M2IIA : ALTERNANCE

Université de Bourgogne, UFR Sciences et Techniques, Département I.E.M.

Rapport d'alternance

Auteur/Autrice
Maria Garcés-Cala

Génération d'Animations de Personnages 3D en utilisant l'Intelligence Artificielle

X&Immersion

Yani Moghraoui
Romain Raffin

2023-24

Table des matières

1.1 Introduction 5
1.2 Présentation de l'entreprise 6
1.2.1 Organisation 6
1.2.2 Produits 6
1.2.3 Contexte 10
1.3 Évaluation du Projet : Analyse et Solutions Envisagées 11
1.3.1 Développé en ce moment 11
1.3.2 L'étude de besoin 12
1.3.3 Propositions envisagées 13
1.3.4 Données utilisées 17
1.4 Travail réalisé 18
1.4.1 Méthodologie 18
1.4.2 MeshTalk 19
1.4.3 FaceFormer 21
1.5 Conclusion 27

Résumé Les studios de jeux vidéo sont confrontés à une charge de travail élevée dans la génération d'animations faciales pour les personnages. Pour aborder cette problématique sans compromettre la qualité des résultats, un outil est en cours de développement, spécialement conçu pour optimiser ce processus.

Ce projet exploite l'Intelligence Artificielle, en particulier les modèles de réseaux neuronaux, qui offrent une solution efficace pour la génération d'animations. Un modèle existant est mis en œuvre, adapté aux besoins de l'industrie du jeu vidéo, permettant aux utilisateurs de générer des animations faciales pour leurs personnages de manière simple et efficace depuis un moteur de jeux vidéo.

Mots-clés Animations faciales, optimisation, intelligence artificielle réseaux neuronaux, industrie du jeu vidéo

---

Abstract Video game studios face a high workload in generating facial animations for characters. To address this issue without compromising the quality of the results, a tool is being developed, specifically designed to optimize this process.

This project leverages Artificial Intelligence, particularly neural network models, which provide an effective solution for animation generation. An existing model is implemented, tailored to the needs of the video game industry, allowing users to generate facial animations for their characters in a simple and efficient manner from a game engine.

Keywords Facial animations, optimization, artificial intelligence, neural networks, video game industry

---

1.1 Introduction

Le présent projet se concentre sur le développement et la mise en œuvre d'outils d'intelligence artificielle pour améliorer les processus de génération de contenu dans l'industrie du jeu vidéo, en particulier dans le domaine de l'animation faciale. En utilisant des modèles de réseaux neuronaux, l'objectif est d'optimiser la création d'animations faciales synchronisées avec le dialogue des personnages, permettant aux studios de développement de jeux vidéo d'accélérer leurs processus de production et d'obtenir des résultats de haute qualité.

Dans ce rapport, le travail réalisé pendant mon alternance chez X&Immersion est détaillé. J'ai participé au développement et à l'essai des projets axé sur ce domaine. Celui-ci se concentre sur l'adaptation et la réentraînement d'un modèle de réseaux neuronaux préexistant pour la génération d'animations faciales réalistes et expressives, basées sur l'audio du dialogue des personnages.

Les différentes étapes du processus sont décrites. Cela inclut la recherche initiale des projets et la compréhension de chaque modèle afin de choisir celui qui est le plus approprié en tenant compte de certaines caractéristiques. Il comprend également les tests effectués et l'intégration du modèle dans l'environnement de développement de jeux vidéo. Les progrès, les obstacles rencontrés et les solutions mises en œuvre au cours du développement du projet sont aussi analysés.

Il est crucial de souligner que le projet actuel est à un stade de développement actif, ce qui implique qu'il contient des informations confidentielles. Par conséquent, il est essentiel de maintenir la discrétion en ce qui concerne la gestion et la distribution du document.

1.2 Présentation de l'entreprise

X&Immersion est une société technologique française créée en 2021, qui a pour objectif d'aider les studios de jeux vidéo dans la création de contenu et la production.
La société fournit des outils permettant la création de textes, d'audio et d'animations de manière efficace. Cela permet d'optimiser les temps de production et ainsi de permettre aux studios d'obtenir les meilleures résultats en se concentrant sur d'autres processus de développement.

Malgré son jeune âge, elle a déjà réalisé des projets qui lui ont permis de se faire connaître pour la qualité de son travail et son utilité. La plupart de ces projets ont été réalisés pour des studios de jeux vidéo, mais aussi pour d'autres organisations ayant besoin de services de génération de dialogues de personnages. Certains des clients avec lesquels ils ont travaillé incluent : mainbot, Microïds, Grand Palais Immersif, Hawkswell, entre autres.

1.2.1 Organisation

Sous la présidence d'Alexandre Folliot, CEO de X&Immersion, et avec les membres de l'équipe exécutive : Oriane Piedevache-Opsomer en tant que COO, Côme Demarigny en tant que CFO et Yani Mogharoui en tant que CR&DO ; l'entreprise a connu une croissance et est composé actuellement de 10 personnes. Celles-ci sont réparties dans 3 départements :

— Marketing / Produit : Ce département est chargé de développer des stratégies de marketing, de promouvoir les produits et services de l'entreprise, ainsi que de gérer l'image de marque. Il est également responsable du développement et de l'amélioration continue des produits et services offerts.
— Tech : Ce département se concentre sur le développement, la maintenance et l'amélioration des solutions technologiques de l'entreprise. Cela inclut le développement de logiciels, la recherche et le développement de nouvelles technologies et services.
— Opérations / Ventes : Ce département est responsable de la gestion des ventes, des finances et de la distribution des produits et services de l'entreprise. Par conséquent, il se concentre également sur l'acquisition de nouveaux clients et la recherche de nouvelles opportunités sur le marché.

En tenant compte de ce qui précède, je suis actuellement en alternance dans le département de Tech. Ce dernier est chargé de développer les outils de génération audio, texte et animation, en assurant la conception, le développement et la mise en œuvre dans les projets des clients selon leurs besoins. Actuellement, il se concentre principalement sur l'amélioration des outils existants avec le soutien de l'IA pour la génération d'animations et audio.

1.2.2 Produits

X&Immersion développe trois produits principalement destinés aux studios de jeux vidéo. Cependant, ils peuvent être utilisés par d'autres organisations ayant besoin de ce type de services. Il est important de noter que ces outils sont utilisés comme plug-in dans les deux

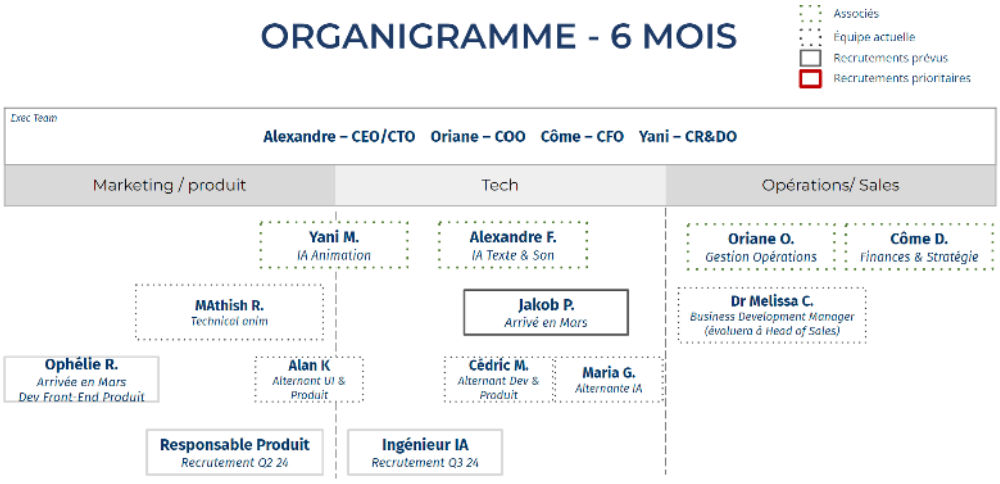


FIGURE 1.1 – Organigramme

applications les plus utilisées sur ce marché : Unity et Unreal Engine. Ci-dessous, une explication plus détaillée de chacun des produits.

Adabot

Adabot est un outil de génération de texte conçu pour rationaliser le processus de développement de personnages et de création de dialogues pour les écrivains. En utilisant les données fournies par l'utilisateur, Adabot génère des modèles d'IA qui imitent le comportement et les schémas de parole de personnages spécifiques. Cela permet aux écrivains de rédiger rapidement des dialogues, des discussions et des récits complexes avec facilité.

L'un des principaux avantages d'Adabot est son intégration transparente avec les logiciels d'écriture existants. En incorporant Adabot directement dans leurs outils d'écriture, les équipes peuvent exploiter ses capacités sans effort, les habilitant à libérer leur créativité tout en maintenant l'efficacité.

Ariel

Ariel est un outil qui permet de générer de l'audio à partir de texte de manière efficace et réaliste. Il utilise des modèles de synthèse vocale formés sur seulement 15 minutes d'audio d'un acteur pour produire des voix naturelles et expressives dans un large éventail de langues.

Ce plug-in fournit une interface facile à utiliser où les utilisateurs peuvent saisir du texte et ajuster divers paramètres pour contrôler le ton, la vitesse et l'intonation de la voix générée. Vous pouvez voir à quoi ressemble l'interface dans Unreal Engine dans la Figure 1.2. Dans Unity, elle a une apparence différente mais contient les mêmes fonctionnalités.

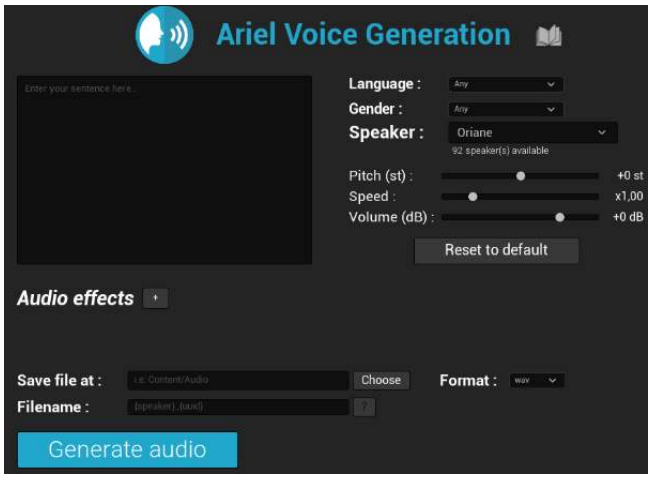


FIGURE 1.2 – Interface Ariel dans UE

Geppetto

Geppetto est un outil qui se concentre sur la génération de phonèmes et d'animations faciales à partir de texte ou d'audio. Ce plug-in utilise des techniques de synthèse vocale et d'animation faciale basées sur des modèles d'apprentissage profond pour créer des animations réalistes de personnages en temps réel dans le moteur de jeu.

Dans la Figure 1.3, vous pouvez voir l'interface dans Unreal Engine. Pour utiliser cet outil, les développeurs doivent d'abord entrer du texte ou de l'audio. Ensuite, l'outil analyse le contenu pour déterminer les mouvements faciaux nécessaires (phonèmes) et produire ainsi une synchronisation labiale appropriée avec la parole. De plus, il permet d'ajuster divers paramètres pour contrôler l'expressivité et la fidélité des animations faciales générées.

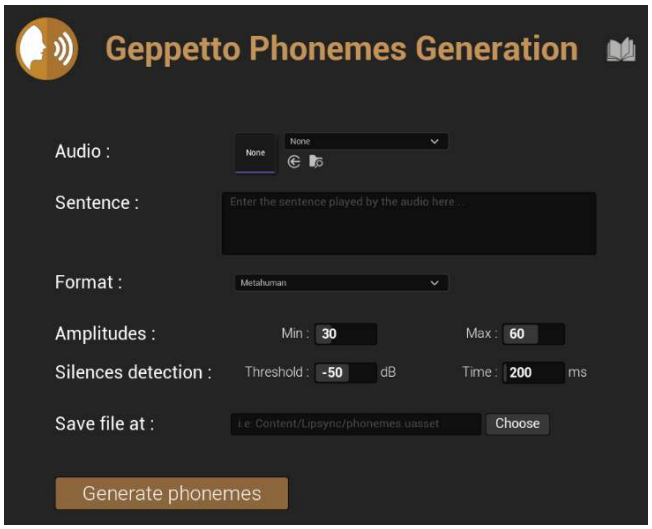


FIGURE 1.3 – Interface Geppetto dans UE

En résumé, ces outils permettent de simplifier les processus. Avec Ariel, l'intégration de l'audio généré par IA permet aux développeurs d'ajouter des narrations, des dialogues et des commentaires de manière plus efficace. Quant à Geppetto, il simplifie la création d'animations faciales synchronisées avec la parole, offrant ainsi aux développeurs ou animateurs plus de flexibilité dans l'animation des personnages en temps réel. Et finalement, Adabot qui génère des milliers de lignes de dialogue en quelques secondes pour permettre aux écrivains de composer rapidement des dialogues et des discussions.

En général, l'entreprise se concentre sur l'amélioration continue de l'expérience utilisateur. Par conséquent, elle s'est concentrée ces derniers mois sur la construction d'un service web qui facilite les tests et la génération d'assets pour ses produits. De plus, elle crée de nouveaux outils ou produits utilisant l'intelligence artificielle, afin de continuer à faciliter la production dans le domaine des jeux vidéo et des contenus visuels.

1.2.3 Contexte

Pour comprendre pleinement les défis de l'animation faciale, il est crucial de contextualiser les besoins du marché et les défis techniques auxquels sont confrontés les créateurs de contenu. Actuellement, la demande d'animations faciales réalistes et personnalisées est en augmentation, en particulier avec l'essor des plateformes de réseaux sociaux, des jeux vidéo et des applications de réalité augmentée et virtuelle. À cela s'ajoute l'augmentation des attentes des utilisateurs finaux, qui exigent des solutions plus rapides et plus précises.

Compte tenu de ce qui précède, il est important d'analyser les besoins des clients, où la production de contenu animé à grande échelle, la réduction des délais de production, la nécessité de personnalisation et la demande croissante de synchronisation multilingue constituent des impératifs incontournables. Face à ces défis, les avancées en intelligence artificielle et en apprentissage profond sont devenues les meilleures outils pour de nouvelles approches automatisées, car elles permettent d'accélérer le processus d'animation tout en maintenant, et même en améliorant, la qualité et les résultats.

Ensuite, les besoins spécifiques du client et les limitations actuelles du processus d'animation faciale seront analysés de près. Les avantages potentiels de l'intégration de l'intelligence artificielle seront également explorés, en mettant en avant les opportunités qu'elle offre en termes de vitesse, d'efficacité, de personnalisation et de scalabilité. Enfin, cela jettera les bases pour la future mise en œuvre d'un modèle basé sur l'IA afin de répondre aux demandes croissantes du marché et aux attentes des clients en matière d'animation faciale.

1.3 Évaluation du Projet : Analyse et Solutions Envisagées

Dans ce chapitre, nous nous plongeons dans la phase précédant le démarrage du travail. Initialement, l'état actuel d'outil dans l'entreprise est expliqué afin de fournir une vision globale et de comprendre pourquoi des améliorations sont souhaitées. Ensuite, une analyse détaillée des besoins des clients potentiels est fournie. Ici, nous analyserons en profondeur la problématique en relation avec les utilisateurs et ce qui est attendu du produit. Aussi, les propositions envisagées et les critères utilisés pour évaluer les modèles seront présentés. Cela afin de garantir que la décision prise est la plus adaptée aux besoins identifiés. Enfin, les données utilisées dans la mise en œuvre, leur provenance et la manière dont elles sont traitées seront détaillées.

1.3.1 Développé en ce moment

Comme mentionné précédemment (*Sous-section 1.2.2)*, l'entreprise dispose de l'outil Geppetto, qui permet de générer des animations faciales synchronisées avec le timing et le ton du discours d'un personnage. Actuellement, ce processus se déroule comme suit :

1. Le processus commence avec un audio ou un texte. Il s'agit du dialogue ou des lignes que l'on souhaite que le personnage animé dise.
2. L'audio ou le texte est traité à travers un modèle de reconnaissance vocale. Cette étape peut impliquer la conversion de la voix en texte (si l'entrée est audio), ainsi que l'identification des mots prononcés.
3. Ensuite, le modèle analyse le texte pour détecter les phonèmes (les unités de son de la parole) et les émotions associées, ainsi que les données de temps correspondantes. Les phonèmes sont essentiels pour synchroniser correctement les mouvements de la bouche du personnage avec la parole, les données de temps aident à aligner l'animation avec l'audio, et l'émotion aide à ajuster l'expression faciale générale.
4. Les données générées précédemment sont alimentées dans une technologie propriétaire qui utilise une base de données d'animations. Ici, les phonèmes et les émotions sont associés à des animations spécifiques de mouvements faciaux.
5. La sortie du processus précédent est appliquée à un modèle de personnage dans un standard 3D, tel que Unreal Engine, Unity ou un système personnalisé. Cette étape adapte les animations générées pour qu'elles correspondent aux modèles et riggings spécifiques du personnage présents dans le moteur ou la plateforme d'animation que vous utilisez.
6. Enfin, vous obtenez une animation faciale 3D qui répond à l'audio original et qui peut être facilement adaptée par les animateurs. Cela signifie que l'animation peut être affinée et améliorée manuellement pour atteindre le niveau de qualité ou d'expressivité souhaité.

Ce processus peut être efficace car il simplifie le travail des animateurs et des artistes. Cependant, il peut encore être considéré comme traditionnel ou manuel, ce qui implique une charge de travail qui peut être réduite davantage avec les bons outils.

1.3.2 L'étude de besoin

Initialement, une étude des besoins des clients a été réalisée afin de trouver la meilleure solution au problème. Certains de ces besoins peuvent inclure :

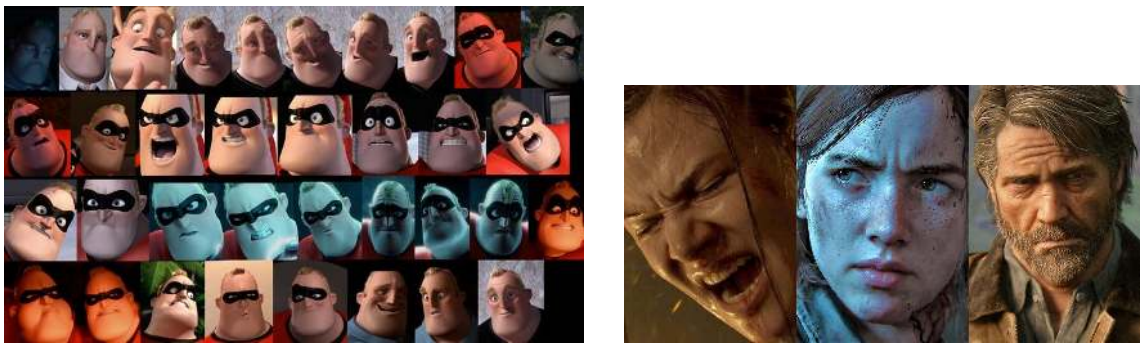
— La production d'une grande quantité de contenu animé avec de nombreux personnages nécessitant une animation faciale, sans perdre en dynamisme et en qualité.
— La réduction des délais de production car ils sont souvent serrés et il est nécessaire d'accélérer le processus d'animation pour respecter les calendriers de livraison sans sacrifier la qualité.
— La nécessité de pouvoir animer une variété de personnages avec différents styles d'animation, des traits faciaux uniques et des expressions émotionnelles diverses.
— La recherche de moyens de réduire les coûts associés à l'animation traditionnelle, qui implique souvent de grandes équipes spécialisées d'animateurs.
— Pour les projets qui seront distribués à l'échelle mondiale, il est nécessaire d'avoir un moyen efficace de synchroniser les animations faciales avec plusieurs langues et dialectes.
— La recherche de moyens de se démarquer de la concurrence en utilisant les dernières technologies en matière d'IA pour offrir des expériences plus immersives et de pointe.

À partir de là, une solution qui offre une combinaison de vitesse, d'efficacité, de personnalisation et de scalabilité est requise. Cela est difficile à réaliser avec des méthodes traditionnelles ou manuelles. Pour cette raison, l'utilisation de l'intelligence artificielle, en particulier les réseaux neuronaux, a été envisagée pour aborder ce processus. Voici quelques raisons pour lesquelles ce choix a été fait :

— Les réseaux neuronaux peuvent apprendre à effectuer des tâches complexes telles que la synchronisation des lèvres et l'expression des émotions à partir de grands ensembles de données. Cela peut réduire considérablement le temps nécessaire pour animer des personnages, ce qui serait particulièrement bénéfique dans les productions avec beaucoup de dialogues.
— Avec suffisamment de données d'entraînement, il est possible de générer des animations faciales qui captent des subtilités et des nuances qui pourraient être négligées dans des processus moins avancés. Cela peut conduire à des animations plus réalistes et convaincantes.
— Le modèle peut être entraîné pour s'adapter à différents styles d'animation ou à des caractéristiques spécifiques des personnages, ce qui permet une plus grande personnalisation sans avoir à créer de nouvelles règles ou ajustements pour chaque situation. Par exemple, Les Indestructibles de Pixar démontre un style expressif et élastique avec des visages aux expressions amplifiées, comme illustré par l'image 1.4a. En revanche, The Last of Us Part II présente un style réaliste où les mouvements de la bouche et les expressions faciales sont capturés avec un niveau de détail extrême, comme le montre l'image 1.4b. Ces exemples montrent comment le modèle pourrait être adaptable à différents niveaux de complexité et styles d'animation.
— Une fois que le réseau neuronal est bien entraîné, il peut être mis à l'échelle pour des projets de n'importe quelle taille avec relative facilité, en maintenant la cohérence de la qualité et du style de l'animation dans tout le contenu.

— Bien que la formation d'une IA puisse nécessiter un investissement initial significatif, une fois établie, elle peut réduire les coûts associés aux délais de production à long terme.

Sur la base des informations précédentes, il a été proposé d'utiliser un modèle qui, grâce à l'entrée d'un type de donnée comme un audio, un texte, un maillage 3D ou un rig, permet de générer des animations faciales pour un personnage. Cela inclut principalement les mouvements de la bouche, ainsi que la partie supérieure du visage (sourcils, front) et les yeux pour ajouter plus de réalisme au résultat. De plus, la sélection et l'utilisation d'expressions et d'émotions ont été prises en compte.



(a) Les Indestructibles (b) The Last of Us II

FIGURE 1.4 – Exemples de styles d'animation

1.3.3 Propositions envisagées

À partir d'une recherche effectuée par l'alternant précédent, des projets ont été listés dans lesquels des modèles de réseaux neuronaux étaient implémentés pour générer des animations 3D de visages avec des émotions et des expressions à partir de l'audio.

Parmi ceux-ci, nous trouvons VOCA [Cudeiro et al., 2019], qui à partir d'un signal audio arbitraire et d'un maillage facial 3D statique en entrée, le modèle produit une animation réaliste de personnage 3D. Cependant, le mouvement de la partie supérieure du visage (comme les yeux et les sourcils) n'est pas fortement corrélé avec l'audio.
Un autre modèle est celui développé par Pham et al. [Pham et al., 2018]. Ce projet mappe directement une séquence d'entrée audio de parole à une série d'activations d'unités d'action faciale micro et de rotations de tête pour piloter un modèle de visage 3D à base de blendshapes. Cependant, il existe certaines limitations dans l'architecture du réseau qui empêchent le modèle de refléter parfaitement l'émotion dans la parole.

Finalement, il a été décidé de commencer par le projet MeshTalk développé par Meta. La principale raison de ce choix est que ce modèle prend en compte le fait que la parole n'encode pas tous les aspects des expressions faciales. De plus, la décision a été influencée par la solide réputation de Meta, une entreprise reconnue pour son expérience et son leadership dans le développement des technologies d'IA.

Ce modèle est conçu pour animer une maillage facial neutre arbitraire en utilisant uniquement la parole. Les développeurs du modèle affirment que la parole ne code pas entièrement les expressions faciales, telles que le clignement des yeux, ce qui peut entraîner des résultats statiques ou inhabituels dans la partie supérieure du visage lorsque des approches basées exclusivement sur l'audio sont utilisées. Pour surmonter ce problème, le modèle utilise un espace latent catégorique pour les expressions faciales, lequel doit posséder certaines propriétés :

— *Categorical :* il faut représenter l'organisation des données en catégories discrètes.
— *Expressive :* il doit pouvoir coder une variété d'expressions faciales.
— *Semantically disentangled :* l'information liée et non liée à la parole doit être partiellement démêlée. Par exemple, le fermeture des yeux ne devrait pas nécessairement être liée à une forme spécifique des lèvres.

Concernant son architecture et son fonctionnement (Figure 1.5), le modèle se compose de deux phases : un encodeur et un décodeur. L'encodeur reçoit des séquences de maillages faciaux et d'extraits audio alignés sur les trames correspondantes. Chaque type de données est traité par son propre encodeur, et les sorties sont fusionnées dans une couche de combinaison avant d'être envoyées à un espace latent catégorique. Cet espace utilise des catégories d'expressions (comme sourire, neutre, triste) pour classer les visages, permettant ainsi de représenter les variations d'expressions faciales par des catégories discrètes. Ensuite, le décodeur prend un modèle facial neutre et mappe l'expression encodée sur ce modèle pour créer une séquence animée de maillages faciaux. L'information passe par une couche bottleneck où le code d'expression est concaténé avec le maillage encodé. Cette couche est suivie de deux couches LSTM pour modéliser les dépendances temporelles, puis de trois couches entièrement connectées qui remappent la représentation aux sommets et construisent le maillage pour la manipulation.

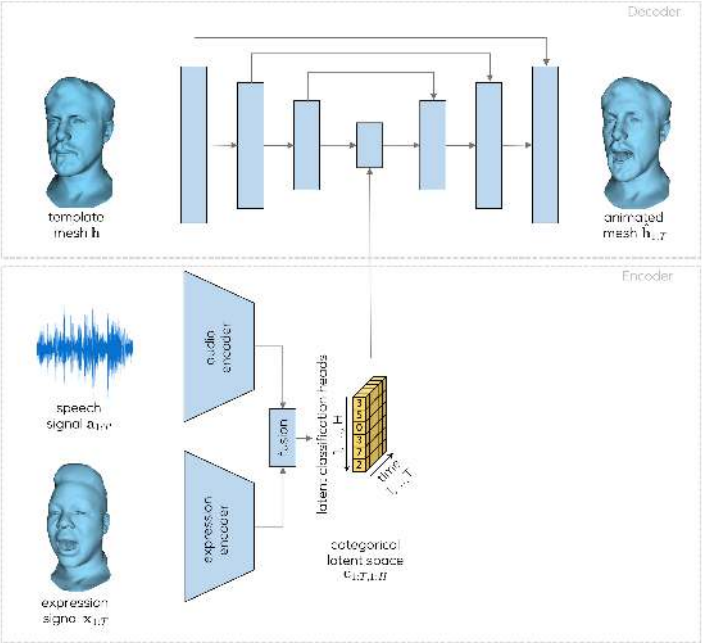


FIGURE 1.5 – Architecture de Meshtalk

Cependant, en raison de problèmes survenus lors du réentraînement du modèle et du manque de progrès du projet, il a été décidé de sélectionner un autre projet. Par conséquent, à partir de la liste initiale et d'une recherche supplémentaire, une nouvelle liste composée de quatre projets a été élaborée. Dans le tableau 1.1, nous pouvons voir les noms de chacun ainsi que les critères utilisés pour sélectionner le projet avec lequel nous continuerions à travailler. Nous avons élaboré un tableau avec quatre critères qui nous ont permis de choisir le projet avec lequel nous procéderions. Voici une brève description de chacun :

— **Accès au code (entraînement) :** Disponibilité du code fourni pour effectuer un éventuel entraînement du modèle.
— **Licence :** Possibilité d'utiliser le projet à des fins commerciales.
— **Accès aux données (test) :** Disponibilité du dataset utilisé par le modèle pour réaliser des tests.
— **Qualité des résultats :** Évaluation de l'expressivité de l'animation, en se concentrant particulièrement sur la bouche, mais aussi en considérant les mouvements dans des zones non directement liées à la parole (comme les yeux et le front).

Comme le montre le tableau 1.1, l'accès aux ressources dans tous les projets est similaire, ce qui a conduit à examiner les deux autres critères. En ce qui concerne la licence, les projets sans licence commerciale ont été écartés en raison de la nature du projet. Enfin, pour analyser la qualité des résultats, un test a été réalisé en utilisant le même maillage et le même audio afin de maintenir le contrôle et la cohérence pour faciliter la comparaison. Ainsi, les deux meilleurs résultats ont été obtenus par FaceXHubert et FaceFormer. Cependant, en raison du critère de licence, la décision a été simple.

Projet	Accès code (training)	Licence	Accès dataset (test)	Qualité résultats
FaceXHuBERT [Haque and Yumak, 2023]	Oui	NonCommercial 4.0 International	Oui	Expressive
FaceDiffuser [Stan et al., 2023]	Oui	NonCommercial 4.0 International	Oui	La bouche est le moins expressive
FaceFormer [Fan et al., 2022]	Oui	MIT	Oui	Expressive
CodeTalker [Xing et al., 2023]	Oui	MIT	Oui	Pas d'expressions dans le front et les yeux

Table 1.1 – Critères

Faceformer est un modèle basé sur l'architecture des *transformers*, conçu pour générer des animations faciales en 3D à partir de la parole. Son objectif est de créer des mouvements faciaux réalistes à partir d'une nouvelle séquence audio, en utilisant comme référence une séquence audio et les mouvements faciaux en 3D correspondants.

Le fonctionnement de Faceformer aborde l'animation faciale comme un problème de séquence à séquence, en prédisant de manière autorégressive les mouvements faciaux. Cette prédiction est conditionnée à la fois par le contexte de l'audio et par la séquence des mouvements faciaux précédents. Ce modèle utilise une approche de encodeur - décodeur, où l'encodeur transforme l'audio en représentations détaillées qui capturent les caractéristiques essentielles de la parole au fil du temps. De plus, l'architecture comprend une couche de style *embedding*, qui intègre des représentations numériques (*embeddings*) codant l'identité du locuteur, permettant ainsi aux mouvements faciaux générés de refléter le style unique de chaque personne.

Côté décodeur, il utilise les représentations de l'audio, l'identité du locuteur et les mouvements faciaux précédents pour prédire, de manière séquentielle, les prochains mouvements faciaux. Il se compose de deux modules principaux : le *biased causal multi-head (MH) self-attention*, qui aide à gérer de longues séquences audio, et le *biased cross-modal multi-head (MH) attention*, qui se concentre sur la synchronisation correcte de l'audio avec les mouvements faciaux. Ce processus est autorégressif, ce qui garantit que chaque prédiction est basée sur les précédentes, garantissant la cohérence et la fluidité de l'animation.

Comparé au modèle Meshtalk, qui décompose l'audio en composants indépendants avant de les combiner pour générer directement un maillage facial, Faceformer adopte une approche séquentielle, générant des mouvements faciaux étape par étape. Alors que Meshtalk se concentre sur la précision de la représentation désentrelacée et la génération directe du maillage, Faceformer est conçu pour gérer de longues séquences audio grâce à ses mécanismes d'attention spécialisée.

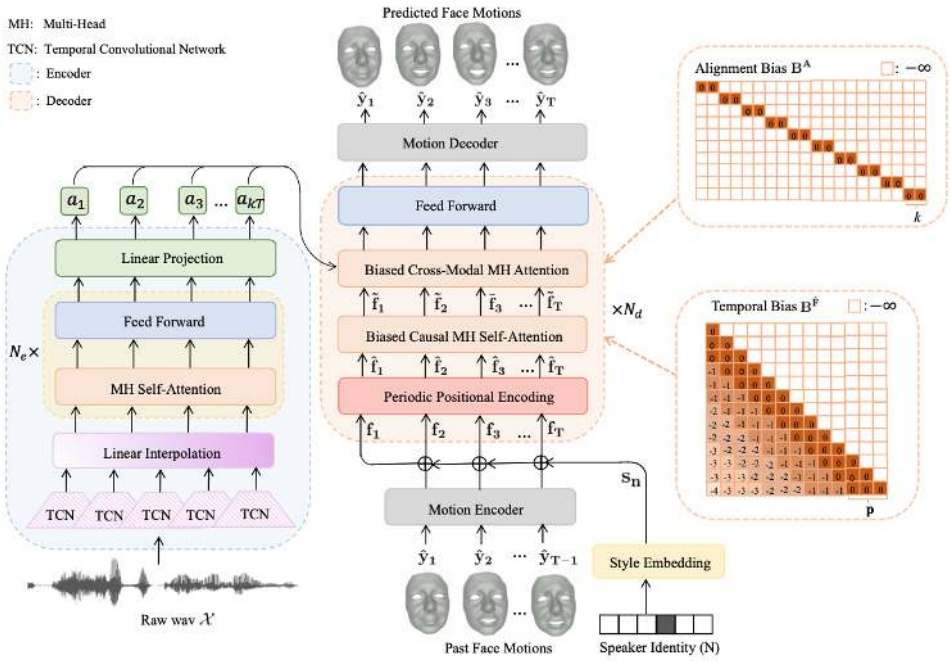


FIGURE 1.6 – Architecture de FaceFormer

1.3.4 Données utilisées

Pour le modèle MeshTalk, l'entraînement du modèle a été réalisé en utilisant un ensemble de données appelé Multiface. Il s'agit d'enregistrements de haute qualité du visage de 13 identités. Tous ont été capturés sur une scène de capture multi-vue effectuant diverses expressions faciales. De plus, il fournit des images brutes capturées de chaque vue de la caméra à une résolution de 2048 × 1334 pixels, des maillages suivis incluant les poses de tête, des textures déroulées à 1024 × 1024 pixels, des métadonnées comprenant des calibrations de caméra intrinsèques et extrinsèques, et de l'audio [Wuu et al., 2022]. Nous avons utilisé 12 entités, pour chacune 50 audios et leurs séquences d'expressions faciales respectives.

L'autre ensemble de données utilisé pour tester les projets de la deuxième phase est le dataset BIWI [Fanelli et al., 2013]. Ce dataset contient environ 15000 images de 20 personnes (6 femmes et 14 hommes). Pour chaque image, il fournit une image en profondeur, l'image RGB correspondante (toutes deux de 640 × 480 pixels) et l'annotation. La plage des poses de tête couvre environ ±75 degrés en lacet et ±60 degrés en tangage. Cependant, dans ce cas, il a seulement été nécessaire d'utiliser le modèle pré-entraîné.

1.4 Travail réalisé

Dans cette section, nous décrivons le travail réalisé jusqu'à présent pendant l'alternance. Plus précisément, la méthodologie employée sera détaillée, laquelle se compose de cinq étapes itératives. Ces étapes peuvent être ajustées ou même omises selon le jugement ou l'avis basé sur le modèle utilisé dans chaque cas. De plus, nous présentons les progrès réalisés, en fournissant un résumé des tâches accomplies et de leur état actuel.

Enfin, nous abordons les inconvénients rencontrés, en nous concentrant sur les difficultés liées à la configuration de l'environnement local, le manque de documentation, l'utilisation adéquate de la topologie, l'expérience ou les connaissances limitées dans la gestion des outils, entre autres défis. Cependant, ces obstacles ont été surmontés grâce à la collaboration et à une recherche approfondie, bien qu'ils aient influencé le développement du projet.

1.4.1 Méthodologie

Comme mentionné précédemment, la méthodologie se compose idéalement de cinq étapes. Cependant, ces étapes peuvent être adaptées en fonction des besoins spécifiques du modèle utilisé dans chaque cas. En effet, certaines étapes peuvent ne pas être pertinentes ou il pourrait être plus efficace de les appliquer dans un ordre différent. Les étapes sont énumérées ci-dessous avec une brève description de chacune :

1. **Recherche et documentation :** Cette étape se concentre sur la recherche et la sélection des modèles, suivies de la mise en œuvre des cas d'utilisation. L'objectif est de comprendre leur fonctionnement et d'analyser comment ils peuvent être intégrés à nos systèmes.
2. **Tests et intégration :** Elle consiste à utiliser le modèle avec l'un de nos personnages pour comparer les résultats avec la technique actuelle. Il s'agit d'identifier les limites du modèle et les éventuels problèmes de cohésion entre l'animation et le lipsync.
3. **Re-entraînement :** Cette étape implique de réentraîner le modèle en utilisant nos propres données. Cela peut inclure la collecte de données, l'entraînement du modèle et la réalisation de tests supplémentaires avec nos personnages.
4. **Changement de modèle :** Sur la base des résultats obtenus ou des améliorations nécessaires, il sera procédé à la modification de l'architecture du modèle pour obtenir de meilleurs résultats avec notre jeu de données.
5. **Intégration :** En tenant compte des besoins de l'utilisateur final, une interface utilisateur sera conçue dans un moteur de jeux vidéo tel que Unity ou Unreal Engine pour intégrer notre système.

Ensuite, le processus réalisé dans les deux projets utilisés pendant l'alternance sera décrit. Il convient de noter que la phase de *recherche et documentation* a été menée dans les deux cas. Étant donné que la procédure était identique dans les deux situations, centrée sur la révision de la documentation et la compréhension du modèle à utiliser, il n'est pas nécessaire d'en détailler davantage dans les sections correspondantes.

1.4.2 MeshTalk

Tests et intégration

Après avoir compris le fonctionnement du modèle, il est important de voir les limitations et les problèmes qui peuvent survenir lors de l'intégration. Dans ce cas, en ce qui concerne la cohésion entre l'animation faciale et le lipsync. Par conséquent, pendant cette phase, des tests de génération d'animations sont effectués avec nos personnages mais avec le modèle pré-entraîné fourni par le projet.

Le modèle nécessite un nombre spécifique de sommets (6172) dans le maillage pour pouvoir réaliser l'animation. Pour cette raison, il a été nécessaire d'ajuster le nombre de sommets en le réduisant. L'algorithme ICP présent dans le programme r3ds wrap a été utilisé. Cela a permis d'obtenir le mouvement attendu des lèvres.

Re-entraînement

En voyant qu'il est possible d'utiliser d'autres maillages et d'avoir obtenu un résultat avec lequel il est possible de travailler, nous procédons au re-entraînement du modèle en utilisant autres données.

Changement de modèle

À cette étape, avec le modèle déjà entraîné, des tests de génération d'animations commencent. Si nécessaire, des changements sont apportés au modèle et les résultats sont analysés. Actuellement, nous en sommes à cette phase. Nous avons généré des animations avec les modèles obtenus, cependant, l'animation ne présente pas de mouvement notable des lèvres.

Initiallement, nous avons généré un tableau de bord *TensorBoard* pour analyser le comportement du modèle à ce moment-là. Comme le montre la Figure 1.7, le modèle apprend correctement et à la bonne vitesse.

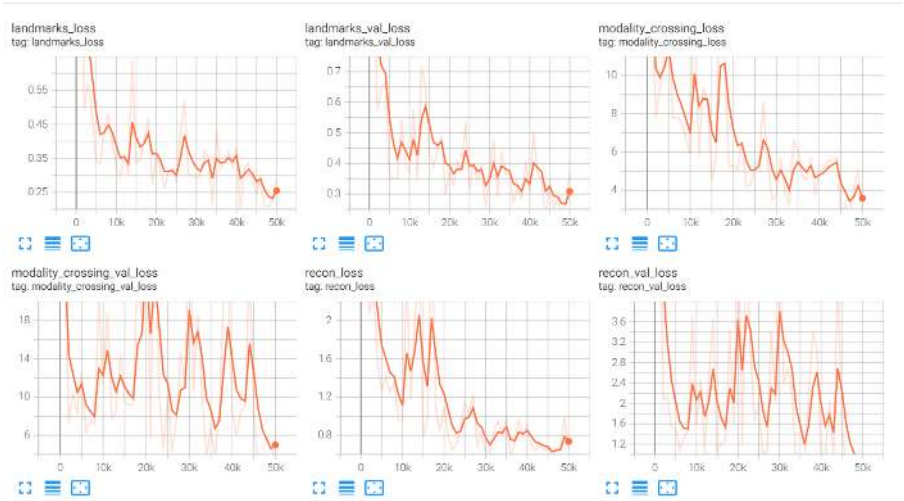


FIGURE 1.7 – Tensorboard

À partir de là, nous avons continué à analyser le code implémenté et les données. Initialement, nous avons détecté un problème lié à la taille des séquences de données audio utilisées pour l'animation des maillages faciaux, car elles variaient en longueur. Cela nécessitait des ajustements pour uniformiser les séquences. Pour résoudre ce problème, nous avons ajouté des données de remplissage avec des valeurs de $(0, 0, 0)$ aux sommets des maillages lorsque l'audio était plus court, afin que toutes les séquences aient une longueur uniforme.

Cependant, lorsque l'audio contenait du silence ou était vide, le maillage se positionnait également en $(0, 0, 0)$. Cela rendait difficile pour le modèle d'interpréter correctement la structure des mouvements faciaux, entraînant un visage neutre avec un squelette comprimé en fonction de l'audio. Pour remédier à cela, nous avons décidé de réinitialiser les valeurs du maillage à sa position initiale pendant les périodes de silence.

En lien avec ce qui précède, un autre problème est survenu. Nous avons observé un saut brusque dans la position des maillages entre les moments où un son était émis et les périodes de silence. Ce saut provoquait une absence de mouvement dans le maillage en raison d'un manque de continuité dans un modèle qui dépend de cette continuité pour apprendre. La solution proposée était de réaliser une transition *smooth* (lerp) entre la dernière position connue et la position de repos pendant le remplissage. Cependant, en raison de l'incohérence dans le remplissage (car nous n'ajoutions pas la même quantité de silence à la fin de chaque séquence), il était impossible d'établir une base temporelle stable pour cette interpolation. Par conséquent, cette solution n'a pas été mise en œuvre en raison de sa complexité mathématique, et il a été décidé de continuer avec un autre modèle d'IA.

Progrès

Dans le tableau 1.2 sont enregistrés les temps consacrés à chaque activité ainsi que les étapes qui n'ont pas été complétées. Étant donné que les résultats du réentraînement du modèle n'ont pas été satisfaisants, il a été décidé d'essayer un autre projet. En conséquence, les étapes restantes n'ont pas été réalisées, mais des connaissances précieuses ont été acquises, qui seront utiles pour l'implémentation du modèle suivant.

Tâches	**Jours**	**État**
Configuration et compréhension des outils de l'entreprise	10	Terminé
Choisir le projet	1	Terminé
Recherche et documentation	10	Terminé
Tests et intégration	19	Terminé
Re-entraînement	38	Commencé
Changement de modèle	9	Commencé (Bloqué)
Intégration avec Unity ou Unreal Engine	0	Pas commencé

Table 1.2 – Tâches

Inconvénients

Voici quelques-uns des inconvénients rencontrés lors de la mise en œuvre du projet Meshtalk. Initialement, la configuration dans un environnement local s'est avérée assez difficile. L'installation de CUDA et de Torch est toujours assez compliquée car leurs versions doivent correspondre. En raison de cela, nous avons décidé d'utiliser Google Colab pour commencer les tests. Cependant, comme il s'agit d'un environnement peu flexible pour un projet aussi important que celui-ci, nous avons décidé de réinstaller les bibliothèques sur le serveur de l'entreprise.

De plus, beaucoup de temps a été consacré à la réduction des sommets pour utiliser le modèle pré-entraîné. Nous avons choisi d'utiliser l'algorithme ICP (Iterative Closest Point), qui aligne progressivement deux ensembles de points en identifiant, pour chaque point d'un ensemble, le point correspondant le plus proche dans l'autre. Cependant, la compréhension de son fonctionnement a rendu son développement difficile. Heureusement, avec l'aide d'un expert dans le domaine, nous avons pu résoudre le problème car il connaissait un outil permettant d'effectuer ce processus facilement.

Finalement, une autre problème majeur a été le manque de documentation concernant l'implémentation du DataLoader pour l'entraînement du modèle, ainsi que sur le modèle en général. Malgré la réalisation de multiples tests avec divers paramètres et les tentatives de contacter l'auteur du projet et d'autres utilisateurs ayant obtenu de bons résultats, aucune réponse utile n'a été reçue pour aider à résoudre le problème. En conséquence, il n'a pas été possible de trouver une solution concrète à ce désagrément.

1.4.3 FaceFormer

Tests et intégration

Pour les tests, des animations ont été générées avec un même personnage en utilisant différents audios, dans le but d'analyser la qualité des expressions faciales. Le modèle pré-entraîné sur le jeu de données BIWI a été utilisé avec les poids *(weights)* inclus dans le projet.

Pour réaliser ces tests, il a été nécessaire de modifier certains fichiers, notamment la façon dont la vidéo était rendue. Cela était dû au fait que le projet utilisait une bibliothèque qui n'était pas compatible avec le serveur où le modèle a été déployé. Par conséquent, il a été décidé d'utiliser une bibliothèque alternative *(trimesh)* qui permettait son exécution sans problème.
De plus, la topologie de notre mesh a dû être adaptée à celle du jeu de données BIWI. Ce processus a été réalisé de manière similaire au projet précédent.

Avec ces modifications, nous avons procédé à des tests depuis Blender. Nous avons utilisé Rigify pour générer un rig facial du personnage, afin d'obtenir certains points de référence (spécifiquement les os) pouvant être liés aux points du mesh utilisé dans le modèle. Cette méthode d'animation *(rigging)* consiste à créer une structure squelettique permettant de manipuler et d'animer un modèle 3D. Ce processus inclut l'utilisation du weight painting, qui attribue des influences spécifiques aux différents os du rig, déterminant ainsi comment chaque partie du

maillage se déforme en réponse aux mouvements des os. Ces deux étapes sont essentielles pour assurer un mouvement fluide et naturel du mesh. Le résultat avec le rig est montré dans la Figure 1.8.

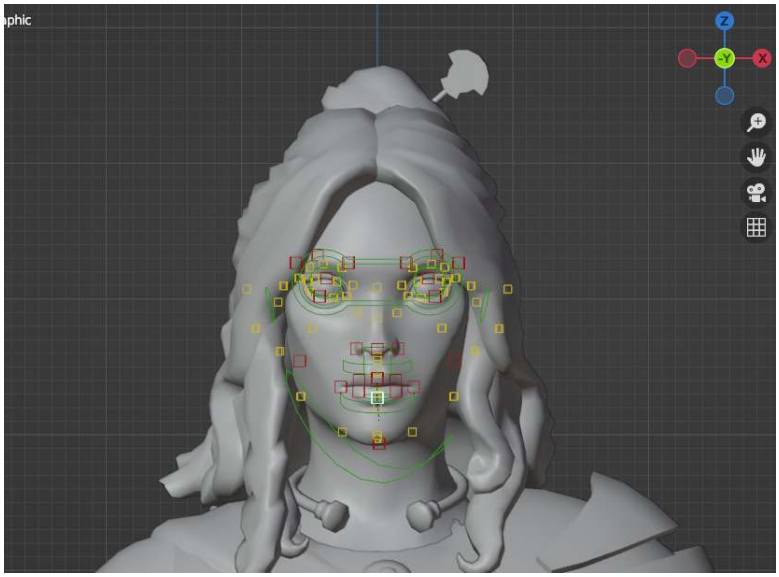


FIGURE 1.8 – Personnage avec rig

Étant donné que le processus génère un fichier .npy contenant les informations des mouvements de chaque point du mesh sur un nombre déterminé de frames, il a seulement été nécessaire de traiter ces données. Compte tenu de la grande quantité de points du maillage, ce qui aurait augmenté la complexité et le temps de traitement pour la création des courbes d'animation dans Unity et Unreal, il a été décidé d'utiliser environ 60 points. Ces points correspondent à des landmarks situés dans des zones clés qui permettent de capturer avec précision les mouvements faciaux les plus importants et assurer des mouvements fluides. Ainsi, ont été inclus la pointe du nez, le contour des yeux, le contour du visage, les sourcils et les lèvres. Cependant, la possibilité d'ajouter davantage de landmarks à l'avenir n'est pas exclue, afin d'évaluer si cela améliorerait la qualité de l'animation et quel impact cela aurait sur la vitesse de traitement.

À partir de cela, il a été décidé de créer un fichier permettant de relier chaque point à un point de référence sur le visage du personnage. Un exemple de la liste est montré en 1.1.

Listing 1.1 – Example liste des landmarks (sources/rig_info.csv)

```
bone name,point,bone_parent
brow.B.L,17510,ORG-face/master_eye.L
brow.B.L.001,3141,ORG-face/master_eye.L
brow.B.L.002,3245,ORG-face/master_eye.L
brow.B.L.003,5987,ORG-face/master_eye.L
brow.B.L.004,14903,ORG-face/master_eye.L
lid.B.L,6075,ORG-face/master_eye.L
```

Avec les données filtrées, les informations ont été organisées dans un fichier .npz, permettant de stocker des données dans un format similaire à un dictionnaire. Dans ce fichier, deux clés principales ont été utilisées pour sauvegarder les informations : "bones", qui correspond à la liste des os ou landmarks, et "frames", où la liste des coordonnées pour chaque point est stockée. Dans Blender, un script simple a été créé pour effectuer les tâches suivantes :

1. Lire le fichier .npz.
2. Itérer sur la liste des os du rig.
3. Trouver l'os correspondant dans la liste.
4. Itérer sur la liste des frames de l'os actuel.
5. Calculer une valeur delta, qui correspond à la différence entre les coordonnées actuelles et celles de la frame précédente pour cet os.
6. Appliquer cette valeur à la propriété de translation de l'os correspondant.
7. Insérer une keyframe dans la ligne de temps avec la nouvelle valeur.

Lors de l'exécution de ce script, un mouvement du mesh a été observé, mais il s'est révélé quelque peu irréaliste, notamment sur l'axe y des points de la bouche, où il a été remarqué que ces points dépassaient légèrement les limites normales. Sur l'image 1.9, on peut voir ce problème. Pour contrôler ces mouvements, il a été décidé de valider que la position finale après l'application de la translation ne dépasse pas certaines limites pour les deux lèvres. Pour la lèvre supérieure, la position ne devait pas être inférieure ou égale à 0, et pour la lèvre inférieure, elle ne devait pas être supérieure ou égale à 0.

Figure 1.9 – Problème dans blender

Avec ce changement, un bon résultat et un bon fonctionnement ont été obtenus, donc il a été décidé de poursuivre l'implémentation dans Unity.

Intégration avec Unity

Nous avons procédé à l'intégration du modèle dans Unity. Pour cela, une API a été développée en utilisant FastAPI dans un environnement WSL avec Ubuntu. L'idée est que, lors de l'appel de l'endpoint depuis Unity, la fonction du modèle qui génère l'animation soit exécutée. Ensuite, les coordonnées des points sélectionnés initialement sont envoyées au format JSON. La structure de l'API est illustrée dans 1.2, où l'on trouve une liste d'os *(bones)*, chacun avec trois clés :

- **name :** Une chaîne de caractères correspondant au nom de l'os.
- **frames :** Une liste de listes contenant les coordonnées (x, y, z) pour chaque frame.
- **parent_name :** Une chaîne de caractères indiquant la position de l'os dans la structure du rig.

Listing 1.2 – Réponse JSON (sources/response.json)

```
{
    "bones": [
        {
            "name": "brow.B.L",
            "frames": [
                [0.15244612097740173,-0.094734787940979,0.004913059528917074],
                [0.15244699453378652,-0.09449824032845436,0.004888475987479672],
                [0.15254293536985075,-0.09454105504147418,0.004748030916436926]
            ],
            "parent_name": "ORG-face/master_eye.L"
        },
        {
            "name": "brow.B.L.001",
            "frames": [
                [0.13313128054141998,-0.10727313905954361,-0.017636945471167564],
                [0.13303770976407187,-0.10684168812903491,-0.017716090037644683],
                [0.13304299851516624,-0.10671413235076062,-0.017816267157723376]
            ],
            "parent_name": "ORG-face/master_eye.L"
        }
    ]
}
```

En Unity, nous avons effectué un appel à l'API pour recevoir les données et créer un clip d'animation. Le processus de calcul de la position et de son application aux os correspondants a été similaire à celui utilisé dans Blender, ce qui a facilité une mise en œuvre rapide. Par la suite, il a seulement été nécessaire d'ajuster certains paramètres, de résoudre des problèmes liés à l'échelle et à l'orientation du mesh (qui seront détaillés après), et d'ajouter le son.

Progrès

Dans le tableau 1.3, vous pouvez voir le temps consacré à chacune des activités lors de l'implémentation de ce modèle. En général, plus de temps a été consacré aux tests et à l'intégration avec Unity.

Tâches	**Jours**	**État**
Choisir le projet	1	Terminé
Recherche et documentation	5	Terminé
Tests et intégration	30	Terminé
Re-entraînement	0	Pas commencé
Changement de modèle	0	Pas commencé
Intégration avec Unity ou Unreal Engine	35	En cours

Table 1.3 – Tâches

De plus, une nouvelle étape sera lancée, consistant en la capture de nos propres séquences par des techniques de capture de mouvement. Pour entraîner le modèle, deux types de données d'entrée sont nécessaires : l'audio et les séquences d'images ou de vidéos correspondantes. Par conséquent, nous prévoyons d'utiliser une caméra et un microphone pour capturer les informations nécessaires. Voici le processus proposé :

1. **Scan du visage du sujet de test :** Un scan du visage est réalisé, qui peut être de basse résolution, suffisant pour obtenir un nuage de points. Bien qu'il soit possible d'utiliser la photogrammétrie, cette technique peut être longue et fournir une qualité trop élevée pour nos besoins.
2. **Traitement du nuage de points :** Le nuage de points est traité pour l'ajuster à un modèle de base (template mesh). Cet ajustement inclut l'ajout de caractéristiques spécifiques pour créer un modèle (mesh) personnalisé. Pour cette phase, l'algorithme ICP (Iterative Closest Point) peut être utilisé.
3. **Dessin des repères (landmarks) :** Avant de commencer les enregistrements, il est crucial de marquer les repères importants sur le visage du sujet. Cela facilitera l'application et le traitement ultérieur des données.
4. **Enregistrements de test :** Le nombre de phrases et de personnes pour la capture vidéo est défini. En général, environ 12 personnes sont enregistrées, chacune prononçant environ 40 phrases de 3 à 4 secondes chacune.
5. **Application des coordonnées au maillage personnalisé :** Avec les coordonnées obtenues des repères, les valeurs de chaque frame sont appliquées au mesh personnalisé pour générer un nouveau mesh avec les positions correspondantes.

Pour chaque sujet, une séquence de maillages sera générée, ce qui évitera la nécessité de scans et d'ajustements continus, comme c'est souvent le cas dans d'autres projets. Il est important de noter que ce processus devra être répété pour chaque sujet de test. De plus, il convient de souligner que cette proposition est initiale et donc sujette à des modifications si une méthode plus efficace est trouvée pour sa mise en œuvre. Des outils comme Kinect ou ARKit pourraient faciliter le travail et simplifier quelques étapes de l'implémentation.

Inconvénients

Initialement, l'un des problèmes rencontrés était la manipulation du rig. Nous avons décidé de le faire manuellement, mais cette approche s'est révélée plus lente que prévu en raison de problèmes de weight painting. Les points de chaque os n'étaient pas correctement assignés, et certains points ne correspondaient pas à l'os approprié.
Pour résoudre ce problème, nous avons recherché une solution alternative et avons opté pour l'utilisation d'un outil appelé FaceIt, qui génère automatiquement le rig avec seulement quelques ajustements sur les landmarks du mesh. Cependant, lors de la tentative d'exportation du mesh avec le rig depuis Blender, FaceIt ne le permettait pas, ce qui nous a contraints à configurer un nouveau rig. Cette fois, nous avons choisi Rigify, qui a permis l'exportation du rig sans problème.

Un autre défi est survenu lors de l'utilisation du rig dans Unity. Le mesh était lié à deux rigs : un facial et un pour le corps. Cependant, Unity ne permet pas d'utiliser les deux rigs simultanément pour exécuter l'animation. Nous avons donc décidé de conserver uniquement le rig facial et de supprimer le rig corporel pour poursuivre les tests. Par la suite, il sera nécessaire d'aborder ce problème en unissant l'articulation de la tête avec celle du corps pour maintenir un seul rig. De plus, lors de la création de l'animation clip, les os n'étaient pas présents dans le projet. Nous avons donc ajouté au fichier de la liste des os, 1.1, le chemin du parent pour spécifier la position de chaque os dans la structure du rig.

Enfin, nous avons rencontré des problèmes avec l'échelle et l'orientation des axes. En raison d'un échelle incorrect, les points du mesh montraient des mouvements excessifs ou insuffisants. Cela a été résolu en utilisant une configuration spécifique lors de l'exportation du mesh depuis Blender. Quant à l'orientation, étant donné que trois outils différents sont utilisés —le modèle, Blender et Unity— les axes peuvent être définis de manière différente. Pour y remédier, nous avons réalisé des tests dans un environnement contrôlé avec un seul os, ce qui nous a permis d'ajuster les valeurs des axes. Par exemple, les valeurs de l'axe z dans Blender pourraient correspondre à l'axe y dans Unity. Cette approche a non seulement résolu le problème d'orientation, mais a également facilité le débogage, nous permettant d'identifier des erreurs dans le script Python liées au traitement des données.

1.5 Conclusion

Ma mission au sein de l'entreprise X&Immersion est de développer un outil permettant aux studios de jeux vidéo de réduire leur charge de travail tout en obtenant les meilleurs résultats dans la génération d'animations pour leurs personnages.

Actuellement, le projet se trouve à une phase prometteuse. La génération d'animations prend environ 30 secondes, ce qui est un bon indicateur. Cependant, il serait utile de revoir l'architecture pour évaluer s'il est possible d'améliorer les performances. Une autre amélioration souhaitée concerne la qualité de l'animation, en se concentrant sur la précision et le réalisme. Pour y parvenir, il serait important d'investir dans la création d'un ensemble de données propre, en utilisant des techniques avancées telles que la capture de mouvement. Cela permettrait d'entraîner les modèles avec des données plus représentatives et spécifiques au projet.

Au cours du processus, j'ai rencontré divers défis, tels que la configuration initiale de l'environnement de développement, l'intégration de modèles pré-entraînés dans nos systèmes, et l'optimisation des performances du modèle, en plus de problèmes plus techniques et liés aux connaissances. Cependant, grâce à mon effort et le travail d'équipe, il était possible de surmonté ces obstacles et progressé de manière significative dans le développement du projet. Les difficultés que j'ai rencontrées ont enrichi mon expérience et mes compétences techniques, posant ainsi les bases pour les futures étapes de développement.

Bibliographie

[Cudeiro et al., 2019] Cudeiro, D., Bolkart, T., Laidlaw, C., Ranjan, A., and Black, M. (2019). Capture, learning, and synthesis of 3D speaking styles. *Computer Vision and Pattern Recognition (CVPR)*, pages 10101–10111.

[Fan et al., 2022] Fan, Y., Lin, Z., Saito, J., Wang, W., and Komura, T. (2022). Faceformer : Speech-driven 3d facial animation with transformers.

[Fanelli et al., 2013] Fanelli, G., Dantone, M., Gall, J., Fossati, A., and Van Gool, L. (2013). Random forests for real time 3d face analysis. *Int. J. Comput. Vision*, 101(3) :437–458.

[Haque and Yumak, 2023] Haque, K. I. and Yumak, Z. (2023). Facexhubert : Text-less speech-driven e(x)pressive 3d facial animation synthesis using self-supervised speech representation learning.

[Pham et al., 2018] Pham, H. X., Wang, Y., and Pavlovic, V. (2018). End-to-end learning for 3d facial animation from speech. In *International Conference on Multimodal Interaction*.

[Stan et al., 2023] Stan, S., Haque, K. I., and Yumak, Z. (2023). Facediffuser : Speech-driven 3d facial animation synthesis using diffusion.

[Wuu et al., 2022] Wuu, C.-h., Zheng, N., Ardisson, S., Bali, R., Belko, D., Brockmeyer, E., Evans, L., Godisart, T., Ha, H., Huang, X., Hypes, A., Koska, T., Krenn, S., Lombardi, S., Luo, X., McPhail, K., Millerschoen, L., Perdoch, M., Pitts, M., Richard, A., Saragih, J., Saragih, J., Shiratori, T., Simon, T., Stewart, M., Trimble, A., Weng, X., Whitewolf, D., Wu, C., Yu, S.-I., and Sheikh, Y. (2022). Multiface : A dataset for neural face rendering. In *arXiv*.

[Xing et al., 2023] Xing, J., Xia, M., Zhang, Y., Cun, X., Wang, J., and Wong, T.-T. (2023). Codetalker : Speech-driven 3d facial animation with discrete motion prior.