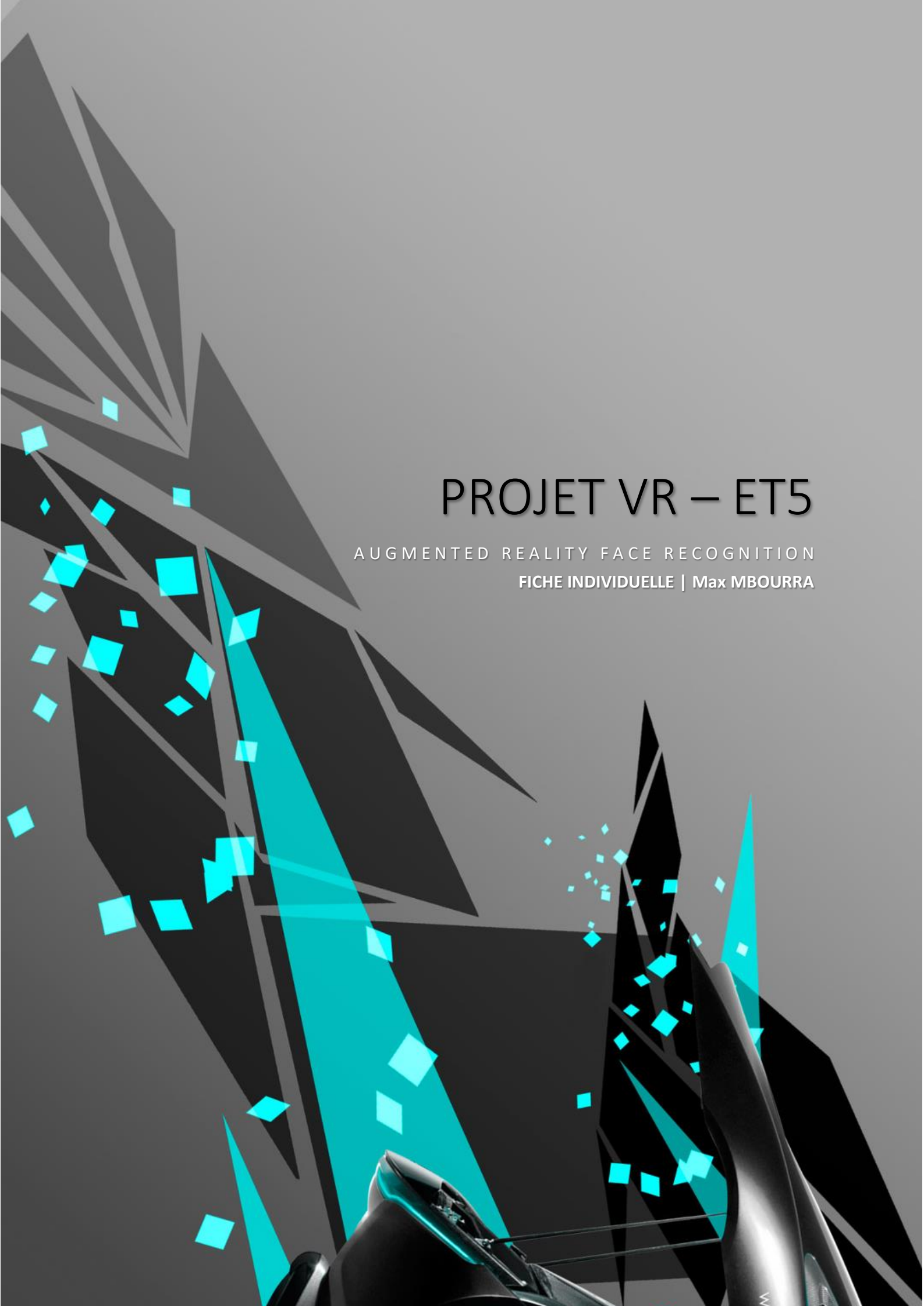

PROJET VR – ET5

AUGMENTED REALITY FACE RECOGNITION

FICHE INDIVIDUELLE | Max MBOURRA

INTRODUCTION

A l'heure actuelle, les applications faisant usage de réalité augmentée prennent une place croissante au sein du quotidien de chacun. En effet, quasiment chaque appareil actuel disposant d'une caméra a au moins une application en réalité augmentée, que ce soit les smartphones avec les filtres photos/vidéos intelligents de Snapchat, les jeux XBOX 360 exploitant la Kinect ou, plus récemment, les appareils HoloLens, se vouant exclusivement à des applications en réalité augmentée.

C'est avec la volonté de développer un programme pouvant s'insérer dans ce contexte que le projet présenté lors de la soutenance est une application de Réalité Augmentée, simplement nommée AUGMENTED REALITY FACE RECOGNITION. Le but de cette application est d'exploiter les caméras des appareils pour effectuer des tâches de reconnaissance, telles que la reconnaissance d'émotions ou de visage. Initialement prévue pour un déploiement uniquement sur HoloLens, le projet a connu d'importantes difficultés qui ont contraint l'équipe à repenser le mode de réalisation ainsi que le fonctionnement même de l'application, ce qui a mené à l'élaboration de trois applications indépendantes (Locale, Vuforia et HoloLens) usant d'un même Web Service.

Le groupe chargé de mettre en place ce projet se constitue de AZEMARD Thomas, BRINDAMOUR Benjamin, MBOURRA Max, SHI Yao et WOZNICA Anthony.

TRAVAIL REALISE

Peu après les débuts du projet, nous avons constaté qu'il n'était pas possible de réaliser ce que nous voulions uniquement en C# avec les librairies standard. Par conséquent, nous nous étions séparé les tâches, de manière à pouvoir avancer indépendamment les uns des autres.

Nous avions en premier temps conçu l'idée d'un HoloLens connectée par protocole TCP avec un serveur Python, de manière à faire passer le flux vidéo de l'HoloLens vers le serveur Python à la manière d'une Webcam. Cependant, l'ensemble de nos tentatives de connexion TCP entre le server python et une application C# ont été pendant un long moment un échec. Nous n'avons pas eu de meilleur résultat en utilisant les fonctionnalités de connexion d'Unity.

Nous avons réparti le groupe en plusieurs parties afin d'avoir une application présentable pour la soutenance. C'est pourquoi, je me suis orienté sur une application mobile en utilisant la plateforme Vuforia sur Unity. Le tableau ci-dessous résume l'ensemble des tâches que j'ai réalisé durant le projet

Tâche	Solution apporté
Connexion TCP entre client C# et server python	Je n'ai pas trouvé la solution pour la connexion avec l'hololens . Une grande quantité de recherche
Connexion TCP entre client C# et server python sur une application android, Vuforia	La connexion a pu être établi en utilisant la classe TcpClient du langage C# du namespace System.Net.Sockets
Déploiement d'une application android de réalité augmenté	Pour le déploiement de l'application, j'ai utilisé le building d'Unity.Cependant cette solution conduit à l'installation d'android studio. Car pour le déploiement nous avons du sdk-android qui est présent avec android studio. Pour avoir une caméra supportant la réalité augmenté, j'ai installé Vuforia. L'utilisation de cette caméra est obligatoire, car Unity n'a pas des disposition pour une application de réalité augmenté . Les caractéristiques de cette plateforme est déployable sur plusieurs supports
Mettre des boutons, des champs textuels et des inputs sur la caméra	Pour mettre en place une interface sur la caméra de réalité augmenté, j'ai utilisé l'objet canvas
Faire connecter l'application à n'importe quelle machine faisant tourner le server python en local	Proposer d'utiliser la seconde adresse Ip de la machine dans le réseau. Cela permet se connecter à la machine présente dans le même réseau.
Mettre en place une connexion web service entre l'application Unity et le we service python	Unity gère assez bien les connexion aux web service. Avec les classe suivantes: WWWForm, pour le contenu de l'envoi et UnityWebRequest pour le canal de communication.
Capture continue du flux image	Pour la capture de l'image de façon continue. J'ai implémenté une

	fonctionnalité permettant d'enregistrer une image dans un objet Texture2D et qui envoie cette capture au server. Pour laisser le temps au server de réaliser l'analyse la capture se fait à chaque 100 frame

Quelques remarques observées

La gestion des actions asynchrones ne sont pas gérées, de la même façon entre une application C# et Unity. J'ai appris que pour ces actions asynchrone, Unity utilise la fonction StartCoroutne(). Cette fonction prend en argument une fonction qui nous retourne un IEnumerator. C'est avec ces 2 éléments que nous avons pu établir une connexion au web service qui nous retourne une réponse. Car sans, nous avons bien l'envoie de l'image mais nous ne recevons pas la réponse du web service.

Le web service nous retourne les résultats de l'analyse en sous le format d'un JSON. Unity possède une fonction pour gérer les objets en forme JSON. Cependant, cet objet ne fonctionne pas si bien. C'est pour cela que nous avons utilisé l'asset JSONObject qui gère mieux les objets de format JSON.

Une application Unity déployé sur android ne sait pas où enregistrer les photos. Pour combler ce défaut, nous avons utilisé l'Asset NativeGallery. Cependant nous avons décidé de ne pas garder les cette fonctionnalité. Dans le but de ne pas prendre plus de mémoire sur le smartphone et ralentir l'application.