctionnement d'une application Android, savoir lire et utiliser la documentation des API Android, savoir créer une application Android de base et utiliser certaines fonctionnalités avancées (affichage graphique, réseau, multimédia …), savoir debugger une application

Annexe L.

Résultats attendus
PAN1 : Démontrer que l'environnement de développement est en place et que des applications simples fonctionnent sur un émulateur Expliquer le cycle de vie d'une application et la notion d'activité PAN2 : Réaliser une application Android sur émulateur mettant en oeuvre le cycle de vie, l'enchaînement de plusieurs activités, y compris en tâche de fond, et le lancement d'une autre application. PAN3 : Réaliser une application spécifique (à définir en fonction des besoins du projet). Cette application fonctionnera sur un smartphone ou une tablette et pourra mettre en oeuvre: la communication via le réseau, par exemple selon le protocole HTTP, un affichage graphique simple avec quelques éléments interactifs, la lecture ou la capture de données multimédia (son, image, vidéo) Décrire le fonctionnement et l'intégration du module dans le prototype allégé PAN4 : Analyser comment le module est intégré dans le prototype, quelles pistes d'améliorations seraient à envisager (performance, simplicité)

Annexe I. Module SES

Module SES (Entretiens semi-directifs , Observations en situation naturelles) – Groupe 3.1

Encadrant: Julien Morel ()

Réalisé par : Jamal Skouri ,Stan Hannebelle ,Karim Khandid

Volume estimé : TH PACT + TH perso – complexité

Descriptif

Les entretiens semi-directifs peuvent être utiles en amont ou en aval du processus de conception. Ils consistent à recueillir la parole d'usagers (ou d'usagers potentiels) dans le but de comprendre leurs pratiques au sens large et surtout le sens qu'ils leur donnent.Ils consistent en une conversation guidée autour de plusieurs thèmes définis en amont, mais dont l'ordre et l'importance sont laissés ouvert. L'objectif des entretiens n'est pas de vérifier des hypothèses, ni de quantifier des résultats comparables, mais de faire émerger et de recueillir des discours (des éléments de vocabulaire, des définitions, des valeurs revendiquées, du sens très prosaïque, etc.)La méthode des entretiens débouche sur des préconisations précises dans le processus de conception. Celles-ci prennent la forme de contraintes ou d'éléments d'inspiration qui portent tant sur la manière de présenter la technologie développée que sur son possible ancrage dans des pratiques effectives.Le module consiste à réaliser et analyser une série d'entretiens. Il s'articule autour de quatre temps :

- élaboration d'un guide d'entretien complet
- choix de la population à interroger
- recueil de données (passation, enregistrement et retranscription des entretiens)
- rédaction d'un document de synthèse dédié au reste du groupe

Son encadrement consiste en une formation rapide aux techniques de l'entretien et à l'élaboration du guide. À la demande des élèves, un retour après chaque entretien est vivement conseillé.

Module Observation :

L'observation directe peut être utile en amont du processus de conception. Elle consiste à accéder, en situation naturelle à des attitudes, des comportements, des gestes, et plus généralement des cours d'action. Plus que des usages au sens strict, elle permet d'analyser des écologies complexes où de très nombreux paramètres sont en jeu. Sa mise en œuvre débouche sur des préconisations précises dans le processus de conception. Celles-ci prennent la forme de contraintes (difficultés à prendre en compte, dimensions à éviter…) ou de points d'inspiration (meilleure compréhension

des contextes d'usages possibles). Le module consiste à réaliser une série d'observations. Il s'articule autour de trois temps :

● élaboration d'une problématique claire et articulée au projet
● recueil de données pertinent et contrôlé
● rédaction d'un document de synthèse dédié au reste du groupe

Son encadrement consiste en une formation rapide aux techniques d'observation et en un suivi par retours réguliers (essentiels à la réussite du module) entre chaque séance d'observation.

L'utilisation de la vidéo (capture écran smartphone, lunettes-caméra, drone volant, multiplexer, etc.) sera encouragée.

Ressources dont le module dépend
Milieu de l'étude

Objectifs d'apprentissage

Résultats attendus
● compétence : *détachement des préjugés (sens commun, évidences…)* ● compétence : *développement des capacités d'écoute* ● compétence : *traitement et analyse de données qualitatives* ● livrable : guide d'entretien ● livrable : entretiens retranscrits (et leur enregistrement) ● livrable : document de synthèse des résultats destiné au groupe livrable : explicitation dans la présentation finale du projet de l'apport de la method ● livrable : document de cadrage (problématique) ● livrable : recueil de données ● livrable : document de synthèse des résultats destiné au groupe livrable : explicitation dans la présentation finale du projet de l'apport de la méthode

Annexe J. Module Intégration et tests

Module Intégration & test– Groupe 3.1

Encadrant: Soumia Dermouche (soumia.dermouche@telecom-paristech.fr)

Réalisé par : Thibaud Lemaire, Camille Demasson

Volume estimé : TH PACT + TH perso – complexité

Descriptif

L'objectif de ce module est de garantir l'interopérabilité des différents modules, de faire en sorte que tout fonctionne bien, et de faire les tests finaux.

Ressources dont le module dépend

Tous les membres du groupe
Tous les modules
Le dépôt GIT

Objectifs d'apprentissage

-Vérifier que chaque module implémente l'interface prévue (par compilation)
-Vérifier que chaque module a été testé (présence de tests dans le module)
-Créer les tests globaux, équivalents des méthodes de tests des modules au niveau du projet entier
-tests de fonctionnalité : ça fait ce qu'il faut
-tests de performance : ça va assez vite
-tests d'utilisabilité : c'est compréhensible par un utilisateur extérieur
…
-Appliquer un par un les cas de tests globaux prévus dans le plan de test:
-Faire des retours aux créateurs des modules sur les problèmes.
-Trouver des solutions pour les problèmes non spécifiques à un module.

Résultats attendus

PAN1
Description textuelle et schéma décrivant l'architecture du système
formalisme libre mais cohérent et légendé
Diagramme d'activité et/ou de séquence (temporel)
Faisant apparaître les acteurs et actions
Et les transitions avec leur(s) condition(s)
Description des interfaces entre modules (texte)
Dépot Git unique
Avec la bonne structure de dossier
Avec un commit par personne dans readme.txt
PAN2
Plan de test global
Liste des tests de fonctionalité, de performance, d'utilisabilité, d'acceptabilité… (tous les types de test be sont pas forcément pertinents pour votre projet)
Pour chaque test
contexte de test

liste des entrées à fournir
liste des sorties attendus, avec la méthode de validation (est-ce qu'on attend exactement ce résultat au bit près, ou dans une certaine marge d'erreur, ou un résultat de cette forme…)
Squelettes de toutes les méthodes principales
sous forme simulée
Fantôme : méthode vide utilisant System.out.println pour indiquer qu'elle est appelée.
Bouchon : méthode renvoyant toujours le même résultat.
Substitut : implémentation de la méthode très simplifiée.

PAN3 :
rapport sur l'état de l'intégration des modules (quel module, quelle version, quels problèmes) dans le proto allégé
résultat des tests (passés, échoués, résultats quantitatifs/qualitatifs sur les performances, l'acceptabilité …) du proto allégé
Note: le rapport et les résultats se feront en utilisant les modules tels qu'implémentés, testés et livrés par les responsables des autres modules à une date à déterminer entre les responsables de modules. Il est acceptable qu'un livrable du module « test et intégration » ne contienne pas la dernière version d'un module (si ce dernier est en retard). Les retards seront cependant à justifier et feront l'objet d'une appréciation par le jury dans la notation de ces modules.
Note: l'encadrant va vérifier sur le dépôt Git que le code est documenté, lisible, maintenable, que tous les élèves ont contribué…

Annexe K. Module classification : livrable au PAN2

On se donne une fonction distance, en fonction des différentes

```java
public class DTW {
    double[] s;
    double[] t;

    public DTW(double[] s, double[] t)
    {
        this.s = s;
        this.t = t;
    }

    public double DTWDistance()
    {
        double[][] dtw = new double[s.length][t.length];
        for (int i = 0; i < s.length; i++)
        {
            dtw[i][0] = Integer.MAX_VALUE;
        }
        for (int j = 0; j < t.length; j++)
        {
            dtw[0][j] = Integer.MAX_VALUE;
        }
        dtw[0][0] = 0;

        double cost;
        for (int i = 0; i < s.length; i++)
        {
            for (int j = 0; j < t.length; j++)
            {
                cost = distance(s[i], t[j]);
                dtw[i][j] = cost + min(dtw[i-1][j], dtw[i][j-1], dtw[i-1][j-1]);
            }
        }
        return dtw[s.length][t.length];
    }

    private double min(double a, double b, double c)
    {
        if (a < b)
        {
            if (a < c)
            {
                return a;
            }
            return c;
        }
        else
        {
            if (b < c)
            {
                return b;
            }
            return c;
        }
```

Annexe L. Module classification, avancement PAN2

Encadrant : Michel Roux
Élèves : Karim Khandid, Robin Shin, Alexandre Poinso

Fonctionnement attendu : À partir d'une séquence de coordonnées fourni par la Kinnect, le module renvoie des attributs musicaux correspondant à cette séquence, en l'ayant comparée (calcul de similarités) à des séquences pré-enregistrées.

Avancement : Un rendez-vous a eu lieu avec l'expert juste avant le PAN 2. À cette occasion, le principe assez détaillé a été défini, ainsi que le travail à effectuer, dont les tests à mettre en place. Notre encadrant nous a donné des conseils pour l'algorithme à mettre en place, et a soulevé les difficultés que nous devrons traiter (impact du paramètre temps, et recalage de coordonnées). Nous devrons être capables de réfléchir à comment traiter ces difficultés, puis de coder l'algorithme rapidement après PAN 2, et de commencer alors les tests.

Difficultés : La principale difficulté a été en fait de mettre en place le module. Nous pensions d'abord que le module Classification proposé par Slim Essid conviendrait à nos attentes, mais lors du rendez-vous avec Slim Essid, celui-ci nous a annoncé que nos demandes ne correspondaient pas au module qu'il encadrait, surtout en raison du paramètre dynamique présent dans notre projet. C'est pourquoi nous nous sommes tournés vers notre tuteur Michel Roux, qui nous a annoncé qu'il encadrerait lui-même ce module.

Annexe M. Module détection de rythme, avancement PAN2

Encadrant : Bertrand David
Réalisé par : Stan Hannebelle, Khadija El Attar

Fonctionnement attendu : Ce module a pour but d'exploiter les données fournies par la Kinect , notamment l'évolution des positions des différentes parties du corps de l'utilisateur dans le temps , afin de détecter le tempo et le rythme de la danse , qui seront définis respectivement par les pas et le mouvement des mains. De plus , ce module nous permettra d'évaluer la richesse de la danse reliée aux variations des membres de l'utilisateur tel que plus la danse est variée plus la musique contient des sons élémentaires différents .

Avancement : Après une première réunion avec l'expert de ce module, nous avons étudié les documentations qu'il nous a fourni ce qui a engendré de nombreuses réflexions sur une méthode efficace pour la détection du rythme d'un signal. Une méthode de filtrage a été finalement retenue et validée par l'expert se déroulant en deux parties : détection de l'enveloppe énergétique du signal et détection des fronts montants. Nous avons ensuite implémentée une première version du code en Java qu'on a envoyé à l'expert.

Difficultés : A cette étape , nous avons rencontré quelques difficultés puisque la détection de rythme s'applique en général à des fichiers sonores et des signaux en Volt contrairement à nos entrées qui sont des positions en 3D de plusieurs points du corps , la notion d'énergie n'est alors plus la même . De plus, il nous était difficile de trouver une méthode convenable pour la détection sachant qu'on devait maîtriser les interprétations physiques des différentes opérations mathématiques afin d'en exploiter l'effet pour notre but. Enfin, le choix de quelques paramètres nécessaires à notre code ne nous est toujours pas clair.

Annexe N. Module lecture, capture et traitement audio, avancement PAN2

Encadrant : Jean-Claude Dufourd
Élèves : Jamal Skouri, Alexandre Poinso

Fonctionnement attendu : À partir des instructions fournies par le module Séquenceur Musical, ce module doit d'une part produire la musique adaptée à la danse de l'utilisateur, à partir de sons de bases auxquels on a accès, et d'autre part enregistrer cette musique et la rendre accessible par l'application Android.

Avancement : Des contacts ont eu lieu avec l'expert de ce module, et celui-ci a fourni la bibliographie nécessaire à l'implémentation de ce module. Nous sommes actuellement en train de lire ces documents et de réfléchir au code à mettre en place, dans les délais fixés par l'expert, qui nous a fixés PAN 3 comme date limite.

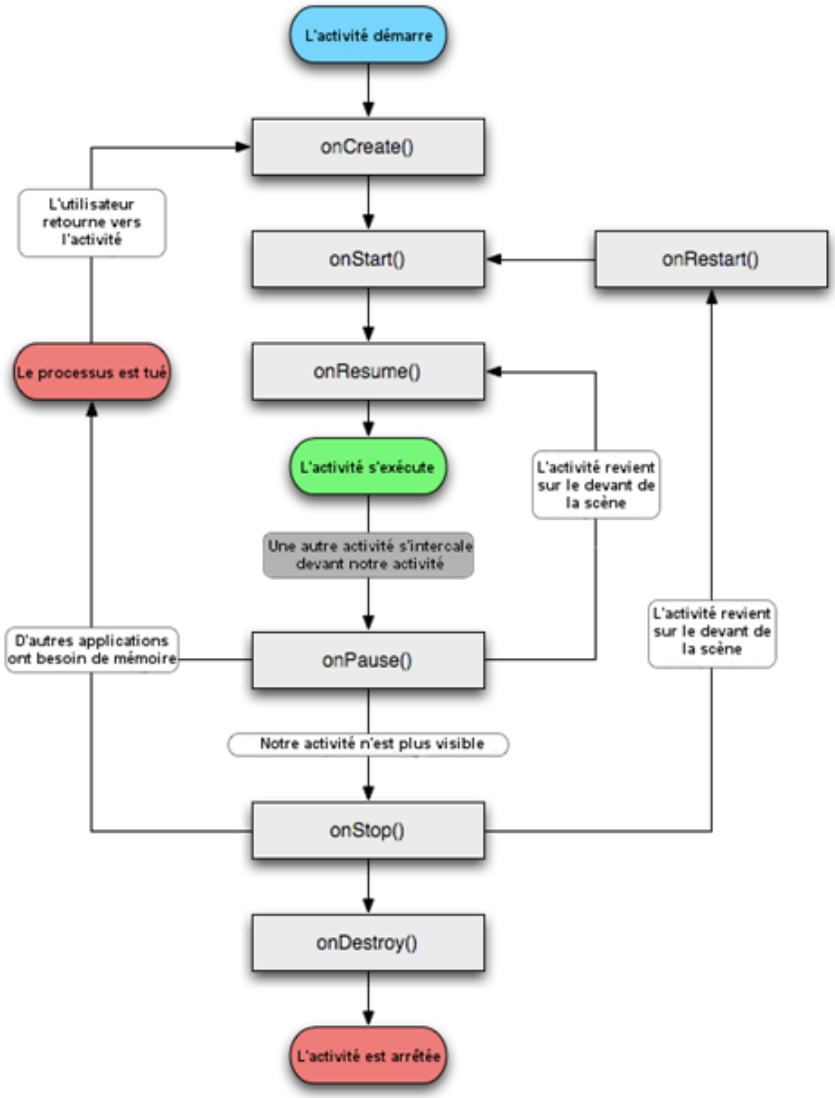
Difficultés : La plus importante difficulté ici a en fait été la définition de ce module. En effet, nous avons dû nous entretenir à la fois avec en Jean-Claude Dufourd et Bertrand David, pour bien définir ce module et le module "Séquenceur Musical". De plus, Bertrand David propose un module intitulé "Synthèse Audio" que nous pensions adaptés pour nous, mais ce nom désigne en fait la création de fichier sons, alors que nous voulons simplement en utiliser des déjà créés.

Annexe O. Module application Android, avancement PAN2

Encadrant : Jean-Claude Dufourd
Réalisé par : Camille Demasson, Khadija El Attar

Fonctionnement attendu : Réaliser une application Android qui permet à l'utilisateur de communiquer le volume et style musical souhaités, et de parcourir les anciennes musiques précédemment créées.

Avancement : En s'aidant des cours disponibles en ligne pour la prise en main de l'environnement de développement et des spécificités des applications Android, nous avons tout d'abord compris des notions essentielles pour la manipulation des applications Android dont : la notion d'activité et le cycle de vie d'une application.



Ensuite, nous avons installé et configuré les outils nécessaires notamment Android Studio , pour se lancer dans la découverte des manières de réaliser et manipuler les

applications Android avec leurs interfaces graphique . Nous sommes en cours d'apprentissage mais nos résultats sont positifs et nos applications , simples à ce stade , tournent sur téléphone.

Difficultés : Nous avons rencontré quelques difficultés dans la création d'une nouvelle activité.

Annexe P. Module Kinect, avancement PAN2

Encadrant : Jean Le Feuvre
Élèves : Robin Shin, Thibaud Lemaire

Fonctionnement attendu : Nous devions faire communiquer la Kinect avec le programme Java, exploiter le squelette et enregistrer des séquences de mouvements. Nous devions également documenter notre bibliothèque.

Avancement : Nous avons réussi à installer et utiliser la bibliothèque J4KSDK. Le code compile. Cependant nous n'avons pas pu tester le code car nous n'avons pas de PC (condition nécessaire à l'installation du SDK kinect). Nous avons emprunté un PC pour réaliser les tests, mais une DLL était manquante. Nous sommes en attente d'un PC spécifique pour le développement de notre système (commandé auprès de Bertrand David) ainsi que de la résolution de ce problème de DLL.
La documentation a été entamée.

Difficultés : La première vient du fait que l'on ne peut tester que sous Windows. La seconde du non chargement d'une DLL lors de l'exécution. Celle-ci est présente mais visiblement, notre bibliothèque ne sait pas où la trouver.

Annexe Q. Module détection de rythme, tests programmés

Encadrant : Michel Roux
Élèves : Stanislas Hannebelle, Khadija El Attar
Entrées / Sorties du module :
Entrées : Coordonnées spatio-temporelles de point du squelette de l'utilisateur délivré par la Kinect
Sorties : Rythme de la danse de l'utilisateur

1/ Tests standards, sans lien avec le reste du projet

Contexte : Juste après l'implémentation du module
(algorithme codé sous Java), afin de contrôler le bon fonctionnement de celui-ci.
Conditions : Un utilisateur effectue une danse sur un rythme (BPM)
connu.
Résultat attendu : On souhaite que le programme
retourne le BPM sur lequel l'utilisateur dansait.
Problèmes possibles : Difficultés liées à la complexité de la danse de l'utilisateur. On pourra alors refaire un test avec un mouvement plus simple (Bras uniquement) avant d'amélioré le programme.

Resultat du test : Ce test effectué avec Bertrand David montre que les données en provenance de la Kinect sont très fiables et peu bruitées. Le principal problème vient du fait que la fréquence d'échantillonnage n'est pas constante, ce qui nécessite une interpolation linéaire des points du squelette. Il apparait que ces données vont permettre de récupérer le rythme assez facilement, sans être trop gourmand en ressources.

2/ Tests d'intégration dans le projet

Contexte : Lorsque notre projet sera suffisamment avancé pour une intégration, nous pourrons effectuer ce test.
Conditions : on demande à l'utilisateur d'utiliser notre application.
Résultat attendu : On souhaite que la musique lancée par notre application présente le rythme de la danse de l'utilisateur.
Problèmes possibles : Des problèmes de synchronisation avec le danseur peuvent apparaitre.

Annexe R. Module calcul dynamique de similarités, tests programmés

Encadrant : Michel Roux
Élèves : Karim Khandid, Robin Shin, Alexandre Poinso

Entrées / Sorties du module :
Entrées : Données fournies par la Kinect, c'est-à-dire des séquences de coordonnées spatio - temporelles
Accès à des séquences similaires pré-enregistrées
Sorties : Des attributs, reliés à la séquence de coordonnées la plus proche de celle fournie, permettant de qualifier la musique, transmis au module "Séquenceur Musical"

1/ Tests standards, sans lien avec le reste du projet

Contexte : Juste après l'implémentation du module (algorithme codé sous Java), afin de contrôler le bon fonctionnement de celui-ci

Conditions : Les élèves créent des séquences de coordonnées du type de celles sur lesquelles le module s'applique, du même format que celles fournies par la Kinect (coordonnées spatio-temporelles).
Par exemple, des formes simples peuvent être implémentées (exemple : créer un "Z", un "N"...) , puis on peut ajouter du bruit sur les signaux (un "Z" ou un "N" courbé).
L'influence du temps sera peut-être dans un premier temps négligé (c'est à dire qu'on considère d'abord des signaux échantillonnés de la même manière) pour vérifier le bon fonctionnement dans ce cas simple, puis les tests seront effectués en ajoutant ce paramètre.

Résultat attendu : On vérifie ici le bon fonctionnement du module seul. Ainsi, par exemple, il faut qu'avec en entrée un "Z" courbé, et dans les séquences enregistrées un "N" et un "Z" (droit), le module doit rapprocher le "Z" courbé du "Z".

Problèmes possibles : Influence du facteur temps dans le calcul de similarité
Il faudra aussi sûrement avoir à recaler les séquences de coordonnées dans un même repère (ce qui revient à négliger l'effet d'une translation)

Résultat du test : Dans le cas décrit ci-dessus (deux séquences de coordonnées proches, et une troisième bien distincte), le programme calcule bien une faible distance entre les deux séquences proches, et une bien plus importante entre une de celles-ci et la troisième. L'algorithme semble donc opérationnel.

2/ Tests d'intégration dans le projet

Contexte : Lorsque les premiers tests seront concluants, que le module Kinect fournira des séquences de coordonnées, et que l'on aura décidé des séquences à enregistrer et des attributs musicaux à donner à chacune (travail commun à tout le groupe ?)

Conditions : On applique le module avec en entrée les séquences fournies par la Kinect, et en ayant rempli (en partie au moins) la base de données de séquences.

Résultat attendu : Il faut bien évidemment que là encore le module soit en mesure capable de trouver la séquence enregistrée la plus proche de celle en entrée.

Problèmes possibles : Contrairement aux premiers tests, on aura ici des signaux réels (et non parfaits), il faudra peut-être adapter le module en conséquence
L'objectif étant toujours de générer la musique en temps réel, il faudra faire attention au temps pris par l'algorithme, et possiblement avoir des choix à faire entre rapidité et performance optimale du module

Annexe S. Module lecture, capture et traitement audio, tests programmés

Encadrant : Jean-Claude Dufourd
Élèves : Jamal Skouri, Alexandre Poinso

Entrées / Sorties du Test :
Entrées : Instructions provenant du module "Séquenceur Musical"
Accès à des fichiers sons basiques (courts)
Sorties : Musique correspondant à la danse de l'utilisateur
Possibilité d'enregistrer cette musique pour lecture via l'application

1/ Tests de bon fonctionnement du module
Contexte : Juste après avoir écrit l'essentiel du code du module

Conditions : Les élèves choisissent des fichiers sons quelconques afin de tester que le module réalise ce pour quoi il est implémenté. On imagine tester d'abord la lecture des fichiers, avant de tester la possibilité d'enregistrement.

Résultat attendu : On doit être capable d'accéder aux fichiers sons choisis, de les lire, et d'avoir la sortie audio correspondante. En outre, on doit aussi pouvoir les enregistrer, et l'application doit avoir accès à ces données enregistrées pour pouvoir les relire.

Problèmes possibles : Comment effectuer le lien entre ce module et l'application pour l'enregistrement et la relecture des musiques ?

Résultat du test : les fonctionnalités standard sont opératives. Ainsi nous sommes capable de jouer un fichier .wav, d'en écrire un simple, et d'enregistrer un son quelconque dans un fichier .wav.

2/ Tests de bonne adéquation du module avec le reste du projet
Contexte : Quand on aura réussi avec succès les premiers tests, et que le module "Séquenceur Musical" fournira des données.

Conditions : On utilise cette fois les données fournies par le module "Séquenceur Musical", et les élèves ont répertoriés des fichiers sons basiques auxquels le module a accès. Là encore, on imagine tester d'abord la sortie audio "en direct", puis la partie enregistrement.

Résultat attendu : Il faut que le fichier son en sortie corresponde aux instructions du “Séquenceur Musical”.

Annexe T. Module Kinect, tests programmés

Encadrant : Jean Le Feuvre
Élèves : Robin Shin, Thibaud Lemaire

Entrées / Sorties du Test :
Entrées : Flux vidéo en provenance de la kinect
Sorties : flux vidéo et squelette des danseurs

1/ Tests de bon fonctionnement du module
Contexte : Juste après avoir écrit l'essentiel du code du module

Conditions : Une personne effectue des mouvements devant la Kinect dans des conditions de plus en plus dégradées (faible luminosité, mouvement rapides ou de grande amplitude, etc…). On peut également essayer de surcharger la machine pour vérifier que l'application est assez réactive pour renvoyer 20 IPS.

Résultat attendu : Le module doit être capable de fournir les données du squelette correctement et assez fréquemment.

Problèmes possibles : Il est possible que l'application ne soit pas assez rapide…

Résultats du test : Le modules fonctionne très bien et est peu gourmand en ressources. La fréquence d'acquisition est stable et tourne autour de 30Hz ce qui va au delà de nos espérances. En faible luminosité, le module répond bien et les données sont fiables.
Il nous reste cependant à effectuer un test en condition de luminosité dégradé, c'est à dire dans un environnement saturé en UV (i.e. lumière du jour).

2/ Tests de bonne adéquation du module avec le reste du projet
Contexte : Quand on aura réussi avec succès les premiers tests, et que les modules classification, détection du rythme et richesse exploiteront les données du squelette.

Conditions : On vérifie que les données sont correctement formatées. Celles-ci doivent respecter les conditions fixées par les interfaces Java définie par les responsables intégration.

Résultat attendu : Intégration parfaite et fluide.

Résultat du test : Le module s'intègre bien avec le reste, notamment grâce à l'interface KinectListener qui décrit comment les modules doivent recevoir les nouvelles données de la Kinect, par évènements.

Annexe U. Module classification, avancement PAN3

Il nous manquait une fonction distance, nécessaire à l'execution de l'algorithme qui a été codé en Java et présenté lors du PAN 2. Nous avons essayé l'algorithme avec la distance euclidienne, et la borne supérieure (aussi appelée distance de Tchebychev). Nous préférons plutôt la distance euclidienne, qui permet de "lisser" la différence entre deux séquences.

En effet, si on imagine le cas où deux séquences S1, S2 sont quasiment identiques, à l'exception d'un seul instant où leur différence est énorme, et où une troisième séquence S3 telle que S1 et S3 sont moyennement distantes à tout instant, alors la distance de Tchebychev donnera une distance bien plus grande entre S1 et S2 que celle entre S1 et S3, au contraire de la distance euclidienne, qui, prenant en compte tous les instants échantillonés, atténuera cet écart, voire donnera une distance plus petite entre S1 et S2. C'est pourquoi notre programme utilise pour l'instant la distance euclidienne, mais si plusieurs tests avec les données de la Kinect se montrent peu concluants, il sera bien entendu possible d'envisager de ré-essayer avec la distance de Tchebychev.

Enfin, nous devions également tester la classification par DTW sur des exemples simples (des motifs en forme de N ou de Z par exemple) et à la main. Cet essai à la main a été concluant.

Les tests utilisant le programme ont été réalisés principalement sous Python, étant donné que ce langage de programmation permet beaucoup plus facilement un affichage graphique des données traitées (tableau de coordonnées) que Java, mais l'algorithme est bien sûr le même.

Enfin, en ce qui concerne l'adaptation de l'algorithme aux spécificités de notre projet, nous avons codé une fonction d'interpolation. En effet, les échantillons de la Kinect ne se font pas à temps constants alors que notre algorithme ne prend en compte pour le calcul de la distance que les coordonnées spatiales des points, supposant que les deux séquences sont échantillonés aux mêmes instants, d'où la nécessité pour nous d'adapter les données de la Kinect. Comme cela avait été suggéré par l'expert, nous avons eu recours à une simple interpolation linéaire pour cela.

Cependant, nous n'avons pas encore essayé la solution de la fenêtre glissante pour résoudre notre problème de détermination du début et de la fin d'un mouvement.

Annexe V. Module détection de rythme, avancement PAN3

Avancement:
Après l'étude des données sortant d'une Kinect qui capture des pas de danse, nous avons développé deux programmes qui prennent en entrée les données sortant de la Kinect et qui renvoient le rythme de la danse en BPM. L'un sous Matlab (pour effectuer des tests rapidement ensuite sous JAVA) qui travaille directement sur les coordonnées des points. L'autre sous JAVA qui travaille sur la distance séparant l'épaule droite du poignet droit. Le premier programme utilise une méthode de filtrage se basant sur la détection d'enveloppe puis celle des pics après un lissage de la fonction,quant au second, il utilise la fonction FFT (Fast

Fourier Transform) qui effectue une transformée de Fourier discrète. Nous avons enfin testé ces différents programmes. Pour des premiers résultats réussis, nous nous sommes orientés vers une autre plus simple méthode facilement testable sur Matlab à travers l'interpolation linéaire d'une coordonnée (x) de la main droite et ensuite la fonction FFT ce qui nous permet d'avoir le BPM.

Difficultés

Nous avons rencontrés quelques difficultés notamment liées à la Kinect. En effet, cet outil ne capture pas des valeurs à intervalles de temps réguliers. Ainsi, par une méthode d'interpolation linéaire, nous avons modifiés nos données pour qu'elles puissent être exploitables par la fonction FFT (Fast Fourier Transform). Egalement, la détection du bon rythme est encore perfectible et le problème de la bonne synchronisation entre la piste audio et la danse devra être corriger pour le PAN 4.

Annexe W. Module lecture, capture et traitement audio, avancement PAN3

Nous avons implémenté les trois livrables demandés par l'expert, concernant des fichiers audios au format .wav uniquement. Ces livrables sont :

- Un programme jouant un fichier .wav et affichant certaines de ses caractéristiques
- Un programme créant un fichier .wav simple et retournant le spectre du son ainsi créé
- Un programme capturant du son, et l'enregistrant dans un fichier .wav

Pour l'adaptation du module à notre projet, nous ré-utiliserons, ou adapterons au moins, le programme jouant des fichiers audios, ainsi que celui enregistrant des fichier audios (notre projet prévoit que la musique jouée soit ensuite mise à disposition de l'application).

Enfin, nous sommes actuellement en train de réfléchir à la manière d'adapter le module au projet. Dans un premier temps, nous pensons devoir être capable de jouer simplement la pulsation au rythme déterminé par le module "détection de rythme". Ensuite, il faudra être capable de jouer simultanément différents sons basiques.

Annexe X. Module application Android, avancement PAN3

Nous avons travaillé sur différentes fonctionnalités d'une application. Tout d'abord nous avons étudié comment passer de notre application à une autre déjà existante (Gmail) pour rattraper notre retard dans les objectifs du PAN 2.

Ensuite, nous avons développé des activités capables de lire des fichiers media et en particulier des vidéos, ce qui constitue une des fonctions principales de notre application.

Enfin, nous avons travaillé sur la communication entre le mobile et l'ordinateur grâce au protocole HTTP ; notre but étant d'envoyer un octet d'information à l'ordinateur de fituring pour lui renseigner, dans un premier temps, la valeur du volume sonore souhaité par l'utilisateur.

Annexe Y. Module Kinect, avancement PAN3

Depuis le PAN 2, nous avons enfin reçu le micro-ordinateur qui a permi de débloquer notre situation. En effet, ne possédant que des MacBook, il nous était impossible de tester notre code prêt depuis un moment pour vérifier son bon fonctionnement car le SDK Kinect n'est compatible que sur Windows.

Après installation du SDK, nous avons pu corriger notre code Java pour obtenir un fichier .csv contenant les coordonnées du squelette nécessaires pour notre projet.

Il nous est maintenant également possible d'enregistrer les données fournies par la Kinect.

Nous avons également implémenté l'interface rédigée dans le cadre du module test et intégration qui permet aux différents modules d'acquérir les données en temps réel par un système de listener.

Il nous reste donc, en accord avec Jean Lefeuvre, à débugger la gestion de l'enregistrement et la restitution de données kinect. Cela ne nous sera pas utile pour le projet à proprement parler mais pourra l'être grandement pour la réalisation des différents tests.

Annexe Z. Module Test et Intégration, avancement PAN3

Depuis le PAN2, nous avons travaillé à l'élaboration des interfaces entre les différents modules. L'objectif est de s'entendre sur le format des données échangées par ces modules.

Pour s'assurer que toutes les données sont traitées dès leur mise à disposition, nous avons conçu un système de "listeners". Dans notre cas, les modules qui utilisent les données de la kinect doivent implémenter l'interface KinectListener. Les modules en question s'enregistrent alors auprès du module Kinect de manière à être avertis des données disponibles. Concrètement, lors de l'arrivée de nouvelles données, le module kinect appelle les fonctions Listeners pour leur transmettre le nouveau squelette. Ce type de programmation évènementielle permet de tirer pleinement profit de la kinect et de travailler en temps réel.

Concernant les tests, ceux-ci ont étés entammés dans la plupart des modules avec des résultats encourageants. Le module Kinect répond à nos attentes et fournit des données qui semblent parfaitement exploitables par les autres modules. En ce qui concerne les tests globaux, ceux-ci sont encore succincts mais il apparaît que les interfaces facilitent grandement la tâche des intégrateurs. Notamment en ce qui concerne l'implémentation des Listeners qui permettent de déployer une architecture temps réel évènementielle.