Synchronisation sensorimotrice sonore : vers une action conjointe musicale entre humain et agent virtuel

Maël Donnard, Gireg Desmeulles, Elisabetta Bevacqua

Synchronisation sensorimotrice sonore : vers une action conjointe musicale entre humain et agent virtuel

Maël Donnard
Gireg Desmeulles
Elisabetta Bevacqua
desmeulles@enib.fr
bevacqua@enib.fr
Lab-STICC, ENIB
Brest, France

RÉSUMÉ

La coordination des mouvements entre les individus est un aspect essentiel du comportement humain. Cela peut aller de la synchronisation des mouvements dans des situations quotidiennes de coordination sociale à des activités plus complexes telles que le sport ou la musique, qui exigent des compétences particulières. Nous avons étudié cette synchronisation entre un agent virtuel et un humain en mettant en œuvre une interaction musicale rythmique simple, mais de haute précision temporelle : taper ensemble dans les mains. Cet article présente notre modèle et son implémentation qui s'appuie sur des outils standards d'animation 3D, ainsi qu'une expérimentation préliminaire. Le modèle est basé sur le concept de boucle sensorimotrice et le système résultant permet à un utilisateur et un agent virtuel de taper dans les mains ensemble. L'expérimentation préliminaire montre que le système est capable de tenir un tempo fixe de façon fiable, de s'adapter au tempo de l'utilisateur et de guider l'humain vers un tempo cible définit au lancement de l'interaction.

CCS CONCEPTS

• **Human-centered computing** → **Sound-based input / output**;
• **Applied computing** → *Performing arts.*

KEYWORDS

Agents virtuels, interaction temps réel, boucle sensorimotrice

1 INTRODUCTION

Le travail présenté ici s'inscrit dans la suite d'un projet art et science [1, 2] qui visait à proposer à des musiciens de jouer des percussions corporelles conjointement avec un agent virtuel. L'agent pouvait jouer et danser des phrases rythmiques en boucle. S'il percevait un appel musical [1] joué par un des musiciens, l'agent changeait de phrase en s'adaptant au tempo de l'appel qu'il venait de reconnaître. Du point de vue du spectateur, la proposition semblait fonctionner, mais du point de vue des musiciens, l'interaction restait limitée. En effet, l'agent virtuel ne possédait pas de mécanisme de boucle sensorimotrice. En conséquence, la compensation des latences dues au logiciel et au matériel ainsi que tout le travail de synchronisation musicale fine entre deux appels étaient portés par les musiciens. Le travail présenté dans cet article vise à aller plus loin dans l'interaction musicale rythmique entre un agent virtuel et un humain en proposant une architecture de boucle sensorimotrice pour l'agent virtuel permettant synchronisation temporelle très fine. Il le fait de façon très modeste, car nous nous intéressons à la phrase rythmique la plus simple qui puisse être : taper dans les mains ensemble.

La mise en ouvre de cette synchronisation de la boucle sensorimotrice entre un agent virtuel et un humain constitue à notre sens une brique importante pour des applications d'activité sociale musicale. Ce type d'activité prend de plus en plus d'ampleur dans les environnements virtuels, allant de la composition à la performance en passant par la pratique récréative, l'enseignement et même l'expérience de concerts virtuels en direct. Ce champ de possibilités a des implications non seulement au niveau artistique, mais aussi au niveau commercial [13].

La synchronisation des mouvements entre les individus est un aspect fondamental des comportements humains. Les exemples de synchronisation vont de situations quotidiennes de coordination des mouvements dans des contextes sociaux, à des situations plus avancées dans des activités demandant certaines compétences comme le sport ou la musique [8]. Étudiée notamment dans [10], la synchronisation sensorimotrice est un phénomène qui caractérise certaines interactions impliquant une synchronisation des mouvements entre plusieurs personnes. Le terme de synchronisation sensorimotrice désigne une coordination rythmique de la perception et de l'action. Contrairement à de nombreux mécanismes du mouvement humain qui sont des réflexes, des réponses réactives à un évènement, la synchronisation sensorimotrice implique une intention. L'individu doit de manière active prédire l'évènement futur (par exemple le moment où tombe la pulsation selon le rythme). Ce mécanisme implique une réponse temporisée, qui a pour but de coïncider avec l'évènement suivant, ce qui distingue la synchronisation sensorimotrice d'une simple tache de réaction où la réponse est effectuée le plus rapidement possible. Dans la littérature, la synchronisation sensorimotrice est souvent étudiée dans une configuration la plus simple possible, consistant en un individu essayant de se synchroniser en tapant un rythme avec le doigt de manière régulière, sur des séquences de stimuli auditifs [5, 11]. Pour effectuer une synchronisation avec un stimulus externe, l'humain effectue, plus ou moins consciemment, plusieurs ajustements au cours de la tâche. On parle de correction de phase et de correction de période, effectuées d'un *tap* à l'autre lors de la synchronisation, si l'humain détecte une imprécision entre son *tap* et le stimulus externe. Ces

1. Un appel musical est une petite phrase, connue de tous, jouée généralement sur une mesure et servant de signal pour déclencher un arrangement, un changement de rythme, etc., qui sera pris en compte par tous les musiciens dès la mesure suivante.

mécanismes sont des mécanismes de base de la synchronisation rythmique sur lesquels notre modèle se base.

Dans la prochaine section, nous présentons plus en détail la boucle sensorimotrice et certaines de ses implémentations dans le domaine des agents virtuels. Les sections 3 et 4 décrivent le modèle de boucle sensorimotrice et son implémentation. Enfin, une brève expérimentation préliminaire, qui nous a permis de recueillir un premier ressenti de l'utilisateur, est présentée en section 5.

2 BOUCLE SENSORIMOTRICE POUR AGENTS VIRTUELS

D'après Hartholt et al. [7], pour qu'un agent virtuel soit efficace, il doit présenter une gamme de capacités simulant celles des humains. Et la capacité à se synchroniser au moyen d'une boucle sensorimotrice est une capacité humaine importante dans notre contexte. Originaire des neurosciences, ce concept désigne une approche du fonctionnement moteur de l'humain qui considère les actions de perception et d'action en tant que boucle [4]. Il s'agit d'un bouclage constant entre l'envoi d'une commande motrice, et le retour sensoriel sur l'état de l'environnement et ses propres actions. Ces retours sensoriels servent ensuite à décider de la prochaine commande motrice à effectuer. Dans le domaine des agents virtuels, nous retrouvons la boucle sensorimotrice dans différents travaux. Prepin et Pelachaud proposent la mise en œuvre d'une boucle sensorimotrice pour modéliser le comportement non verbal d'agents conversationnels dans des situations de dialogue entre eux [9]. Les mouvements sont effectués en fonction des perceptions de l'agent de ses propres mouvements et de son environnement. Les expériences sur ces agents virtuels montrent l'émergence d'une synchronisation des comportements non verbaux lorsque le système simule une compréhension entre les agents. Dans [6, 3], les chercheurs introduisent le concept de couplage sensorimoteur, une mesure de la qualité de l'interaction entre un humain et un agent virtuel impliqués dans une action gestuelle conjointe [2]. Dans la boucle sensorimotrice mise en œuvre, le choix du comportement de l'agent est dicté par la valeur du couplage plutôt que par l'évaluation des actions isolées. Le couplage est calculé comme la distance entre certaines caractéristiques de bas-niveau du comportement de l'humain et de l'agent (vitesse, accélération, amplitude...). Plus cette distance est réduite, plus le couplage est fort. Dans l'application d'un exergame [3] de fitness, le modèle est capable de générer un comportement adaptatif pour l'agent virtuel qui augmente l'engagement de l'utilisateur dans l'interaction ainsi que le sentiment de présence. Bien que les modèles de boucle sensorimotrice pour agents virtuels de Prepin et Pelachaud et de De Loor et collègues soient satisfaisants pour les cas d'application présentés, l'échelle de temps prise en compte pour les réactions est moins exigeante que celle recherchée dans un contexte musical et ils ne prennent pas en compte la modalité sonore. C'est cette modalité que nous nous proposons de mettre en œuvre ici. La perception sonore humaine étant très fine (quelques millisecondes), elle nous offre un défi technique pour atteindre la précision temporelle nécessaire.

2. L'action conjointe est décrit dans [12] comme une action coopérative et coordonnée, dont la qualité repose sur la qualité de sa coordination.

3. An exergame refers to video games that require physical exertion and are considered a form of exercise.

3 MODÈLE DE LA BOUCLE SENSORIMOTRICE

La boucle sensorimotrice pour notre modèle est séparée en plusieurs étapes :

- La perception, l'agent capte des informations sur son environnement et sur lui-même via ses capteurs.
- La décision, l'agent traite les informations qu'il a captées pour déterminer une action à effectuer.
- L'action, l'agent effectue l'action déterminée à l'étape précédente. Cette action aura des conséquences sur l'environnement et sur l'agent lui-même. Ces conséquences vont alors être perçues et prises en comptes pour la suite des actions et ainsi de suite.

3.1 Perception

L'agent virtuel doit pouvoir percevoir son environnement et ses propres actions. Dans le cadre de ce travail, nous nous sommes concentrés sur la perception du son pour détecter les évènements. Nous avons utilisé deux microphones, le premier microphone est dirigé vers l'utilisateur pour écouter les notes qu'il produit, et le deuxième est dirigé vers l'enceinte de l'agent virtuel, pour que celui-ci écoute ses propres notes, percevant ainsi ses propres actions. La perspective de ce travail étant d'avoir une application dans le domaine de la musique, nous appellerons par généralisation « note » les clappements produits par l'agent virtuel ou l'utilisateur. Les dates des notes correspondent aux moments où l'utilisateur ou l'agent virtuel tape des mains. Différentes dates entrent en jeu ici : les dates des notes planifiées par le module de décision (voir 3.2) et celles perçues. Une latence peut être présente entre la date où le système planifie de jouer et où le système perçoit le retour audio de cette note. Le principe d'écouter ses propres actions permet de prendre en compte cette éventuelle latence liée au matériel. Pour manipuler les dates, nous utilisons le temps CPU, d'une précision assez fiable par rapport au temps réel.

Pour déterminer la date d'une note perçue, les signaux des microphones sont transmis à un algorithme de détection d'enveloppe. Le but est de détecter les instants où le volume du son dépasse un certain seuil, ce qui révèle le moment auquel la note est jouée. Cet algorithme utilise le concept de niveau d'énergie : une variable garde en mémoire le niveau d'énergie du signal sonore, augmenté ou diminué en fonction du niveau de chaque nouvel échantillon. Quand l'énergie dépasse un certain seuil, on considère que la date de l'échantillon pris en compte est la date de la note son perçue.

3.2 Décision

Pour avoir un comportement crédible et mettre en place une interaction, l'agent virtuel doit agir en fonction des notes qu'il a entendues. Lorsque son but est de maximiser la synchronisation (diminuer les écarts entre les notes de l'utilisateur et les siennes), il doit adapter ses propres notes à celles de l'utilisateur. Pour cela, un algorithme prenant en compte les différentes dates des notes passées planifie les dates des notes à venir. L'algorithme a ensuite pour rôle de corriger les notes planifiées en fonction du décalage temporel avec l'utilisateur.

3.2.1 L'algorithme de synchronisation. L'algorithme a pour rôle de placer dans le temps les notes qui sont jouées par l'agent virtuel.

Il dispose d'un « intervalle » interne, qui correspond à la distance entre deux de ses notes consécutives, qui détermine donc le tempo actuel (nombre de battements par minute — bpm). Cet intervalle est amené à changer au fur et à mesure de l'exécution, pour s'adapter au tempo de l'utilisateur. Lors d'une interaction, les notes à jouer de l'agent virtuel sont prévues à partir du tempo dicté par les notes de l'utilisateur. L'agent virtuel joue les notes aux moments prévus, en continuant de planifier les suivantes au fur et à mesure. Cependant, pour maintenir une synchronisation avec l'utilisateur qui ne reste pas forcément sur le même rythme au cours de l'interaction, il faut appliquer une correction aux notes prévues. Cette correction à appliquer est calculée à partir de la différence que perçoit l'agent virtuel entre sa propre note jouée et celle de l'utilisateur. Ainsi, l'algorithme cherche à percevoir la prochaine note jouée par l'utilisateur autour de la prochaine date théorique estimée et dans une fenêtre de temporelle (paramétrable en pourcentage) calculée par rapport à l'inverse du tempo. Pour nos cas d'application simplistes, cette fenêtre a été calibrée arbitrairement à 50 % de l'intervalle entre deux battements, soit plus ou moins un quart de temps, ce qui correspond bien à des utilisateurs novices en musique. L'algorithme attend la fin de la fenêtre temporelle de prise en compte de la note n de l'agent virtuel, pour regarder les évènements perçus dans cette fenêtre. Une note perçue hors de cette fenêtre n'aura donc pas d'influence. La note de l'utilisateur u sélectionnée est la note de l'utilisateur entendue la plus proche de la note n (voir figure 1). À partir des dates de ces deux notes, on calcule l'asynchronie entre elles ($asynch_n = date_u - date_n$).

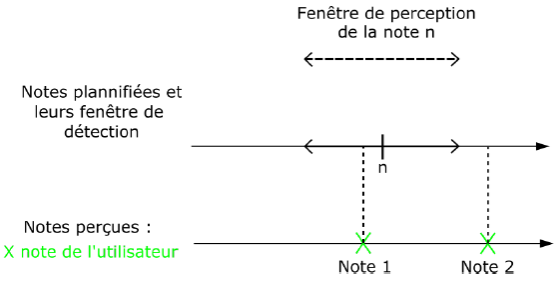

Figure 1 : Représentation sur 2 axes temporelles de ces notes perçues et planifiées, ainsi que la fenêtre de perception d'une note planifiée. Ici, c'est la note 1 de l'utilisateur qui sera prise en compte pour avoir une influence sur la note planifiée suivante, la note 2 n'étant pas dans la fenêtre de perception.

Si l'utilisateur a produit sa note avant l'agent virtuel, l'asynchronie sera positive, si c'est le contraire, elle sera négative. Une fois l'asynchronie entre note utilisateur et note de l'agent déterminée, une correction est appliquée à la note n+1. Pour cette méthode de correction des notes futures, nous nous sommes basés sur l'article de Repp and Keller [11]. Dans ces travaux, les auteurs proposent des calculs de correction pour implémenter un métronome adaptatif. Les auteurs proposent deux étapes dans la correction : la correction de phase, qui vise à modifier directement la date de la note suivante, et la correction de période, qui vise à modifier l'intervalle interne du système. Nous avons repris ces principes pour l'implémentation de la prise de décision de l'agent virtuel. Deux paramètres modifiables dans l'interface de l'application permettent d'ajuster l'influence de ces corrections. Correction de phase :

$$date_{n+1} = date_n + interval_n + PhaseParameter * asynch_n \quad (1)$$

Correction d'intervalle :

$$interval_{n+1} = interval_n + IntervalParameter * asynch_n \quad (2)$$

Dans ces équations, $date_{n+1}$, correspond à la date de la note suivante à déplacer, et $date_n$ correspond à la date de la note qui vient d'être jouée. De même pour $interval_n$ et $interval_{n+1}$ qui correspondent à l'intervalle interne de l'algorithme avant et après la correction. $PhaseParameter$ et $IntervalParameter$ correspondent aux valeurs paramétrables de l'intensité de ces deux corrections (voir figure 2). Plus ces paramètres sont petits, plus l'ajustement se fera lentement.

On obtient donc différents cas possibles à chaque note de l'agent :

- l'utilisateur accélère par rapport à l'agent (la note de l'utilisateur est arrivée avant celle de la machine), l'asynchronie est positive, la prochaine date est avancée dans le temps et l'intervalle est augmenté.
- l'utilisateur ralentit par rapport à la machine (la note de l'utilisateur est arrivée après celle de la machine), l'asynchronie est négative, la prochaine date est reculée dans le temps et l'intervalle est diminué.
- l'utilisateur ne produit pas de note, le système continue de jouer sans être influencé.

À cette étape, l'algorithme ne permet pas encore une boucle sensorimotrice. Pour cela, l'agent s'écoute lui (avec le même principe de niveau d'énergie) autour de la $date_n$ et pour la suite des calculs, la valeur de $date_n$ est remplacée par la valeur perçue. Cet algorithme permet alors à l'agent virtuel de s'adapter totalement au rythme de l'utilisateur. Nous appelons cette configuration mode « suivi ».

3.2.2 Guidage de l'utilisateur par l'agent virtuel. Le couplage est décrit dans [6] comme une action conjointe, impliquant une intention de la part des deux participants. Chaque participant va exprimer son intention, et les deux vont se mettre d'accord sur une situation de synchronisation. Avec l'algorithme de base décrit ci-dessus, une synchronisation peut se mettre en place, mais l'agent virtuel n'y apporte pas d'intention, il suit totalement les actions de l'utilisateur. Nous avons alors introduit l'intention de la part de l'agent virtuel de taper des mains à un certain tempo, avec pour but d'inciter l'utilisateur à le suivre, en le tirant vers ce tempo cible. L'algorithme prend en entrée une valeur de tempo en paramètre (ajustable dans l'interface), et a comme intention de l'atteindre et de maintenir le rythme à ce tempo, sans pour autant perdre l'utilisateur. Pour rester synchronisé avec l'utilisateur tout en essayant de le tirer vers sa cible, l'algorithme va avant tout continuer à s'adapter aux actions de l'utilisateur avec le même principe de correction de phase et d'intervalle que dans l'algorithme de suivi, mais tout en intégrant dans sa correction cette intention de tempo. Celle-ci est liée à un paramètre (appelé $GuidingParameter$) correspondant à l'intensité du guidage. Si ce paramètre est petit, l'agent suivra plus l'utilisateur et décalera peu vers son tempo cible, si ce paramètre est grand, l'agent décalera beaucoup ses notes vers le tempo cible. Pour calculer les corrections vers le tempo cible, nous utilisons toujours

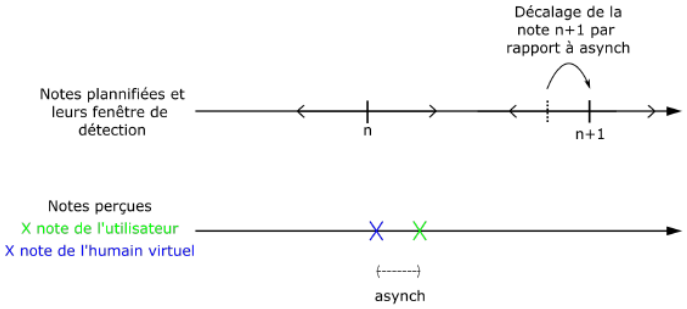

Figure 2 : Décalage de la date de la note n+1 après la détection d'une asynchronie positive sur la date de la note n

les formules 1 et 2, mais nous leur ajoutons la différence entre l'intervalle à atteindre et l'intervalle actuel multiplié par le paramètre *GuidingParameter*.

Ce mode de fonctionnement permet d'ajouter dans l'algorithme une intention de la part de l'agent, sans pour autant cesser la recherche de synchronisation avec l'utilisateur. L'agent va suivre l'utilisateur quand celui-ci ralentit ou accélère, mais va toujours essayer de le tirer vers son tempo cible. Nous appelons cette configuration mode « guidage ».

3.3 Action

Une fois les dates des notes prévues, le système doit effectuer l'action lorsque le moment est venu. Dans ce projet, le comportement de l'agent virtuel est multimodale : il passe par le son et le mouvement. Pour le son, le système doit jouer un son de claquement de mains précisément aux dates prévues par l'algorithme de décision. En plus du déclenchement du son, l'agent virtuel doit animer son corps, notamment les mouvements de ses bras pour qu'ils correspondent au son. Il anticipe donc sur les dates des notes pour déclencher les mouvements de sorte que ses mains se touchent au moment où le claquement de main se fait entendre.

4 IMPLÉMENTATION ET ARCHITECTURE DU SYSTÈME

Pour l'animation 3D de l'agent virtuel, nous avons utilisé le moteur *Unity3D*[4]. *Unity3D* est un logiciel conçu pour le jeu plus que pour la simulation, la précision temporelle de l'ordonnanceur a été pensée pour le rendu visuel. *Unity3D* a donc sa propre échelle de temps qui n'est pas forcément régulière ni prévisible pour la précision exigée.

Pour assurer une interaction multimodale en temps réel respectant les exigences de la précision musicale, nous avons développé une application externe qui utilise le framework *JUCE*[5]. *JUCE* est un framework multiplateforme pour le développement d'applications et de plugins audio. Une application *JUCE* fonctionne à bas-niveau, communicant directement avec les buffers de la carte son en entrée et en sortie et peut utiliser une fréquence d'échantillonnage de 44100 Hz, ce qui permet une grande précision temporelle dans les calculs.

La communication entre l'application *Unity3D* et l'application *JUCE* se fait par connexion TCP et les deux programmes prennent comme référence temporelle le temps CPU, de manière à être synchronisés, et éviter ralentissements et irrégularités du côté de *Unity3D*.

La figure 3 montre l'architecture du système et une photo des différents composants du système. À chaque pas d'exécution, le module *Listener* de l'application *JUCE* récupère un buffer d'entrée, correspondant aux données récupérées par les microphones (perception des actions de l'utilisateur et de l'agent virtuel). Les valeurs contenues dans les buffers sont envoyées vers la fonction de détection d'enveloppe, qui calcule ainsi les dates des notes d'un côté ou de l'autre. Pour le type de son à détecter, la détection d'enveloppe s'est montrée très robuste et peu sensible aux bruits environnants. Ensuite, le module *Decider* exécute l'algorithme de décision décrit ci-dessus dans la section 3.2 en mode suivi ou guidage selon le paramétrage initial de l'application. Lorsqu'un clappement des mains doit être joué, l'application *JUCE* envoie par le biais du module *Server TCP* une commande à *Unity3D* et une copie d'un fichier sonore (correspondant au son des mains qui se tapent) dans le buffer de sortie pour que ces données soient jouées dans les enceintes.

Dans le script *Unity3D*, les dates des évènements sont stockées dans une pile, pour être effectuées les unes après les autres. Le script d'animation des mouvements prend en compte la date de la prochaine note. Si elle arrive dans plus de 400 millisecondes, le script déclenche un mouvement ample des bras ; autrement, il déclenche un mouvement des mains plus réduit dans l'espace. Pour créer ces mouvements des bras, plusieurs positions des poignets (*keyframes*) sont gardées en mémoire (bras ouvert, bras fermés, bras reculés pour mouvement large). Le script place les *keyframe* en prenant le temps CPU comme référence temporelle. Ensuite, le mouvement des bras est obtenu par interpolation des *keyframes* et l'animation de tout le reste de l'agent virtuel est calculée en utilisant la méthode de cinématique inverse à l'aide du plug-in *Final* IK de *RootMotion*[6].

Pour nous assurer du bon fonctionnement de notre système, nous avons fait taper des mains à l'agent virtuel pendant une minute en imposant une fréquence de 60 bpm. Nous avons enregistré les dates de toutes les notes générées par le système et calculé la racine de l'erreur quadratique moyenne de la fréquence instantanée (moyenne glissante sur quatre intervalles) par rapport au

4. https ://unity.com/fr
5. https ://JUCE.com
6. http ://root-motion.com/

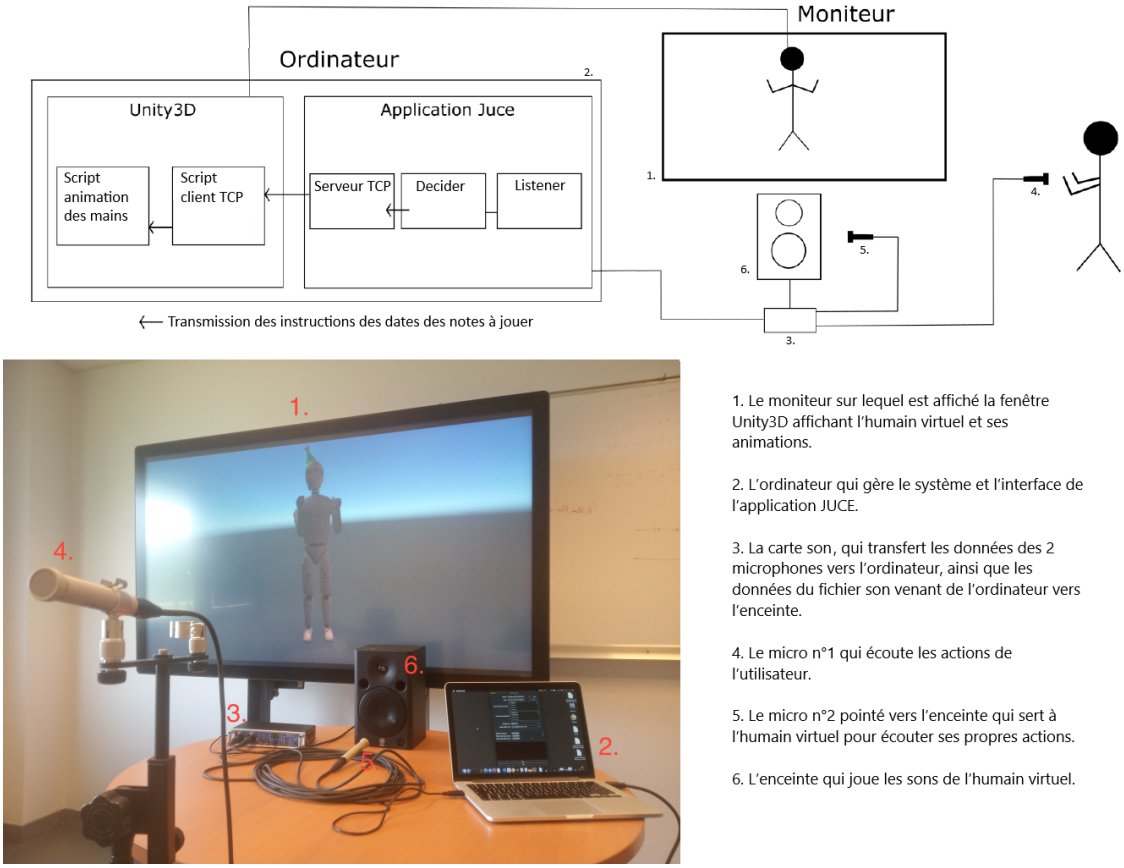

Figure 3 : Architecture globale et photo des différents composants. La carte son est une RME Fireface UCX. Les microphones polyvalents sont des Oktava MK 012-01 MSP2.

tempo donné ($REQM = 0.06bpm, moy = 59,99999bpm$, *écart max* $= 0.345bpm$). Mis à part un incident au niveau de l'écart max, le degré de dispersion des valeurs reste très faible, ce qui nous permet de dire que le système est capable de tenir de façon fiable un tempo imposé.

La figure 4 montre la variation du tempo de l'agent virtuel (ligne rouge continue) et celui de l'humain (ligne verte segmentée) pendant 60 seconds d'interaction en mode guidage, ainsi que l'écart temporel entre leurs notes (ligne bleue pointillée). L'agent tape des mains à 60 bpm pendant 10 secondes avant d'entrer en mode guidage. Nous remarquons que lorsque l'humain ralentit son tempo (à partir de 20 secondes environ), l'agent virtuel le suit (l'écart temporel entre les claquements des mains est négatif, ce qui montre que l'agent est en léger retard pour s'adapter au tempo de l'utilisateur). Dès que l'humain poursuit sur le même tempo, l'agent tente de le ramener à son tempo cible de 60 bpm. Remarquons alors que l'écart temporel entre les claquements des mains est positif, ce qui montre que cette fois-ci l'humain est en léger retard pour s'adapter au tempo de l'agent.

5 EXPÉRIMENTATION PRÉLIMINAIRE

Nous avons mené une expérimentation préliminaire du système pour recueillir le ressenti de l'utilisateur lors d'une interaction avec l'agent virtuel. Nous avons considéré trois modes de comportement de l'agent virtuel qui ont déterminé trois conditions d'interaction :

- A) le mode métronome, où l'agent virtuel reste sur un rythme fixe de 60 bpm, n'effectuant aucune adaptation d'après les claquements de mains de l'utilisateur ;
- B) le mode suivi, décrit dans la partie 3.2.1, dans lequel l'agent virtuel, après 10 secondes initiales à 60 bmp, suit totalement le rythme de l'utilisateur ;
- C) le mode guidage, décrit dans la partie 3.2.2, dans lequel l'agent virtuel, après 10 secondes initiales à 60 bmp, va s'adapter au rythme de l'utilisateur tout en essayant de l'attirer vers son tempo cible de 60 bpm.

Douze personnes (dont une femme) ont accepté de participer à l'expérimentation et, après avoir signé une fiche de consentement, ont interagi avec l'agent dans les trois conditions indiquées (une minute par condition). L'ordre des conditions a varié pour chaque participant. À la fin de chaque condition, le participant répondait à une question sur son expérience avec l'agent : il devait dire s'il sentait que l'agent ne le suivait pas du tout, le suivait totalement

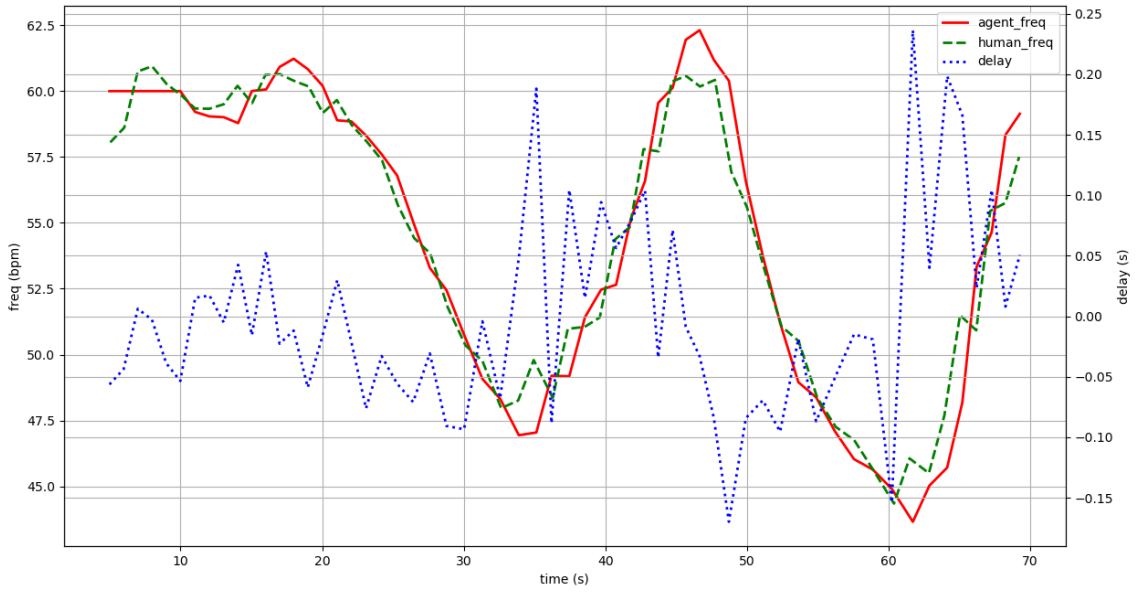

Figure 4 : Variation des tempos (moyenne glissante sur 4 intervalles) de l'agent virtuel (ligne rouge continue) et de l'humain (ligne verte segmentée) pendant 60 secondes d'interaction en mode guidage, ainsi que l'écart temporel entre leurs notes (ligne bleue pointillée).

ou si on était dans une situation de collaboration. Enfin, après la dernière condition, le participant indiquait dans quelle condition il avait ressenti une interaction plus forte avec l'agent virtuel. Le faible nombre de participants ne nous a pas permis d'obtenir des résultats statistiquement significatifs, mais nous avons pu constater que le ressenti des participants est encourageant. Cinq participants ont répondu que la situation C comportait le plus d'interaction, quatre ont répondu la situation B, et trois la A. Ces résultats montrent une tendance des participants à ressentir plus une interaction avec l'agent virtuel dans les conditions B et C, qui sont les situations où les modes suivi et guidage sont actifs.

Cette première évaluation nous a aussi permis de collecter des remarques permettant de soulever plusieurs points à améliorer. Le mode guidage reste encore à perfectionner. Dans le cas où l'agent doit maintenir un tempo auquel il est déjà, il s'avère plutôt efficace, mais lorsqu'il veut pousser l'utilisateur à changer de tempo, le décalage se fait trop brusquement et a tendance à perturber l'utilisateur. Nous devons trouver une valeur plus adéquate pour le paramètre *GuidingParameter*. Des problèmes sont à noter au niveau visuel : les mouvements de l'agent virtuel peuvent être perturbant. Cela peut être causé par un manque de finesse dans l'animation, l'agent effectue des changements brusques dans ses mouvements, passant d'un mouvement ample à un mouvement plus réduit d'un coup lorsque la valeur du tempo dépasse un certain seuil. Ce fonctionnement perturbe l'utilisateur, une évolution progressive de l'amplitude des mouvements en fonction de la vitesse du rythme pourrait être plus efficace pour permettre à l'utilisateur de mieux anticiper le rythme. Il est important de souligner que le niveau d'expertise musicale de l'utilisateur semble impacter fortement son ressenti. Ces différences de profils devraient être prises en compte lors de travaux ultérieurs. Notons que deux utilisateurs supplémentaires, ayant une expérience de l'enseignement musical, ont retrouvé des sensations analogues à l'interaction qu'ils pouvaient avoir avec des élèves dans le cadre du suivi de tempo. Ce ressenti, qui correspond à nos motivations initiales, mérite d'être validé à l'avenir. Enfin, le comportement de l'agent gagnerait en naturel avec une animation « idle » liée au tempo auquel joue l'agent.

6 CONCLUSION

Nous avons présenté une architecture, son implémentation ainsi qu'une expérimentation préliminaire d'un système qui vise un couplage d'une haute précision temporelle avec un agent virtuel, tout en utilisant les outils standards d'animation 3D. Le résultat est un système qui permet une synchronisation rythmique avec l'utilisateur, avec un mode de suivi du tempo efficace et réactif. La mise en place d'un outil plus bas niveau extérieur à *Unity3D* permet des calculs et une gestion du temps satisfaisante pour cette tâche. La notion de couplage, impliquant une intention de la part des deux participants, doit encore être plus explorée. En effet, le mode guidage amenant une intention de la part de l'agent virtuel n'est pas convainquant pour des situations où le système doit changer de tempo au cours de l'interaction. Les animations de l'agent virtuel restent aussi à améliorer, pour obtenir un comportement plus naturel et plus progressif dans l'ampleur des mouvements des bras en fonction du tempo. Enfin, nous souhaiterons, à plus long terme, étudier des patterns polyrythmiques plus élaborés.

RÉFÉRENCES

[1] Elisabetta Bevacqua et Gireg Desmeulles. 2016. Interaction entre humains virtuels et réels dans le cadre de la percussion corporelle. In *Workshop Affect Compagnon Artificiel Interaction WACAI 2016*. ENIB. Brest, France, (juin 2016). https://hal.archives-ouvertes.fr/hal-01411938.

[2] Elisabetta Bevacqua et Gireg Desmeulles. 2017. Real and virtual body percussionists interaction. In *MOCO'17 Proceedings of the 4th International Conference on Movement Computing*. London, United Kingdom. https://hal.archives-ouvertes.fr/hal-01590017.

[3] Elisabetta Bevacqua, Sankovic Igor, Maatalaoui Ayoub, Alexis Nédélec et Pierre De Loor. 2014. Effects of coupling in human-virtual agent body interaction. In (sept. 2014), 54-63. isbn : 978-3-319-09766-4. doi : 10.1007/978-3-319-09767-1_7.

[4] Emilio Bizzi et Robert Ajemian. 2020. From motor planning to execution : a sensorimotor loop perspective. *Journal of Neurophysiology*, 124, 6, 1815-1823. PMID : 33052779. doi : 10.1152/jn.00715.2019.

[5] E. Coorevits, P. Maes et Marc Leman. 2017. Gesture in the communication and control of musical timing. In *the 25th Anniversary Conference of the European Society for the Cognitive Sciences of Music*, 40-47.

[6] Pierre De Loor, Romain Richard et Elisabetta Bevacqua. 2017. Interaction corporelle évolutive entre un humain et un personnage virtuel. *Revue des Sciences et Technologies de l'Information - Série RIA : Revue d'Intelligence Artificielle*, (oct. 2017). doi : 10.3166/RIA.31.1-23.

[7] Arno Hartholt, David Traum, Stacy C. Marsella, Ari Shapiro, Giota Stratou, Anton Leuski, Louis-Philippe Morency et Jonathan Gratch. 2013. All Together Now : Introducing the Virtual Human Toolkit. In *13th International Conference on Intelligent Virtual Agents*. Edinburgh, UK, (août 2013). http://ict.usc.edu/pubs/All%20Together%20Now.pdf.

[8] G. Luck et Petri Toiviainen. 2006. Ensemble musicians ? synchronization with conductors ? gestures : an automated feature-extraction analysis. *Music Perception - MUSIC PERCEPT*, 24, (déc. 2006), 189-200. doi : 10.1525/mp.2006.24.2.189.

[9] Ken Prepin et Catherine Pelachaud. 2013. Basics of intersubjectivity dynamics : model of synchrony emergence when dialogue partners understand each other. In *International Conference on Agents and Artificial Intelligence*. T. 271. (Jan. 2013), 302-318. isbn : 978-3-642-29965-0. doi : 10.1007/978-3-642-29966-7_20.

[10] Bruno Repp. 2006. Sensorimotor synchronization : a review of the tapping literature. *Psychonomic bulletin & review*, 12, (jan. 2006), 969-92. doi : 10.3758/BF03206433.

[11] Bruno Repp et Peter Keller. 2008. Sensorimotor synchronization with adaptively timed sequences. *Human movement science*, 27, (juill. 2008), 423-56. doi : 10.1016/j.humov.2008.02.016.

[12] Gérard Sensevy. 2011. *Le sens du Savoir. Eléments pour une théorie de l'action conjointe en didactique*. De Boeck, 800. https://halshs.archives-ouvertes.fr/halshs-00856455.

[13] Luca Turchet. 2023. Musical metaverse : vision, opportunities, and challenges. *Personal and Ubiquitous Computing*, 27, (jan. 2023), 1811-1827. doi : 10.1007/s00779-023-01708-1.