Indexation multimédia pour la reconnaissance de gestes et le contrôle de la production sonore

Guillaume Claude

Guillaume Claude

Master informatique spécialité
Recherche en informatique

Encadrement

Gildas Ménier
Sylvie Gibet

Université de Bretagne sud - Laboratoire Valoria - équipe Seaside

Indexation multimédia pour la reconnaissance de gestes et le contrôle de la production sonore

Résumé

Une partie de plus en plus grande de la musique d'aujourd'hui est d'origine électronique. Il lui est souvent reproché de manquer d'expressivité en comparaison à la musique traditionnelle. Pour pallier à ce problème, différentes méthodes ont vu le jour proposant l'utilisation de différentes techniques de synthèse ou de contrôle. Dans ce document nous proposons une approche utilisant des méthodes d'apprentissage dans le but d'estimer le son qui serait produit par instrument en fonction d'un geste de jeu obtenu par capture de mouvement. L'avantage de cette méthode est qu'elle permet de positionner le son résultant de la requête par rapport à des sons connus et ouvre ainsi des possibilité pour, par exemple, définir les paramètres de synthèse. Ce rapport comporte trois grandes parties : une étude bibliographique, une description détaillée de la méthode employée et une mise en pratique de cette dernière au travers d'un framework dans le cadre de la reconnaissance du jeu de timbale.

Table des matières

Introduction

Le but d'un instrument de musique est de traduire les gestes d'un musicien en sons. Plus le musicien est expérimenté, plus il va maitriser ces gestes et donc être capable de s'exprimer de façon précise. Moore [16] utilise la notion d'«intimité du contrôle» pour déterminer la correspondance entre l'ensemble des sons musicaux productibles par l'instrument et les capacités psychophysiologiques d'un musicien rompu à son utilisation. En en fonction de ces gestes, le musicien va influer sur différents paramètres de la musique [9] :
- le tempo
- la tonalité
- le niveau sonore
- le timbre
- l'articulation des sons (notes)

Ces paramètres vont permettre au musicien d'exprimer des émotions. L'instrument est donc à la fois un contrôleur du son et sa source.

De nos jours, une grande partie de la musique est d'origine électronique. On lui reproche souvent son manque d'expressivité en comparaison à la musique traditionnelle. Pour palier à ce problème, de plus en plus de méthodes de synthèse (modèles physiques, synthèse additive ...) et de contrôle de cette synthèse (capture de mouvements, reprise des structure d'instruments acoustiques ...) sont proposées. Un nouveau type d'instrument à alors vu le jour : les DMI *(Digital Music Instrument)*. Contrairement aux instruments traditionnels, le contrôleur et le synthétiseur du DMI sont deux parties indépendantes. La difficulté consiste alors à trouver des solutions logicielles et matérielles offrant la même souplesse qu'un instrument classique.

Dans ce document nous proposons une méthode de mise en correspondance (*mapping*) des gestes du musicien obtenus par motion capture et des sons qui seraient produits par ces gestes. Dans un premier temps nous proposons une étude bibliographique décrivant en premier lieu les approches possibles du mapping gestes/sons puis les méthodes utilisées dans la reconnaissance de gestes. Dans un second temps nous décrivons le principe de notre approche et des méthodes utilisées. Enfin, nous étudions une implantation de cette méthode sous la forme d'un framework dans le cadre du jeu de timbale. En conclusion, nous évaluons les points forts et points faibles de cette méthode et proposons des ouvertures possibles du sujet et des perspectives de recherches.

I - Etude bibliographique

1° - Mapping de gestes musicaux et de sons

Nous allons tout d'abord nous intéresser aux méthodes utilisées pour définir, à partir d'un geste, le son produit par un système. Il existe deux grandes catégories de solutions à ce problème : les méthodes supervisées, qui associent directement des informations du gestes à un système de synthèse, et les méthodes non supervisées qui passent par des techniques de *machine learning*.

a - mapping supervisés

Les méthodes supervisées font correspondre des paramètres obtenus par analyse des données du mouvement aux paramètres de synthèse. La figure 1 est une représentation de mapping supervisé dans sa forme la plus simple comme elle est utilisée dans [15].

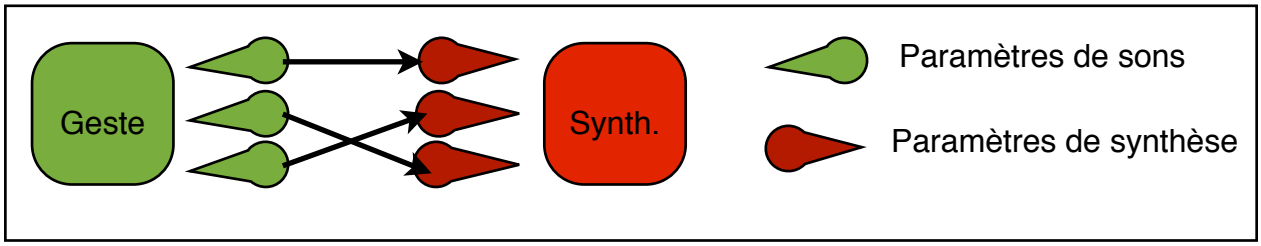


Figure 1 : Principe du mapping supervisé

Il existe plusieurs types de mappings supervisés. La figure 1 représente un mapping qui fait correspondre un paramètre du geste à un seul paramètre de synthèse. Mais d'autres méthodes peuvent utiliser différentes combinaisons de paramètres afin d'obtenir un contrôle plus fin de la synthèse [17]. Il est par exemple possible d'utiliser plusieurs paramètres du geste pour définir un seul paramètre de synthèse, d'utiliser un seul paramètre du geste pour définir plusieurs paramètres de synthèse, ou bien une combinaison de ces différentes approches (figure 2).

Les mappings supervisés les plus avancés [18] passent par des couches intermédiaires pour permettre d'utiliser plusieurs contrôleurs avec un même synthétiseur et inversement (figure 2). En règle générale, le mapping supervisé est effectué via une

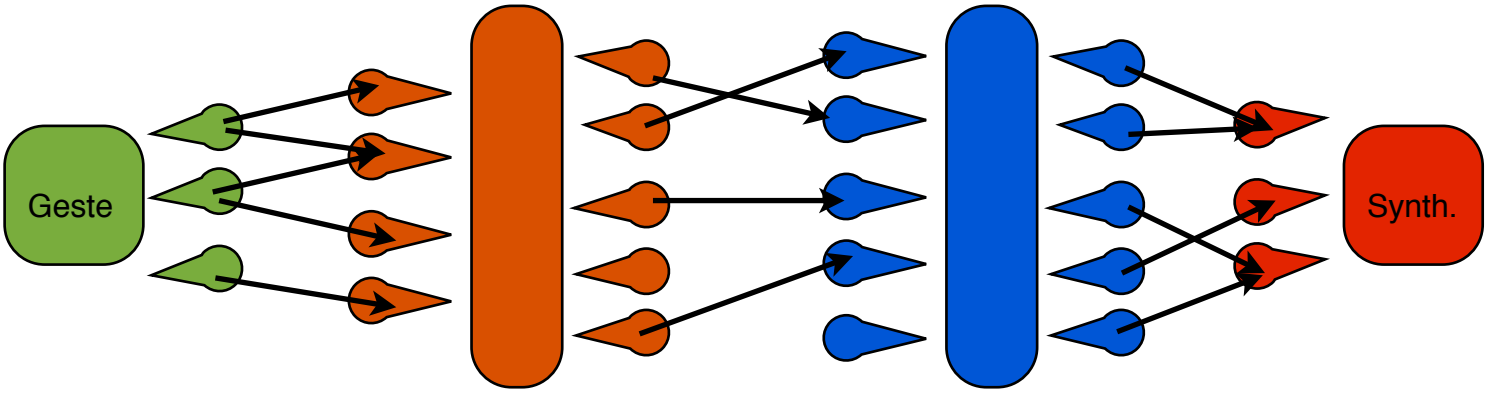


Figure 2 : Mapping utilisant trois couches

librairie utilisant des techniques d'analyse et de traitement du signal (recherche d'extremums, calculs de vitesses etc...).

La principale difficulté lors de la réalisation d'un mapping explicite est de trouver quel paramètre ou quelle combinaison de paramètres définit un paramètre précis du son synthétisé (i.e. force de frappe associée au volume sonore sur un instrument à percussion).

b - Mapping non supervisé

Le mapping non supervisé n'extrait pas directement les paramètre du geste pour les utiliser lors de la synthèse. Le geste capturé sert de requête dans une base de donnée de mouvements ([3-5] et [8]). Pour effectuer le mapping, les gestes sont associés à des jeux de paramètres de synthèse ou à des sons enregistrés. Ce type de base de données utilise des techniques de *machine learning* et de reconnaissance de séries temporelles multidimensionnelles qui seront détaillées dans les sections suivantes de ce document.

Les principaux problèmes du mapping non supervisé sont la non-reconnaissance d'un geste ou le retour de mauvais résultats qui entrainent de mauvaises réactions de la part du système voir pas de réaction du tout. Mais l'avantage est qu'il est possible d'associer les gestes à autre chose qu'un synthétiseur, comme par exemple des sons pré-enregistrés.

c - Comparatif

Les deux types de *mapping* ont chacun leurs points forts et leurs points faibles. Le tableau suivant résume ces différences :

	Supervisé	**Non supervisé**
+	•temps réel	•temps réel possible (en fonction des méthodes) •permet d'associer des sons enregistrés ou des paramètres de synthèse
-	•besoin de connaitre quels paramètres des gestes associer à un ou plusieurs paramètre de synthèse	•Besoin d'apprentissage •Possibilités de faux-positifs

Dans la suite de ce document, nous avons choisis de nous intéresser à un *mapping* non supervisé du fait de sa capacité à utiliser des sons enregistrés ou synthétisés. Nous allons donc maintenant étudier certaines méthodes utilisées dans le contexte de la reconnaissance de mouvements ou de séries temporelles.

2° - Reconnaissance de gestes

Depuis quelques années, la capture de mouvements est devenue de plus en plus courante que ce soit dans la simple utilisation de l'outil informatique (écrans et tablettes tactiles), dans les médias (jeux vidéos, cinéma) ou dans le sport (analyse des mouvements des sportifs pour améliorer les performances). L'utilisation de la capture de mouvements comme interface permet de simplifier des tâches (i.e. animation d'humanoïdes pour le cinema) ou de rendre une interaction avec l'ordinateur plus naturelle (i.e. balayage de la main pour tourner les pages d'un document numérique) et de tenir compte d'un certain réalisme des actions produites par rapport à des tâches diversifiées. Le problème étant que, pour l'instant, le coût du matériel permettant la capture de mouvement est encore élevé, même si certaines technologies commencent a apparaitre dans les foyers comme les écrans tactiles ou les périphériques pour jeux vidéos.

Le problème traité dans cette partie sera celui de la création et de l'utilisation d'une base de données de gestes. Il s'agira donc d'analyser les principales techniques pouvant être appliquées l'indexation et la recherche de mouvements dans une base de données. Les méthodes proposées devront favoriser une discrimination efficace des mouvements de façon à ce que le résultat obtenu (classification ou mouvement similaire dans la base) soit le plus pertinent possible par rapport à un mouvement requête donné.

Dans une première partie nous exposerons la structure des données obtenues lors de la capture de mouvements et les différents points de vues possibles à leur sujet. Dans une seconde nous exposerons le principe général utilisé pour la reconnaissance de mouvements. Dans une troisième nous étudierons différentes méthodes proposées dans la littérature. En annexe nous détaillerons quelques outils régulièrement utilisés dans ce type de problématiques. Le but de ce document n'est pas de présenter un catalogue de toutes les technologies utilisées en reconnaissance de mouvement (ces informations peuvent être trouvées dans la littérature, par exemple [1]), mais de donner une vue d'ensemble structurée des différentes approches possibles du problème.

a - Caractéristiques des données de capture de mouvements

Avant de travailler avec des données, il faut que l'organisation de celles-ci soit bien définie. Dans le cas de la capture de mouvements, ces données comportent deux types d'informations :

- la structure hiérarchique (Figure 3), organisée sous une forme arborescente (le plus souvent ensemble d'articulations ou jointures entre segments qui constituent un squelette).
- l'évolution au cours du temps des données de mouvement qui s'expriment sous la forme de trajectoires aux différentes articulations.

Les données mouvement sont constituées par une suite de valeurs représentant l'état des jointures de la hiérarchie à un instant t (Figure 4), par rapport à leur état initial. Le nombre de valeurs contenues dans cette partie est fonction du nombre de points de mesure sur une durée donnée à un certain taux d'échantillonnage.

Il existe plusieurs possibilités pour représenter les données de mouvement. La plus simple à appréhender étant surement de voir ces données comme un ensemble de n séries temporelles (trajectoires) à une dimension. A chaque pas de mesure correspond une valeur pour chacune de ces séries. Il est également possible de les considérer

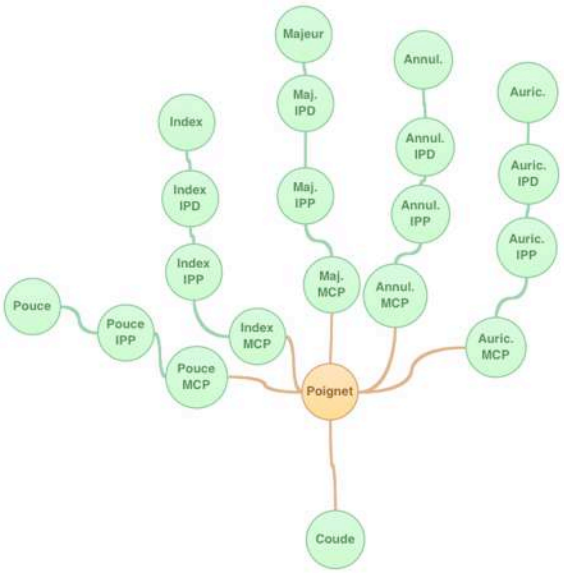


Figure 3 : Exemple de hiérarchie (main)

comme une unique série temporelle multi-dimensionnelle. Dans ce cas, à chaque pas de mesure correspond un point dans un espace à n dimensions. D'autres ([8]) proposent de voir chaque ensemble de mesures à un instant t comme représentant d'une pose du corps observé. Nos données peuvent donc former une série de poses. Enfin, il est possible de considérer ces données comme un tout : une matrice n×m, avec m = d×e avec d la durée de la capture et e le taux d'échantillonnage, et d'effectuer des calculs directement à partir de cette matrice.

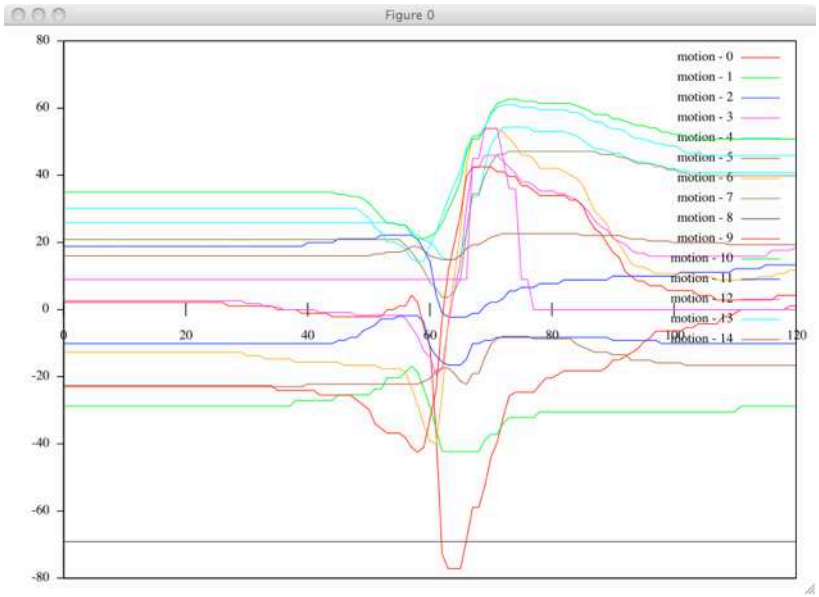


Figure 4 : Exemple de donnée de capture de mouvement (main)

En fonction de l'approche choisie, il est possibles d'utiliser différents outils qui permettront de construire une base de données qui permet soit de reconnaitre la catégorie de mouvements à laquelle appartient un mouvement requête, soit de retrouver des mouvements stockés au préalable et qui sont similaires à ce mouvement requête.

b - Principe général de la reconnaissance de mouvements

Le principe général de la reconnaissance de mouvement est le même que celui des autres méthodes de reconnaissance. Des données sont utilisées dans une phase d'apprentissage, où elles sont intégrées dans la base et sont associées à des classes comme par exemple un type de mouvement. La phase d'apprentissage divise ces données de base dans le but de les utiliser de deux façon distinctes. la première partie est utilisée pour construire la base. La seconde sert à effectuer des tests pour vérifier qu'elle répond de façon correcte à une requête (bonne catégorie ou mouvements similaires). C'est dans la construction et l'utilisation de la base que différentes méthodes peuvent être appliquées, le but de l'application étant de répondre le plus rapidement possible et surtout de retourner une bonne réponse (pas forcément la meilleur).

c - Techniques de reconnaissance de mouvements

1° - Changement d'espace, réduction de données et apprentissage

L'une des difficultés rencontrées lorsque l'on traite des données de mouvement concerne leur aspect multidimensionnel. En effet il faut tenir compte d'au moins 22 mesures par pas de temps pour capturer le mouvement d'une main et entre 40 et 100 pour le mouvement du corps. De plus, capturer et traiter des mouvements diversifiés nécessite de stocker de grandes quantités d'information : 2200 valeurs pour une seconde de capture de la main à 100 hertz, 8,6 Ko avec 4 octets par valeur soit 1 Mo de données pour deux minutes. Il peut être pertinent de changer d'espace de représentation des mouvements afin de ne garder que les éléments les plus pertinents du point de vue de la méthode et de l'application choisies. Ces données caractéristiques du mouvement pourront alors servir dans la base comme une représentation de celui-ci notamment dans l'utilisation d'un indexe.

Un exemple simple de ce principe peut être rencontré dans [5]. Les mouvements sont caractérisés par deux valeurs à savoir l'accélération minimum et maximum du mouvement (respectivement A_{min}, A_{max}). Les données sont ensuite divisées en trois classes suivant trois types de mouvements définis, puis une analyse discriminante linéaire (LDA, c.f. Annexe) est effectuées sur les points. Ces données sont alors stockées en mémoire et servent de points de comparaison pour notre mouvement requête.

Dans le même esprit, [3] propose une réduction de dimensions en utilisant le principe de décomposée en valeurs singulières (SVD, c.f. Annexe), afin de représenter la matrice des données (c.f. chapitre I) par un seul vecteur caractéristique de quelques valeurs. Nous pouvons facilement supposer que, si deux mouvements sont similaires, leurs matrices de données sont similaires, et donc d'après les propriétés de la SVD, leurs quelques premiers vecteurs caractéristiques le sont également (Figure 5). Nous pouvons donc utiliser le premier vecteur, ou les quelques premiers vecteurs, pour comparer deux mouvements et passer d'un calcul sur un grand nombre de valeurs à un calcul sur beaucoup moins de valeurs (22 valeurs au lieu de 2200 dans le cas d'une seconde de capture de la main). [3] va même un peu plus loin dans cette optique, et réutilise la SVD une seconde fois sur la matrice des quelques premiers vecteurs, afin d'en extraire un vecteur caractéristique de dimension réduite.

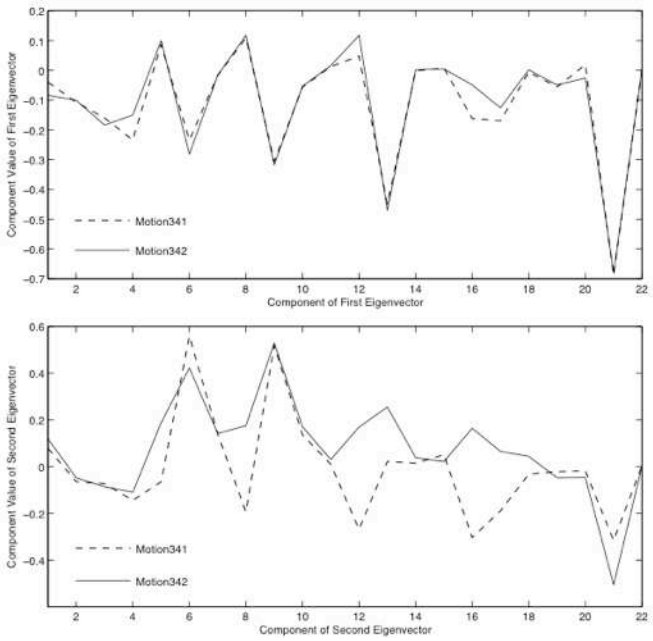


Figure 5 : Comparaison des différentes composantes des deux premiers vecteurs caractéristiques de deux gestes similaires (source [3])

D'autres approches proposent de s'intéresser à la forme générale de la trajectoire afin de ne pas avoir à comparer point par point les données. [2] expose un principe définissant une enveloppe englobante minimum (MBE, *Minimum Bounding Envelope*) autour de la courbe en fonction des valeurs maximum et minimum et en fonction d'un intervalle de temps donné autour du point considéré. Cette enveloppe permet ensuite de définir des rectangles englobants (MBR, *Minimum Bounding Rectangle*) d'une largeur commune définie, et d'une hauteur contenue entre le minimum et le maximum de l'enveloppe dans l'intervalle de cette largeur. Les rectangles ainsi obtenus (Figure 6) servent d'indexes pour la trajectoire dans la base.

Beaucoup de méthodes de reconnaissance utilisent des automates. Le point de vue de ces méthodes sur les mouvements est de les voir comme des séquences de poses, qui peuvent être représentée par les états de l'automate. Un mouvement devient donc une séquence d'états, et cette séquence peut alors être utilisée pour retrouver la catégorie du mouvement. L'utilisation de machines à états finis (FSM, *Finite States Machine*) correspond dans ce cas. C'est par exemple le principe utilisé dans [8]. L'ensemble de données d'apprentissage est utilisé afin que chaque mesure de la capture d'un mouvement représente un point dans un espace à n dimensions. Une fois ces données collectées, l'utilisation de l'algorithme k-means permet de définir les états de notre FSM. le choix du nombre d'état est un facteur important puisque peu d'états réduit la sensibilité aux variations possibles, mais trop d'états impliquerait trop de sensibilité dans la reconnaissance.

L'utilisation des modèles probabilistes, et notamment des modèles de Markov cachés (HMM, *Hidden Markov Model*) [10] dans la reconnaissance de mouvement est une méthode courante. L'idée principale est de supposer qu'un mouvement est issu d'un HMM et d'utiliser les données d'apprentissage pour construire des chaines de Markov (Figure 5) dans le but de les comparer avec une chaine construite à partir du mouvement

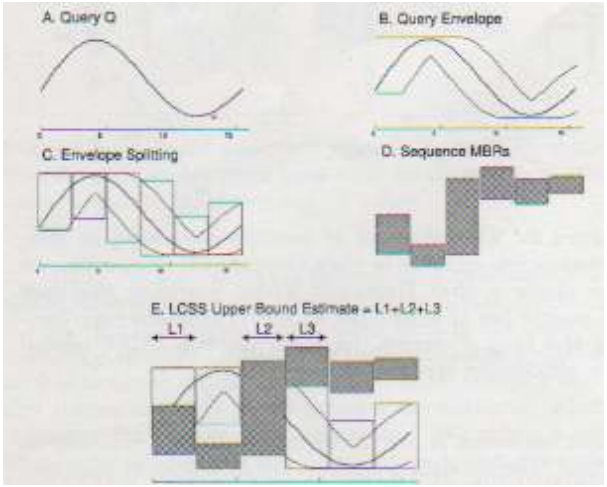
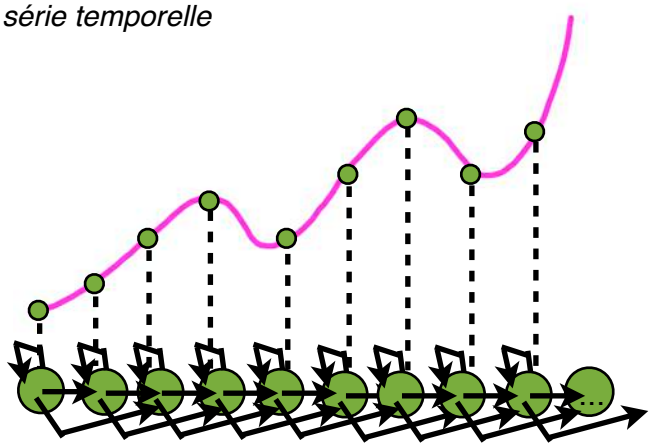


Figure 6 : construction des MBR (source [2])

requête. L'observation d'un état de la chaine est associée à une mesure dans la série temporelle suivant un pas définit (par exemple tous les 10 échantillons) par une loi de probabilité et une probabilité de transition entre états est définie. [4] est un bon exemple d'utilisation des HMM pour la reconnaissance de mouvements et notamment de gestes musicaux. Lors de l'apprentissage, une probabilité suivant une loi normale centrée sur le point mesuré est associée à l'état courant, le maximum étant donné à la valeur de l'observation. Les probabilités de transition entre entre états sont définies telles que P[ili] = a_0, P[i+1li] = a_1, P[i+2 li] = a_2, et une loi normale est utilisée pour les observations. Basé sur les expérimentations, [4] préconise une utilisation telle que a_0=a_1=a_2 = 1/3, modifier ces valeurs permet de gérer les probabilités que le mouvement requête soit plus rapide, moins rapide ou aussi rapide que le mouvement utilisé pour le HMM. Alternativement, [4] propose un modèle n'utilisant que deux transitions possibles a_0 et a_1.

Une fois les données projetées dans ce nouvel espace, elle peuvent être utilisées et organisées dans la base de données.

Figure 7 : Construction d'un HMM à partir d'une série temporelle



2° - Organisation des données et reconnaissance

l'organisation des données dans la base est importante car c'est en grande partie d'elle que va venir la vitesse à laquelle va répondre la base à une requête donnée. Lors de la requête, les données de celle-ci sont projetées dans le même espace que celui utilisé lors de l'apprentissage (deux valeurs pour [5], un vecteur pour [3], ou quelques MBR pour [2]). Ces informations servent à la base pour retrouver les informations demandées (classe ou élément le plus similaire).

La méthode la plus simple consiste à parcourir un a un tous les éléments lors de la recherche. Cette approche couteuse (O(n)) est utilisable dans le cas ou nos informations sont de petite taille comme c'est le cas dans [5]. Lors de l'interrogation de la base, les données du mouvement requête sont soumises au même calcul que précédemment (extraction des A_{min} et A_{max}). Les données ainsi extraites sont comparées aux données en mémoire par l'algorithme des k plus proches voisins (k-nn, c.f. Annexe) dont le résultat donnera la classe à laquelle appartient la requête. Il est envisageable dans ce cas de trier les informations de façon à y accéder plus vite (via une recherche dichotomique par exemple, complexité d'accès en $O(\log_2(n))$). Mais tous les problèmes ne s'y prêtent pas, surtout si la taille de la base devient conséquente. Dans le cas des données multidimensionnelles cela peut être encore pire du au grand nombre de donnée pour un seul point dans l'espace. Pour gérer ce problème, le plus efficace consiste à éliminer les données dont nous sommes certains, à partir d'un test à faible coût, qu'elles ne sont pas pertinentes. C'est le principe des indexes multidimensionnels (c.f. Indexes Multidimensionnels, Annexe).

Dans [2] un R-Tree est construit pour indexer tous les MBR. on peut alors l'utiliser pour retrouver ceux qui se croisent avec les MBR de la requête. On les utilise alors pour retrouver les mouvements associés (via par exemple une table hachée). On peut ensuite retourner les mouvements organisés en fonction de la surface totale partagée avec les MBR de la requête ou les comparer directement via une DTW ou une LCSS (c.f. Calcul de distances, Annexes). Dans [3] un principe similaire est utilisé. Un arbre est construit, avec à chaque étage un découpage en sous ensembles de l'ensemble [-1,1] (puisque les valeurs contenues dans les vecteurs sont compris entre -1 et 1), chacun pointant vers un élément de l'étage suivant. Chaque niveau de l'arbre est associé à une dimension des vecteurs caractéristiques. Les feuilles de l'arbre sont les vecteurs indexés. Lors d'une recherche, on retourne tous les vecteurs qui appartiennent aux sous ensembles contenant les valeurs des dimensions du vecteur requête plus ou moins une marge d'erreur. Nous pouvons alors, comme précédemment, calculer la distance entre ces mouvements

Dans le cas des FSM, lors de la reconnaissance chacun de des points du mouvement lu est analysé pour connaitre l'état auquel il correspond (appartenance à des sous-ensembles flous, c.f. [13]). Si l'état courant est différent de l'état observé précédemment, alors l'état courant est ajouté à la liste des états traversés par le mouvement, et le nouvel état devient l'état courant. Si le nouvel état est un état final (dernier état d'un mouvement d'apprentissage), alors la correspondance entre la chaine d'états observés et la chaine des mouvements liés à cet état final est mesurée et l'on retourne le ou les meilleurs résultats.

Dans le cas des HMM, la reconnaissance se fait en construisant un nouvel HMM au fur et a mesure de la lecture des données et en le comparant avec les HMM construits lors de l'apprentissage via des algorithmes de programmation dynamique comme la DTW ou l'algorithme de Viterbi [14]. [4] propose de calculer pour un automate à n états une valeur

$\alpha_t(i)$ à un instant t pour un état i, avec $b_i(O_t)$ la probabilité d'observer O_t, π le vecteur des probabilités des états initiaux et a_{ij} probabilité de transition de i vers j tel que :

Il est alors possible d'estimer la vraisemblance d'un élément des données d'apprentissage par rapport au mouvement suivit actuellement via :
Plus cette valeur est élevée pour un élément des données d'entrainement, plus cet élément est proche du mouvement suivit actuellement.

3° - Discussion sur les méthodes observées

Equation 1 : calcul de $\alpha_{t+1}(i)$

$$\alpha_{t+1}(i) = [\sum_{i=1}^{n} \alpha_t(i)\, a_{ij}]\, b_i(O_t)$$

Equation 2 : calcul de $\alpha_1(i)$

$$\alpha_1(i) = \pi_i\, b_i(O_1)$$

Comme nous avons pu le voir précédemment, le principe général de ces méthodes

Equation 3 : Vraisemblance

$$S(i) = \sum_{i=1}^{n} \alpha_t(i)$$

est toujours le même. La difficulté est donc d'en choisir une efficace ou d'en proposer une nouvelle plus adaptée aux besoins du problème rencontré.

Les méthodes présentées dans cet état de l'art ont toutes des résultats convaincants (aux alentours de 90% de bons résultats). Cependant, les méthodes prenant en compte plusieurs types de données, comme des séries temporelles autres que du mouvement, semblent avoir de moins bon résultats ([2] par exemple peut voir ses scores chuter à tout juste plus de 50% en fonction du type de séries analysées que des méthodes étudiées spécifiquement pour un domaine précis : [5] obtient des scores allant jusqu'à 100% de bonnes réponses pour certains mouvements. Cependant cette méthode est efficace dans ce contexte car elle n'est appliquée qu'à une catégorie restreinte de mouvements (trois type d'attaques de l'archet d'un violon). Utiliser l'accélération minimum et maximum du mouvement pour une nombre plus grand de gestes différents ou pour lesquels elles n'influent pas sur le sens du gestes risque de ne pas être pertinent. Le principe des MBR utilisé dans [2] peut avoir des difficultés à reconnaitre certains mouvements proches si la vitesse d'exécution de la requête est trop différente (par exemple deux fois plus rapide) de celle de la référence dans la base. Il est donc important de définir les gestes qui seront utilisés (nombre, durée, vitesse d'exécution etc...).

Il est important de souligner que beaucoup de paramètres entrent en ligne de compte pour la rapidité et la précision de la réponse de la base : le nombre d'éléments pris en compte dans l'algorithme k-nn [5] peut changer le choix d'une classe, le nombre de valeurs conservées pour les vecteurs caractéristiques lors d'une SVD [3], le pas, les probabilités de transitions et les loi de probabilité pour une observation donnée pour les HMM [4], la marge autour de la courbe pour la MBE et la largeur des MBR [2] ou encore le nombre d'états à retrouver par l'algorithme k-means lors de la construction d'un FSM [8]

agissent sur la capacité de la base à éliminer ou non de bonnes ou mauvaises réponses ce qui peut conduire au retour de mauvais résultats ou à l'élimination de bons.

Certaines méthodes envisagées pour la production interactive (comme la production de sons) doivent fonctionner en temps réel, ce qui élimine toutes les méthodes ayant une complexité algorithmique ne permettant pas d'atteindre ces objectifs. Les méthodes demandant des calculs trop couteux ([3] (SVD : $O(n^3)$)) ou sur l'intégralité des données du mouvement ([2], [3] et [5]) sont donc inappropriées. Les méthodes par HMM [4] et par FSM [8] semblent de ce fait mieux indiquées pour cette catégorie de problèmes de par leur capacité à traiter les données dans un flot. Cependant, il est possible de proposer des représentations de mouvements nécessitant une phase préalable d'apprentissage, et d'indexer les mouvements de telle façon que la complexité de recherche dans la base de données soit relativement faible et permette l'utilisation d'algorithmes de reconnaissance plus sophistiqués.

Si l'économie de mémoire est un problème, les méthodes utilisant des représentation de données complexes pour chacun des mouvements d'apprentissage comme les HMM [4] ou les MBR [2] risquent de poser des problèmes. Dans le cas des HMM, [4] propose une solution pour amoindrir le problème en utilisant une fenêtre glissante sur les données afin de ne pas les garder entièrement en mémoire. Dans tous les cas, c'est surtout la quantité de mouvements stockés dans la base qui définit cette charge mémoire et toutes ces méthodes nécessitent une quantité assez conséquente de données d'apprentissage afin d'être efficaces.

d - Conclusion

Nous avons présenté ici différentes méthodes de reconnaissance de mouvements. Nous avons pu voir que le choix des méthodes utilisées pour répondre à un problème de reconnaissance de geste dépend grandement des besoins de ce dernier (temps réel, économie de mémoire, type de données etc...).

Dans la suite de ce document, nous allons présenter une méthode visant à effectuer le *mapping* de manière relative à des données (couples gestes/son) présentes dans une base. Nous utilisons des techniques présentées ici de manière à estimer positionner le son qui serait produit par l'instrument lors de l'exécution du geste requête.

II - Approche proposée

1° - Vue d'ensemble

La méthode présentée ici à pour but d'estimer un son qui serait produit par un geste musical capturé. Il utilise comme références des gestes de jeu g_i du même instrument appartenant à G ensemble des gestes de la base et leurs sons associés s_{gi} appartenant à S ensemble des sons de la base. Pour ce faire, nous utilisons des méthodes de *machine learning*. Dans un premier temps les sons associés aux gestes de la base sont projetés dans un espace P de dimension n. Les distances entre les gestes D et entre les sons E, dans l'espace projeté sont utilisées pour entrainer un perceptron multicouche qui permettra le passage de l'espace des gestes à l'espace de sons projetés.

2° - Espace des sons projeté

Comme vu précédemment, il peut être intéressant de réduire la dimension des données à traiter pour réduire la complexité du calcul.. Dans notre cas, nous voulons représenter chaque sons par un point p_i sa projection dans l'espace P. La difficulté vient du fait que les relations de voisinage dans l'espace S doivent être conservées le plus possible dans P. Pour ce faire, nous utilisons une méthode proche des *self-organizing maps* [19] de T. Kohonen : chaque son s_i est associé à un point p_i dont les coordonnées sont tirées aléatoirement dans cet espace. Ces p_i sont alors déplacés par paires, tirées aléatoirement pour faire correspondre la distance dans l'espace initial et la distance actuelle dans l'espace projeté. La correction des coordonnées des p_i s'effectue en les déplaçants sur la droite passant par ces deux points.

Soient A et B deux sons de S et p_A et p_B leurs projection dans P. E(A,B) une distance entre les deux sons et $d(p_A,p_B)$ une distance entre leurs projections dans P. La formule suivante permet de calculer une correction estimée c :

Equation 4 : Correction des projections

$$\forall x \in [1,n]:$$
$$c = \frac{\left(\left(\frac{p_B[x]-p_A[x]}{d(p_A,p_B)}\right) * E(A,B)\right) - (p_B[x]-p_A[x])}{2}$$

Avec $p_i[a]$ une valeur associée à p_i pour la dimension a de P avec $1 \leq a \leq n$. Les points PA et PB sont alors mis a jour suivant l'algorithme :

$$\forall x \in [1,n]$$
$$p_B[x] = p_B[x] + (c * r)$$
$$p_A[x] = p_A[x] - (c * r)$$

où r représente une valeur faible ($r << 1$) qui permet d'éviter une oscillation autour du résultat optimum. La *figure 8* représente ce principe dans un espace à deux dimensions.

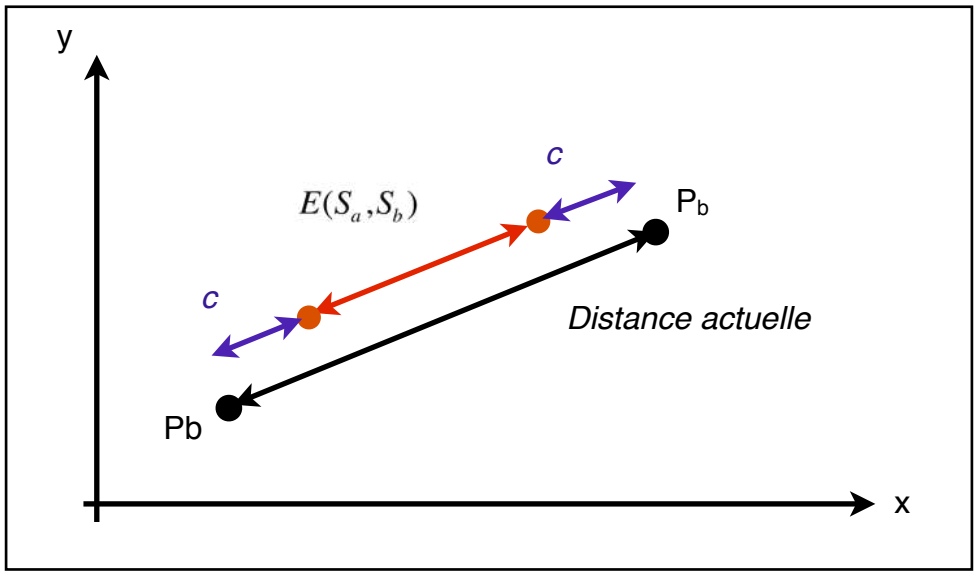


Figure 8 : Correction des coordonnées des projections dans un espace P tel que n=2

Lorsque la somme des corrections sur l'intégralité de l'ensemble se stabilise nous pouvons considérer ce nouvel espace correctement organisé.

3° - Correspondance entre l'espace des gestes et l'espace projeté

Une fois l'espace des sons projeté obtenus, les données (gestes et sons) sont utilisées pour entrainer un réseau de neurones de type perceptron multicouches à k entrées et k sorties qui effectuera le *mapping* entre l'espace des gestes et l'espace projeté. Soit la relation f, telle que g_i -> $f(s_{gi}) = p_i$ vérifiée dans la base pour tous g_i. Pour chaque gestes g_i de la base, nous recherchons les k plus proches voisins V_i telle que k-ppv(g_i) = { V^l_i avec 1≤ l ≤ k }, par rapport à $D^l_i(g_i, V^l_i)$ distance entre le geste gi, et son voisin V^l_i, Les k D^l sont présentées aux entrées du perceptron. Les valeurs attendues en sorties sont les distances $d^l_i(p_{gi}, p^l_i)$,distance euclidienne entre les sons s_{gi} et s^l_i dans P. les figures 9 et 10 résument ce principe :

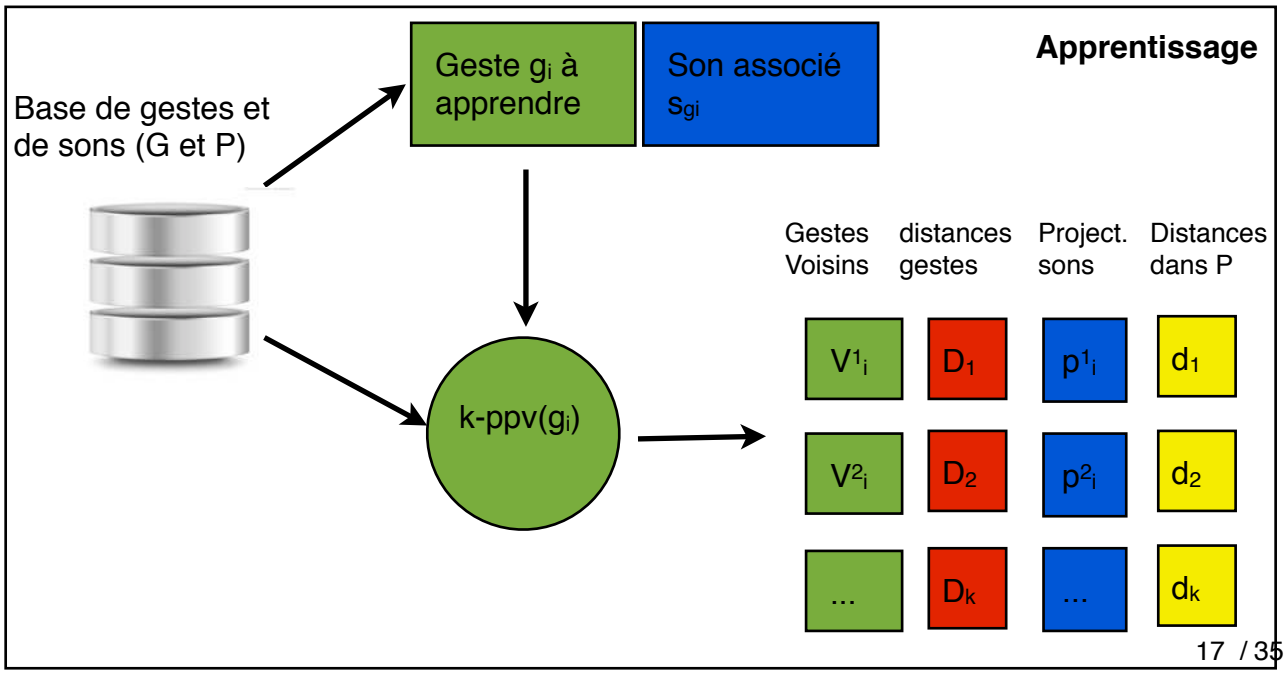


Figure 9 : Requête à la base pour l'apprentissage

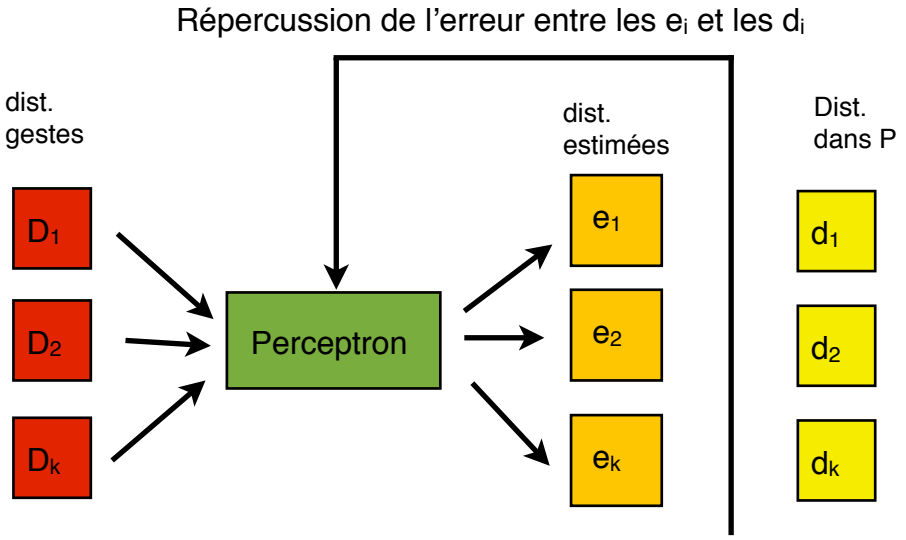


Figure 10 : apprentissage de la correspondance entre G et P

4° - Recherche de la projection du son d'un geste requête

Une fois le *mapping* appris par le perceptron, il est possible d'utiliser la base de données pour effectuer un mapping. Le principe utilisé est très proche de celui de l'apprentissage. Un geste requête R est présenté à la base. Cette dernière calcule les k plus proches voisins dans l'espace des gestes connus et retourne une liste contenant ces derniers ainsi que les distances D_i associées par rapport à la requête et les p_{vi} (figure 11). Ces distances sont présentées aux entrées du perceptron qui retourne des estimations e_i des k distances d_i entre le point p_R du son S_R associé à la requête et les p_i des sons associés aux k plus proches voisins S_{Vi} (figure 12). Comme les coordonnées dans l'espace projeté sont connues, il devient alors possible d'estimer la position dans de p_R en utilisant les distances estimées E_{vi} aux p_{VI} par le perceptron et leurs coordonnées.

Par la suite, rechercher p_{sr}, projection du son associé à la requête, revient chercher le point d'intersection de toutes les hyper-sphères de centre p_i et de rayon $e^l_i(p_{gi}, p^l_i)$ distance estimée par le perceptron entre les sons s_{gi} et s^l_i dans P (figure 13). Ceci étant équivalent à résoudre le système d'équations 5.

$$E_1^2 = ((p_R[1] - p_1[1])^2 + ... + (p_R[k] - p_1[k])^2)$$
$$...$$
$$E_k^2 = ((p_R[1] - p_k[1])^2 + ... + (p_R[k] - p_k[k])^2)$$

Equation 5 : Recherche de I_R

La difficulté réside ici dans le fait que les valeurs e_i ne sont pas des valeurs exactes, mais des valeurs approchées des distances réelles. L'approche que nous proposons ici prend en compte ce problème. L'idée est de trouver un point pour lequel la somme des erreurs entre les et les distance entre p_{sr} et les p^r_i est la plus faible possible. Pour ce faire, nous utilisons un principe similaire à celui utilisé lors de la création de l'espace image des sons :

Un point est tiré avec des coordonnées aléatoires dans P. Puis, jusqu'à ce que l'erreur se stabilise, ces coordonnées sont corrigées de la même manière que lors de la création de l'organisation de P, sauf que cette fois-ci, seul le nouveau point est déplacé. Ce point est

alors considéré comme la projection p_{sr} du son s_r de la requête. Il devient alors possible d'utiliser cette projection par exemple pour estimer des paramètres de synthèse ou rechercher l'élément p_i le plus proche dans la base afin de jouer le son s_{gi} associé.

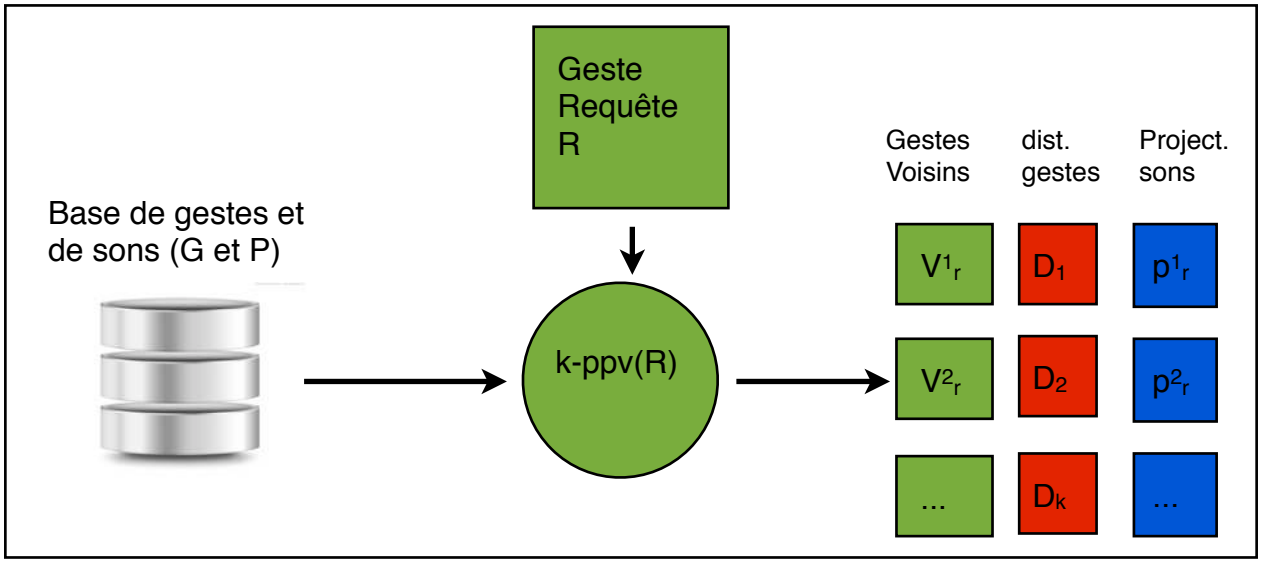


Figure 11 : Requête à la base : calcul des k plus proches voisins

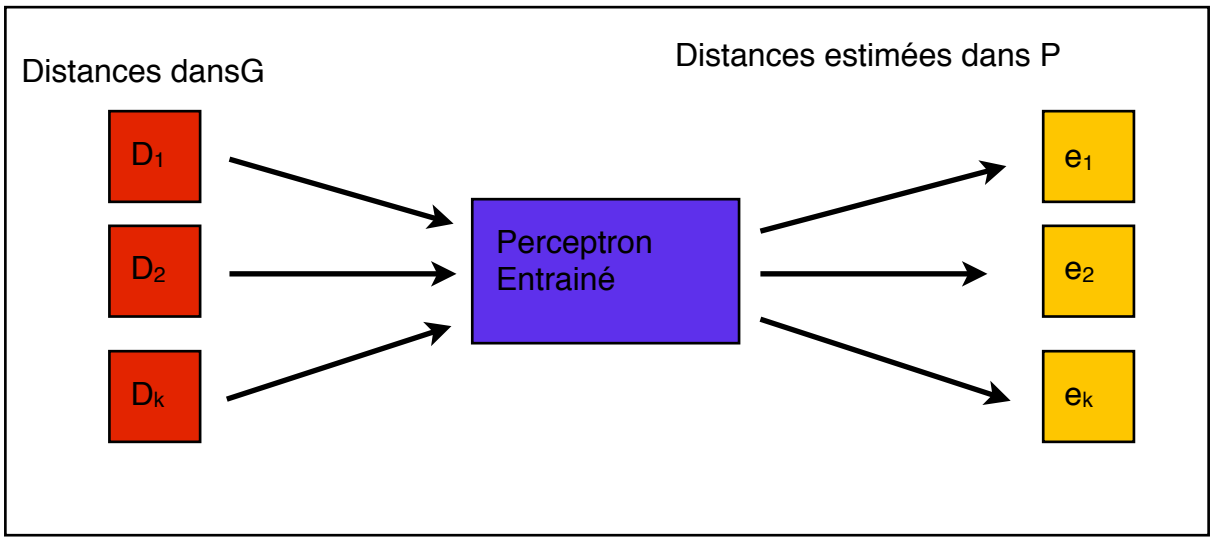


Figure 12 : Réponse du perceptron lors d'une requête

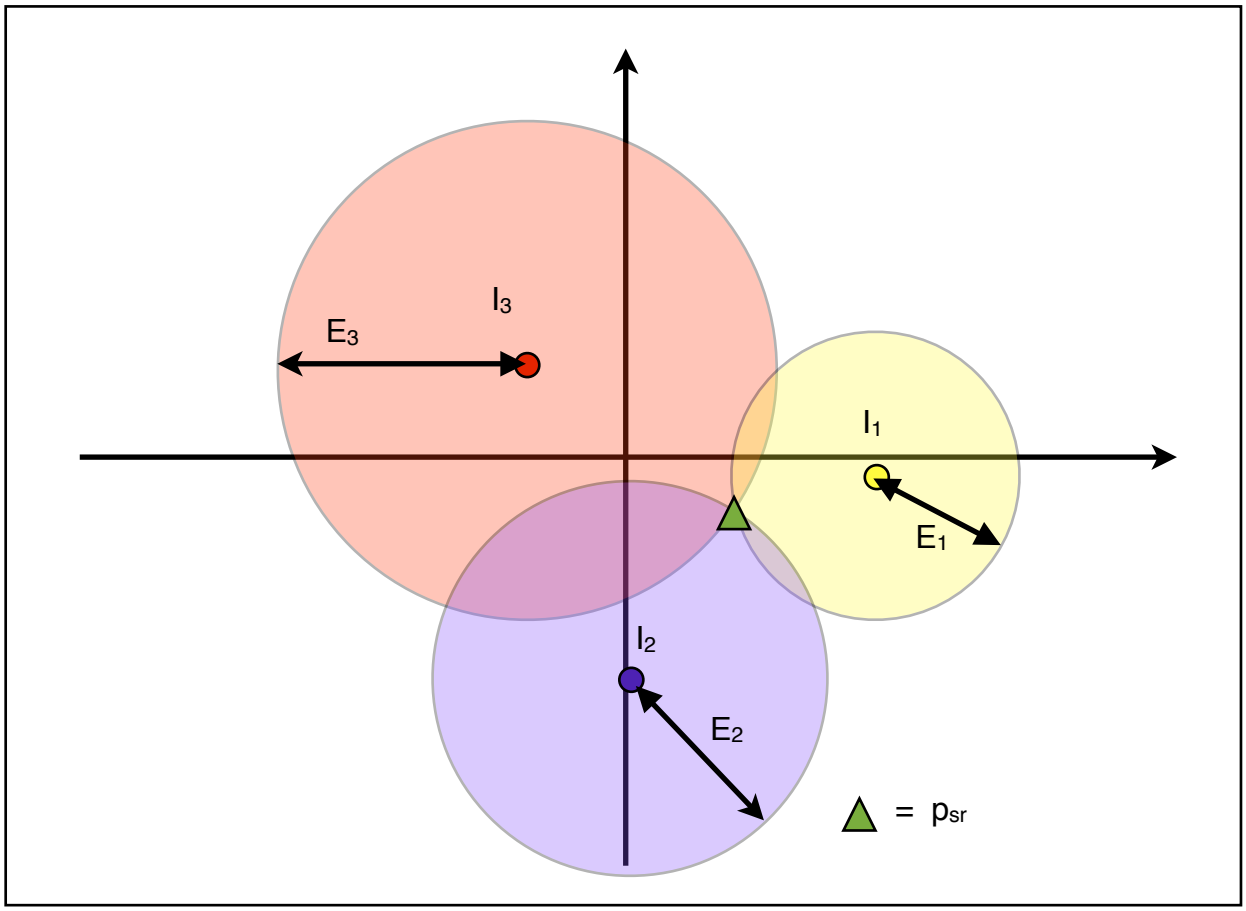


Figure 13 : estimation de p_{sr} avec n=2

III - Expérimentations

1° - Protocole expérimental

a - Généralités

Les expérimentations ont étés effectuées sur une base de données contenant des gestes de jeu d'un timbalier. Les données de gestes sont au format BVH (*Biovision Hierarchy*), échantillonnées à 250 hertz et ont étés obtenues via un système Vicon 460. Parallèlement à la capture de mouvements, les sons du jeu ont été enregistrés au format WAV échantillonnés à 44000 hertz. Ces sons sont représentés en PCM sur 16 bits (*Pulse Code Modulation*). A chaque échantillon correspond donc un son précis appartenant à un ensemble de 2^{16} sons possibles. Par la suite, nous appellerons courbe d'un son, l'évolution du signal sonore au cours du temps.

b - Jeu de timbale

Les timbales sont des instruments à percussion, constitués d'un fût de cuivre recouvert d'une peau. La peau est frappée par des maillets. Elle peut être frappée à différents endroits pour obtenir des sons différents (figure 15). Il est également possible d'ajuster la manière de frapper la peau (force, angle etc...) par des effets de jeu pour moduler encore cette sonorité.

Figure 14 : un ensemble de timbales (source Wikipédia)

c - Base de données utilisée

La base de données utilisée a été réalisée dans le laboratoire IDMIL (www.idmil.org) de l'université de McGill au canada. Elle contient différentes combinaisons de zones de frappe (Centre, tiers et cercle) et de d'effet de jeu (Legato, tenuto, accent, vertical accent et staccato). Ceci pour les deux mains du timbalier. Elle est composée de trois cent cinq couples geste/sons.

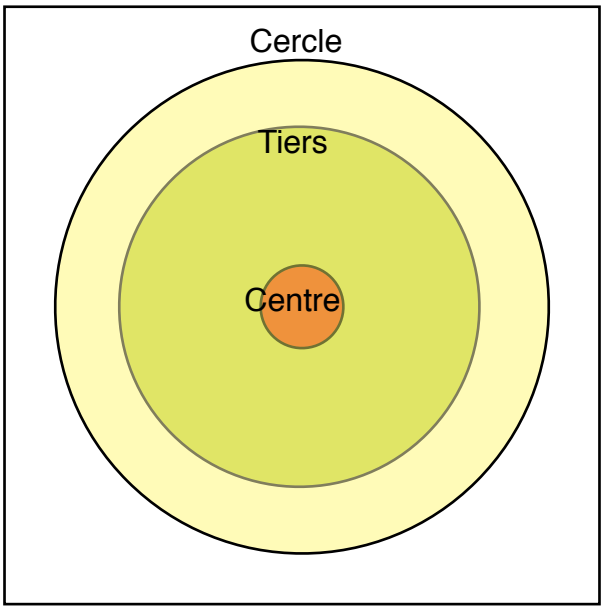


Figure 15 : Zones de frappes de la peau de la timbale

Dans le cadre des expérimentions, deux autres bases ont été mises en place en utilisant des sous ensembles de la base principale. La base Legato ne contient que des gestes de type Legato mais utilisant les trois zones de frappe, elle est composée de soixante couples gestes/sons. La base Center contient uniquement des frappes au centre de la timbale, mais pour les cinq types d'effets de jeu. Elle est composée de 100 couples gestes/sons. Dans chaque cas, deux tiers de la base sert à l'apprentissage du *mapping* et un tiers aux tests.

c - Comparaisons

Lors de l'apprentissage ou du *mapping*, il est nécessaire de calculer des distances entre les sons et entre les gestes. Ces distances sont pré-calculées puis stockées dans la base de manière à accélérer le processus de mapping.

Dans le cas des sons nous utilisons une comparaison des courbes de sons basée sur une enveloppe minimum englobante (MBE : *minimum bounding envelope* [2]). L'utilisation de ce procédé n'est possible que parce que la forme générale de tous les sons de la timbale est la même (figures 16 et 17). Les paramètres changeants entre chaque sons sont la hauteur maximum du signal et la longueur des zones d'attaque et d'affaiblissement. La distance obtenue est égale à la différence entre le deux surfaces. La complexité du calcul de la distance est en O(2n), et elle permet de s'affranchir d'une maximisation de l'erreur entre les deux courbes due au caractère sinusoïdal des signaux étudiés.

Les sons utilisés dans la base de données possèdent le défaut d'avoir étés découpés à la main dans des séries de plusieurs frappes. Afin de corriger les problèmes liés aux décalages temporels induits par les erreurs d'estimation de début et fin de son, ainsi que le non-alignement de ces sons, la MBE a été calculée dans une fenêtre utilisant comme repère la valeur maximum du signal et possédant. Après étude des données, nous avons choisit une fenêtre laissant 800 échantillons à la zone d'attaque et 14 000 à la zone d'affaiblissement (voir figure 16).

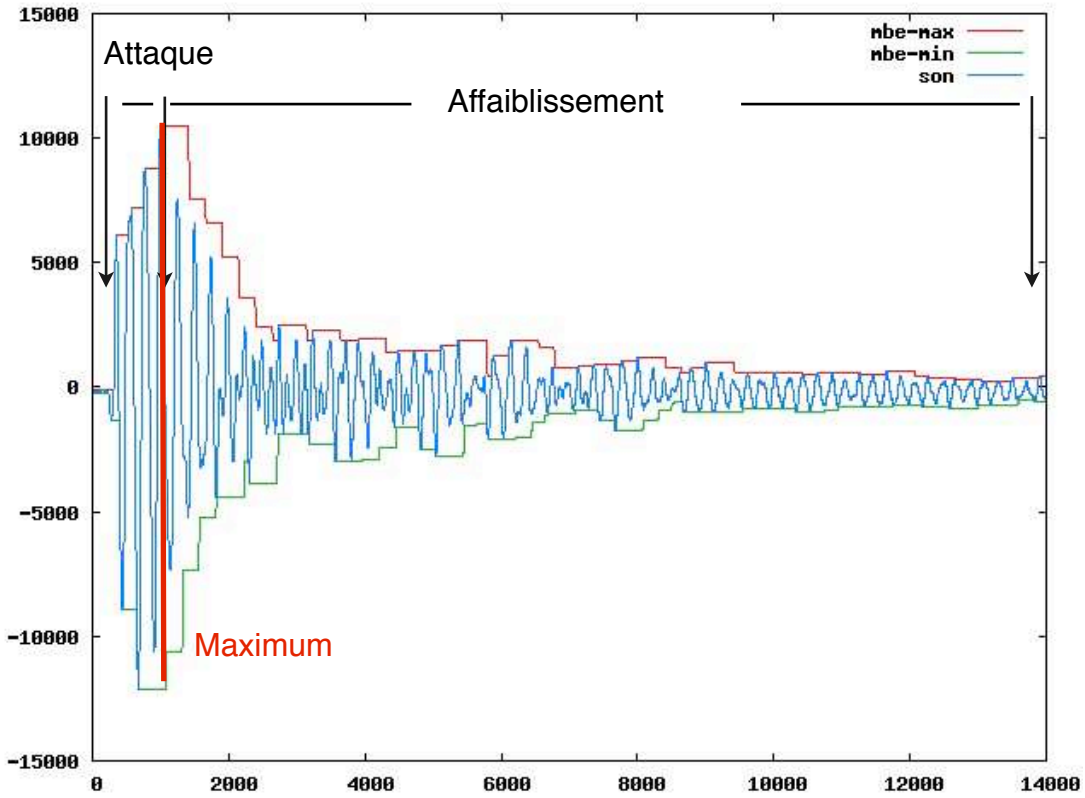


Figure 16 : MBE d'un son de timbale

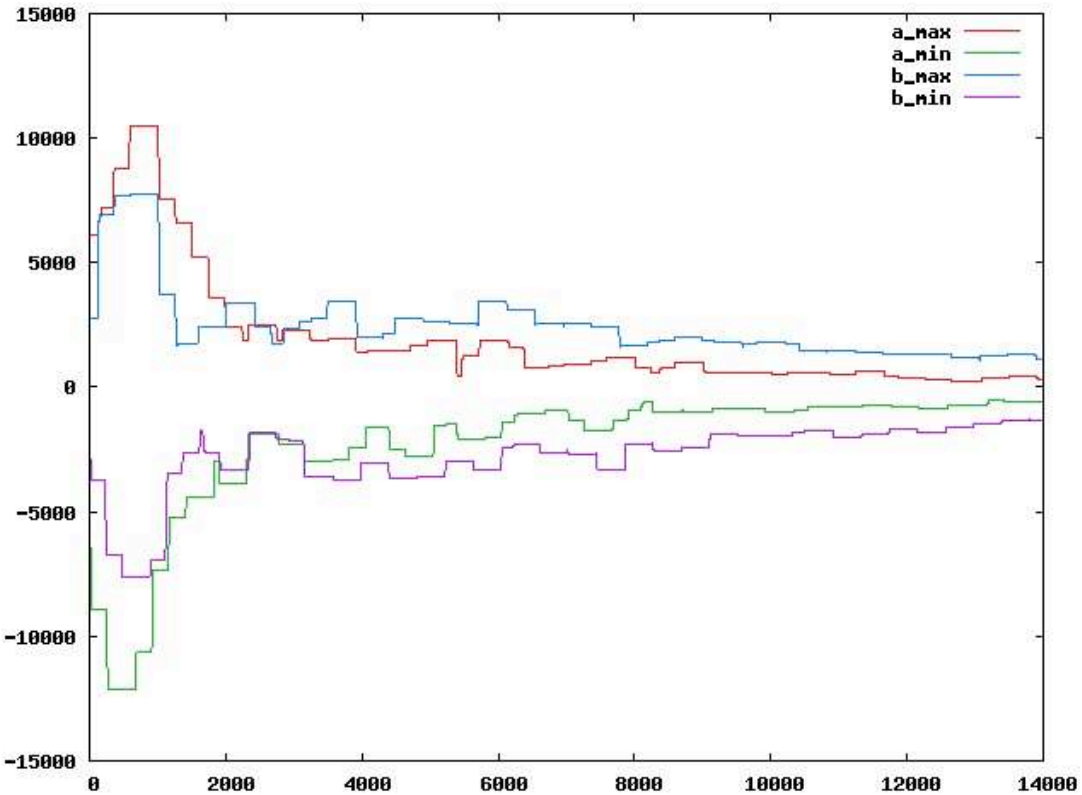


Figure 17 : Comparaison de deux sons a et b en utilisant leurs MBE

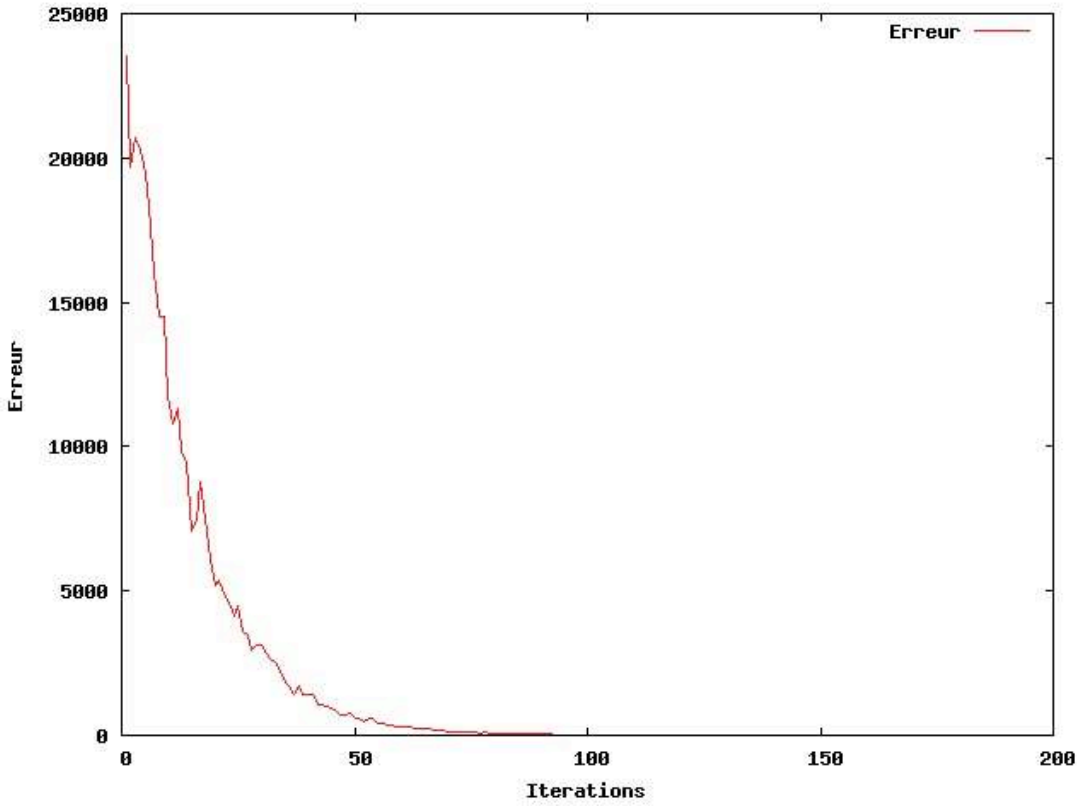


Figure 18 : Exemple d'évolution de l'erreur de distance entre l'espace d'origine et l'espace image en fonction du nombre d'itérations (3 dimensions)

Dans le cas des gestes, nous utilisons la somme des DTW de 18 des séries composant les données de captures. Dans les fichiers de captures, les gestes sont représentés par 69 séries temporelles échantillonnées à 250 hertz représentant chacune un degré de liberté d'une articulation (rotation suivant un des trois axes). De façon à ne conserver que les données du geste musical, nous n'utilisons que les informations concernant les poignets, les coudes et les épaules du musicien.

2° - Dimensionalité de l'espace de projection

Lors de la création de l'espace de projection des sons, un élément important est le choix de la dimensionalité de ce dernier. Nous avons voulu estimer la pertinence du choix de ce nombre de dimensions en fonction de deux critères : l'erreur de distance par rapport à l'espace initial et le nombre d'itérations pour parvenir à une stabilisation de cette erreur. Nous avons effectué les tests sur un ensemble d'une quarantaine de sons correspondant à des frappes au centre, au tiers et sur le cercle de la timbale.

D'après la figure 20, nous pouvons remarquer qu'au delà de quatre dimensions, l'erreur n'évolue plus réellement et qu'il n'est donc pas nécessaire de dépasser cinq dimensions. En ce qui concerne le nombre d'itérations, et d'après la figure 21, il semble se stabiliser à partir de trois dimensions. Le choix du nombre de dimensions de l'espace des sons projetés peut donc être guidé par ces deux critères. Il est cependant probable qu'utiliser un autre calcul de distance entre les sons pourra potentiellement modifier ces résultats.

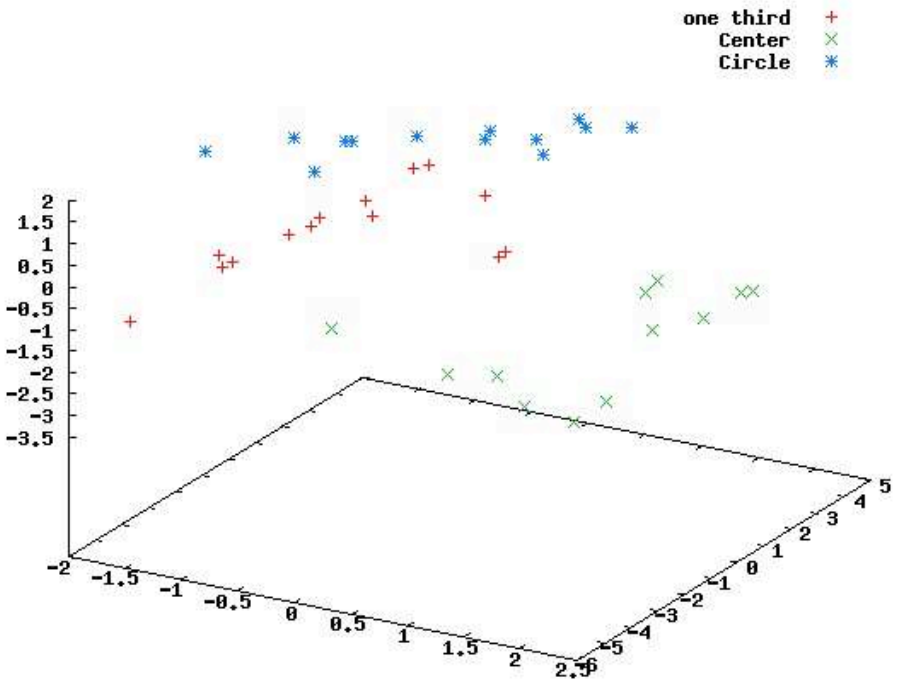


Figure 19 : Exemple d'état de l'espace image après stabilisation (3 dimensions)

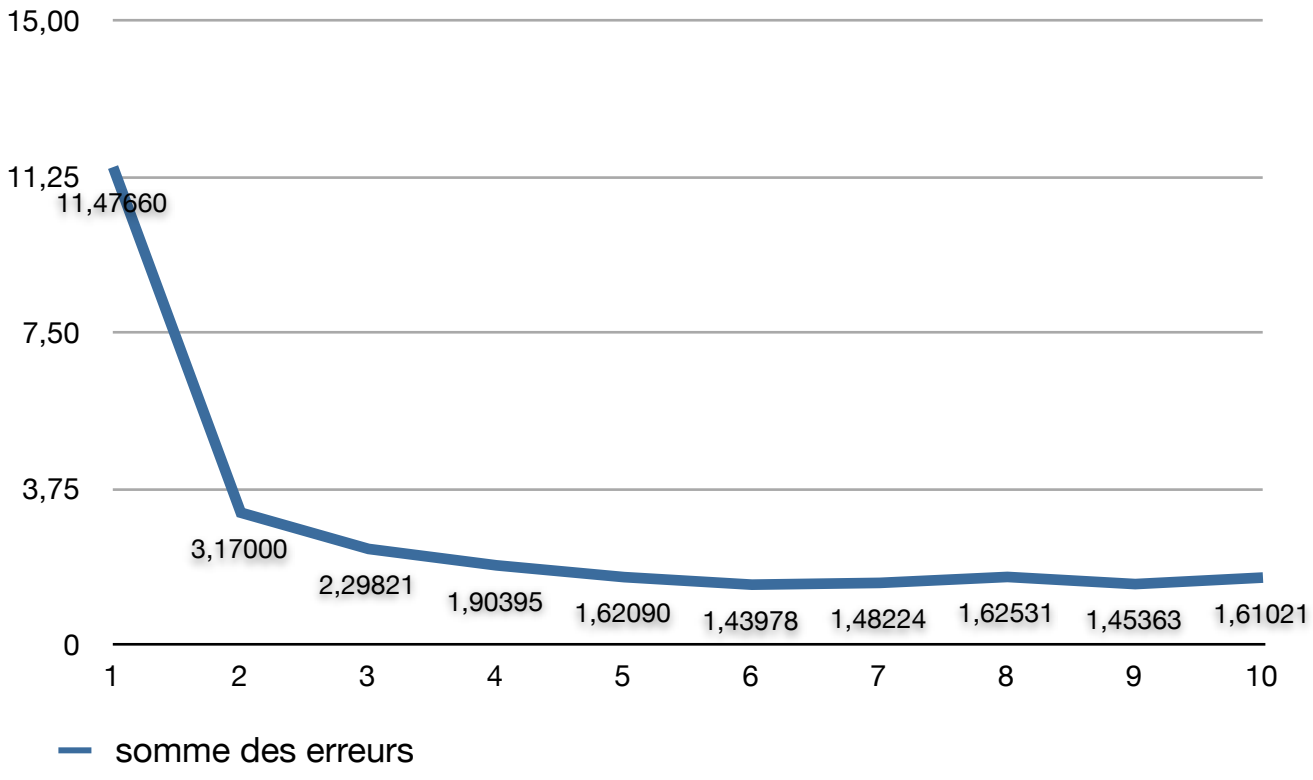


Figure 20 : Evolution de l'erreur entre l'espace initial et l'espace image en fonction du nombre de dimensions

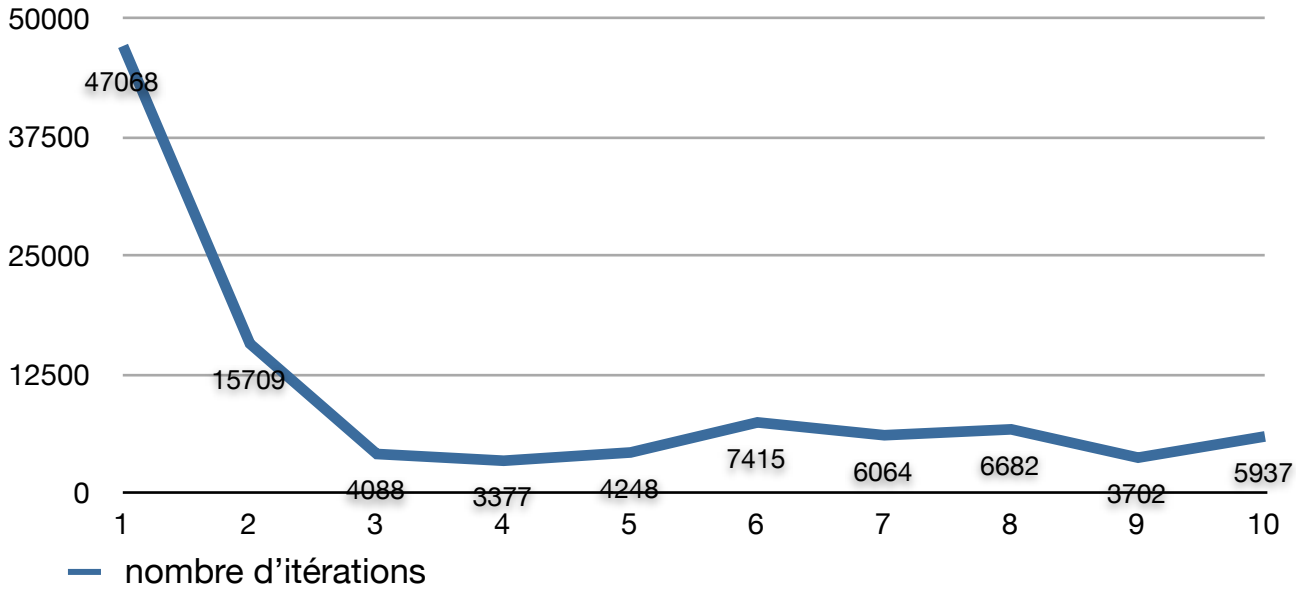


Figure 21 : Evolution du nombre d'itérations avant stabilisation de l'espace en fonction du nombre de dimensions

3° - Estimation de l'efficacité du mapping

Une fois les distances calculées et l'espace des sons projeté défini, il est possible d'utiliser la base de données de test pour effectuer le *mapping*. Pour estimer son efficacité, nous avons utilisé les 3 plus proches voisins de du résultat obtenu pour définir sa zone de frappe ou l'effet de jeu associé.

a - Reconnaissance de la zone de frappe

Dans un premier temps, nous avons voulu estimer la qualité du mapping des gestes avec la zone de frappe. Chaque zone de la timbale définit un son précis. Plus la zone est loin du centre, plus ce son est aigu. Ce changement est celui qui est le plus facilement reconnaissable par l'oreille humaine. Les tests ont été effectués sur la base Legato

Base Legato : nombre de réponses obtenues

requêtes / réponses	Centre	Tiers	Cercle
Centre	5	0	1
Tiers	0	7	0
Cercle	0	0	7

b - Reconnaissance des effets de jeu

L'autre élément important du jeu de timbale est la possibilité d'influer sur la tonalité du son produit en adaptant la manière dont le maillet entre en contact avec la peau de l'instrument. Nous avons pour cela effectué des tests sur deux des bases : la base center et la base complète.

Base Center : nombre de réponses obtenues

	Legato	Accent	Staccato	Tenuto	Vert. Acc.
Legato	0	1	2	1	0
Accent	0	3	1	0	0
Staccato	0	0	3	0	0
Tenuto	0	0	1	3	0
Vert. Acc.	0	0	0	1	3

c - Reconnaissance globale

Lors de l'utilisation de la base générale (contenant des gestes de toutes les combinaisons zone de frappe/effet de jeu) il nous a été impossible d'obtenir un espace de projection des sons stable. Les données étant aléatoirement réparties dans l'espace et l'erreur entre les distances de l'espace S et celles de P ne réduisant pas au fur et à mesure des itérations. Une fois l'apprentissage du perceptron effectué, les réponses de ce dernier ne correspondent pas aux requêtes.

d - Discussion

Dans le cadre d'une reconnaissance de la zone de frappe seule, la méthode proposée semble plutôt efficace avec un taux de reconnaissance de 95% dans nos tests. La reconnaissance des effets de jeux est moins probante avec une reconnaissance de 63% des gestes. Cependant, seuls les gestes de type legato semblent réellement poser un problème. Il est possible que cela vienne du fait que ce geste est celui de base du jeu de timbale et qu'il comporte donc moins de caractéristiques déterminantes que les autres puisque les résultats atteignent 80% sans ce type de gestes. Dans tous les cas l'utilisation d'une base complète prenant en compte les deux paramètres (zone de frappe et effet de jeu) n'a pas été efficace. Il faut donc revoir une partie des outils utilisés. Il est possible de remplacer la distance actuelle utilisée sur les sons par exemple en utilisant des outils d'analyse comme une transformée de Fourier.

IV - Conclusion

Dans ce document nous avons présenté une méthode permettant d'effectuer le *mapping* de gestes musicaux avec des sons. L'implantation de la méthode à été efficace dans des cas particuliers comme comme celui de la reconnaissance de la zone de frappe ou celle du type d'effet de jeu (jeu de timbale), mais elle a montré ses limites lors de la combinaison de ces deux paramètres. L'utilisation d'autres métriques entre les sons et les gestes permettra peut-être de régler le problème. Il est par exemple envisageable d'utiliser des informations comme une transformée de Fourier ou des descripteurs comme ceux proposés pour mpeg-7 dans [20]. D'autres domaines d'application de la méthode pourraient aussi être envisagés dans le but de faire correspondre d'autres paires de séries temporelles. Enfin, des expérimentations sur la pertinence de l'utilisation du perceptron dans ce contexte peuvent être effectuées.

Annexes

1° - Techniques d'analyse de données

a- Calculs de distances

Le calcul de la distance entre deux objets est important dans toutes méthodes de reconnaissance et de recherche. Le calcul de la distance permet de mesurer la similarité entre deux éléments d'un même espace. Plus cette distance est proche de 0, plus les deux éléments sont similaires. Au contraire, plus cette distance est élevée, plus les éléments sont différents. Le calcul de distance est en général assez lourd, il vaut donc mieux le réserver uniquement aux quelques données proches de notre requête, et pour ça l'utilisation des indexes est donc primordiale dans les bases de grosses et/ou nombreuses données.
Il existe de nombreuses méthodes de calculs de distances. La plus connue étant certainement la distance euclidienne. Cette distance est cependant un cas particulier d'un ensemble de mesures appelé p-norms. Les p-norms sont les calculs de distances issus de l'équation A-1 :

Equation A-1 : Pour deux vecteurs v_a et v_b de dimension n :

$$p\text{-}norm(v_a, v_a) = \sqrt[p]{\sum_{i=1}^{n}(v_{ai}-v_{bi})^P}$$

Ces normes s'appliquent bien dans des cas où le calcul de distance peut s'effectuer entre des vecteurs de dimension fixe, mais il devient impossible de les utiliser sur des vecteurs de tailles différentes comme ce peut être le cas entre deux série temporelles. Il est, par exemple, possible de calculer la distance entre deux mouvements en comparant directement les différentes séries temporelles composant ces mouvements. Dans cet optique, l'utilisation de distances comme la déformation temporelle dynamique (DTW, *Dynamic Time Warping*) ou le calcul de la sous-séquence commune la plus long (LCSS, *Longest Common Subsequence*) de travailler avec des séries temporelles de taille différentes. L'avantage est que si un mouvement est effectué légèrement plus vite ou moins vite que la référence dans la base, ce type de mesure de distance permettra tout de même de les trouver similaires. Vous pouvez vous référer à [11] pour obtenir plus d'information concernant le calcul de distance et la comparaison de séries temporelles.

b - Décomposition en valeurs singulières (SVD)

La décomposition en valeur singulière (SVD, *Singular Value Decomposition*) permet d'exposer la structure géométrique d'une matrice, afin de trouver la dimension qui possède la plus grande variance d'après les vecteurs de cette matrice. Pour la matrice A de dimensions m×n, nous obtenons les matrices $U=[u_1,...,u_n]$ de dimensions m×m et $V=[v_1,...,v_n]$ (de transposée V^t) de dimensions n×n deux matrices orthogonales et D une matrice diagonale dont les valeurs $[d_1,...,d_n]$, organisées tel quel que $d_1 \geq d_2 \geq ... \geq d_n$ sont les valeurs singulières de la matrice A (Equation A-2 et Figure A-1).

Ces valeurs singulières pondère les vecteurs de V tel que V_1 soit plus important que V_2 et ainsi de suite. V_1 représente donc l'axe sur lequel les données de A ont la plus grande dispersion (projection sur l'axe ayant la plus grande p-norme).

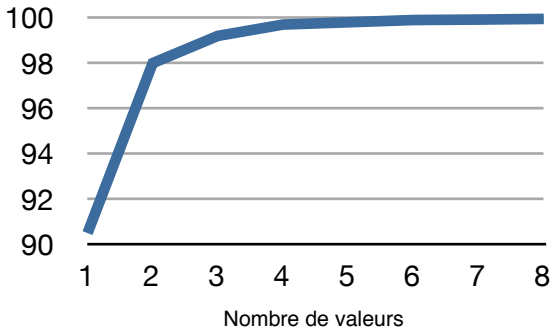


Figure A-1 : Importance cumulée des valeurs singulières

Le défaut du calcul de la SVD est qu'il est d'une complexité élevée : $O(4mn^2+8n^3)$ pour le seul calcul de D et V. Même si cela peut être amoindri par une des propriétés de la SVD qui ramène à un calcul sur une matrice carrée n×n (Equation A-3), et puisque généralement m >> n le temps de calcul s'en trouve largement réduit ($O(12n^3)$). Par exemple pour une matrice de capture de mouvement de la main de 5 secondes à 30 Hz, n = 22 et m = 150. Le calcul par l'équation 6 se fera en 375584 itérations et par l'équation 7 en 127776 itérations.

Equation A-2 : SVD de la matrice A

$$A=U\,D\,V^t$$

Equation A-3 : SVD de la matrice $A^t A$

$$A^t A = V\,D^2\,V^t$$

Pour plus d'informations sur la SVD, un tutoriel est accessible [12].

c - Analyse discriminante linéaire (LDA)

L'analyse discriminante linéaire (LDA, *Linear Discriminant Analysis*) permet de maximiser la distance entre les classes de données et de minimiser la distance entre les

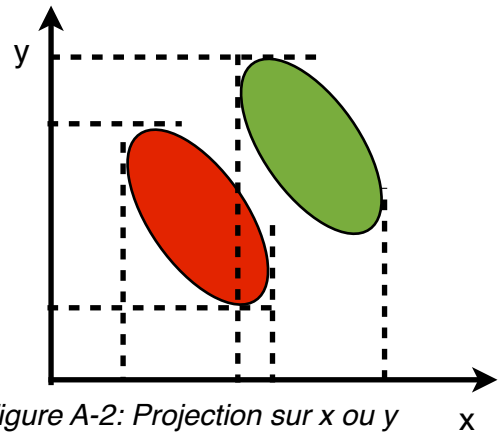


Figure A-2: Projection sur x ou y

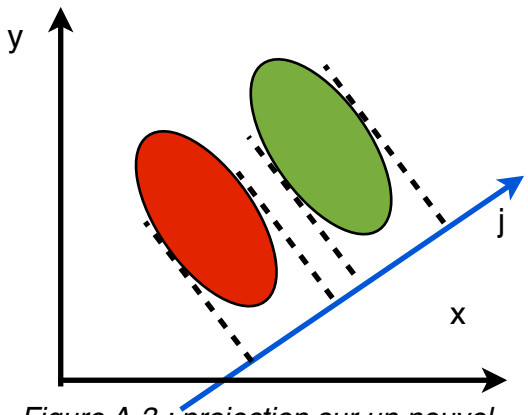


Figure A-3 : projection sur un nouvel axe j

éléments d'une classe tout en réduisant la dimension des données. La méthode la plus connue de LDA utilise le discriminant de Fisher, qui permet de projeter des données d'un espace à n dimensions dans un espace à n-1 dimensions les figures 5 et 6 montrent un exemple d'utilisation du discriminant de Fisher sur deux classes dans un espace à deux dimensions. Sur la figure A-2 les deux classes se chevauchent, rendant la classification d'un nouvel élément difficile. Alors que sur la figure A-3 les deux classes sont bien réparties sur une nouvelle dimension.

d - algorithme k-nn *(K Nearest Neighbors)*

Cet algorithme utilise les k données les plus proches d'un élément pour lui définir une classe. La difficulté étant de choisir judicieusement la valeur de k. La figure A-4 représente un exemple d'utilisation de l'algorithme.

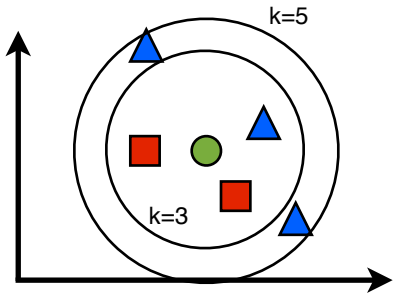


Fig. A-4 : Algorithme k-nn
si k = 3 est un
si k = 5 est un

e - algorithme k-means

L'algorithme k-means fonctionne suivant le principe suivant : k nouveaux points sont tirés au hasard dans l'espace des données (dans notre cas l'espace à n dimensions). Chaque point de nos données est alors associé au point appartenant aux k nouveaux qui est le plus proche, puis chacun des k points est déplacé au barycentre de l'ensemble qui lui est associé. Le processus (figure A-5) recommence en boucle jusqu'à stabilisation. Afin d'éviter les extremums locaux, l'algorithme est effectué plusieurs fois.

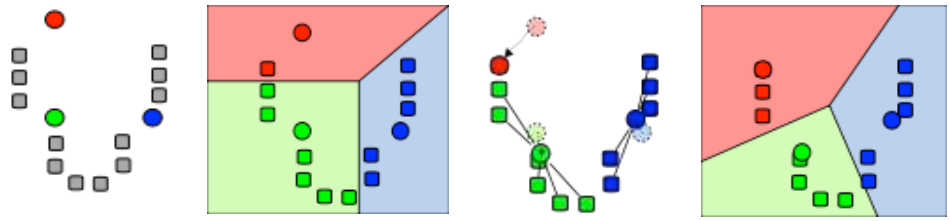


Figure A-5 : Exemple d'exécution d'un pas de k-means pour k=3 (source wikipedia)

f - Indexes multidimensionnels

Les indexes multi-dimensionnels sont des moyens efficaces de palier à lourdeur du traitement de données dans les espaces a plusieurs dimensions. Ils prennent la forme

d'arbres qui organisent les données dans l'espace afin que, lors de la recherche, à chaque niveau de l'arbre un grande partie des données non-pertinentes soit éliminée.
L'exemple type d'index multi-dimensionnel est le R-Tree d'Antonin Guttman [6]. Un R-Tree découpe l'espace des données en sous-espaces contenant un sous-ensemble de ces données. A chaque insertion, un de ces espaces est re-dimensionné pour accueillir le nouvel élément. Lorsqu'un sous-espace atteint un nombre d'éléments définit, il est lui même découpé en deux sous espaces en utilisant les deux éléments les plus éloignés comme repères. Si le nombre de sous espaces dépasse le nombre définit, un nouveau sous-espace, sous la forme d'un nouvel étage, est ajouté à l'arbre sur le même principe. Une fois les données indexées (figure A-6), il est possible d'effectuer dans l'arbre des recherches de type «contient» qui retourne les données qui contiennent entièrement la requête ou de type «intersecté avec» qui retourne tous les éléments qui ont une zone commune avec la requête.

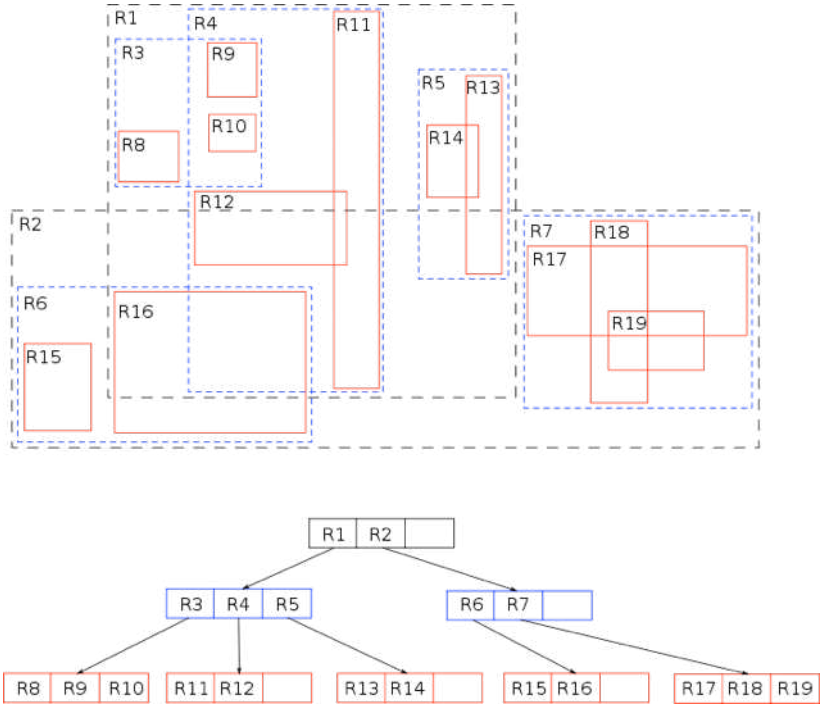


Figure A-6: Exemple d'un R-Tree dans un espace à deux dimensions (source wikipedia)

g - Perceptrons multicouche

Un percetron est un type de neurone artificiel composé de n entrées E_n, d'une sortie S et de valeurs P_n pondérant chacune des entrées. Le résultat de la sortie S=f(x) tel que x soit la somme des entrées pondérées et f une fonction mathématique (en général une

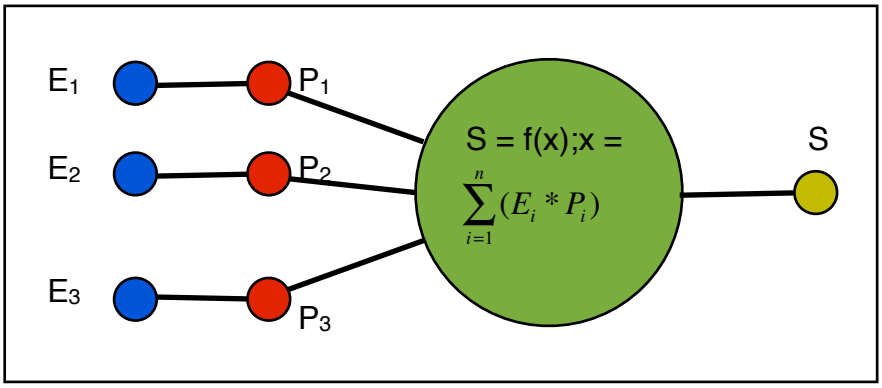


Figure A-8 : Exemple de Perceptron simple

sigmoïde). L'apprentissage s'effectue en corrigeant les pondérations des entrées en fonction de l'erreurs entre le résultat de la sortie et la sortie attendue.

Le perceptron multicouche est une forme de réseau de neurones artificiel comportant n entrées et m sorties. Le principe est d'associer en parallèle et en série des neurones artificiels de type perceptron. la méthode d'apprentissage est la même que celui du perceptron. Les pondérations des entrées des différents perceptrons sont corrigés en fonction de l'erreur entre les résultats des sorties et les résultats attendus. L'algorithme le plus utilisé pour effectuer ces corrections est celui de rétro-propagation du gradient.

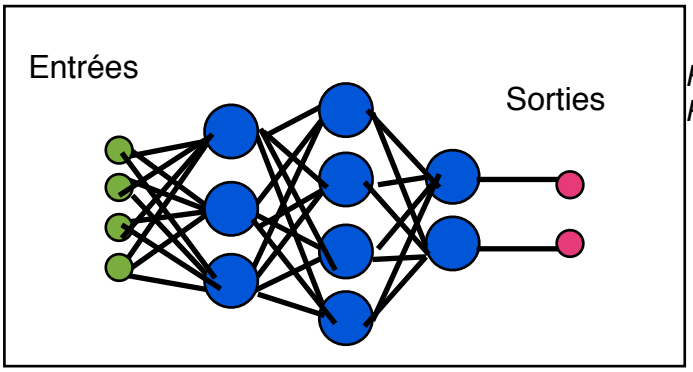


Figure A-8 : Exemple de Perceptron multicouches

De nombreux ouvrages [21] ou d'articles de recherche traitent de ce domaine.

Bibliographie

•[1] «Gesture Recognition : A Survey» Sushmita Mitra, Tinku Acharya. IEEE Transactions on systems, man, and cybernetics - partC : applications and reviews, vol. 37 n°3, may 2007

•[2] «Indexing Multi-Dimensional Time-Series with Support for Multiple Distance Measures» Michail Vlachos, Marios Hadjieleftheriou, Dimitrios Gunopulos, Eamonn Keogh. KDD ’03 Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, 2003

•[3]«Indexing of Motion Capture Data for Efficient and Fast Similarity Search», Chuanjun Li, B. Prabhakaran. Journal of computers, Vol. 1, N°3, june 2006

•[4] «Continuous Realtime Gesture Following and Recognition», Frédéric Bevilacqua, Bruno Zamborlin, Anthony Sypniewski, Norbert Schnell, Fabrice Guédy, Nicolas Rasamimanana. Gesture in embodied communication and human-cumputer interaction, 2010

•[5] «Gesture Analysis of Violin Bow Strokes», Nicolas Rasamimanana, Emmanuel Fléty, Frédéric Bevilacqua. Gesture in Human-Computer Interaction and Simulation 6th International Gesture Workshop, may 2005

•[6] «R-Trees : a dynamic index structure for spacial searching», Antonin Guttman. SIGMOD '84 Proceedings of the 1984 ACM SIGMOD international conference on Management of data, 1984

•[7] «Robust Fisher Discriminant Analysis», Seung-jean Kim, Alessandro Magnani, Stephen P. Boyd.Advances in neural information processing systems. Vancouver and Whistler, British Columbia, Canada. 2005

•[8] «A State-Based Approach to the Representation and recognition of Gesture», Aaron F. Bobick, Andrew D. Wilson. IEEE Transaction on pattern analysys and machine intelligence, vol. 19, n° 12, december 1997.

•[9] «On the beat : Human movement and timing in the production and perception of music», sofia dahl. Doctoral thesis in speech and music communication, 2005

•[10] «A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition», Lawrence R. Rabiner. Proceedings of the IEEE, vol. 77, N°2, february 1989

•[11] «A Tutorial on Indexing and Mining Time Series Data», Eamonn Keogh. The 2001 IEEE International conference on Data Mining, november 2001

•[12] «SVD and LSI Tutorial», E. Garcia. http://www.miislita.com/, 2007

•[13] «Fuzzy sets», L.A. Zadeh,.Information and Control, Vol. 8, Issue 3, Pages 338-353, june 1965

•[14] «Implementing the Viterbi algorithm», Hui-Ling Lou. IEEE, Signal Processing Magazine Vol.12, Issue 5, Page 42, september 1995

•[15] «Virtual Gesture Control of Sound Synthesis : Analysis and Synthesis of Percussion Gestures», Alexandre Bouënard, Marcelo M. M. Wanderley, Sylvie Gibet. Acta Acustica united with Accustica, Vol. 96, February 2010

•[16] F. Richard Moore. «The Dysfunction of MIDI», in Computer Music Journal, Vol. 12, N °1, Spring 1988, MIT

•[17] J.B. Rovan, M.M. Wanderley, S. Dubnov, P. Depalle. «Instrumental Gestural Mapping Strategies as Expressivity Determinants in Computer Music Performance» in Proc. of the Kansei Workshop, Genova, p. 68-73. 1997

•[18] J. Malloch, S. Sinclair, M.M. Wanderley. «A Network-Based Framework for Collaborative Development and Performance of Digital Musical Instruments». Lecture Notes in Computer Science, 2008, Volume 4969/2008, p.401-425

•[19] T. Kohonen, «The Self-Organizing Map», Proceedings of the IEEE, vol. 78, N°9, p1464-1480, septembre 1990

•[20] G. Peeters, S. Mc Adams, P. Herrera, «Instrument Sound Description in the Context of MPEG-7»Proceeding of ICMC2000 (International Computer Music Conference), Berlin, Germany, August 27th - September 1st, 2000

•[21] M. Minsky, S. Papert, «Perceptrons : an introduction to computational geometry», Cambridge, MA : MIT, 1988