CONAV : Rapport Vuforia

Initiation à Vuforia et création d'une application de Réalité Augmentée

Alexis DUPUIS & Loïc HENTZ

Option Réalité Virtuelle

Sommaire

1 Introduction

Lors d'un TP de VSION, nous avons déjà eu l'occasion de manipuler et de créer une première application de réalité augmentée, à l'aide d'ArUco. Les modèles que l'on utilisait pour l'application étaient cependant très simples, puisque modélisés à partir d'OpenSceneGraph. L'outil présenté aujourd'hui présente l'avantage d'être facilement compatible avec Unity. Vuforia est un framework permettant de développer facilement des applications de RA simples. L'intérêt de Vuforia par rapport à ArUco est que l'on utilise des images au lieu de marqueurs. Il est plus simple de proposer ses propres images en guise de marqueurs (même si, on le verra, celles-ci doivent répondre à certaines contraintes pour être correctement exploitables par Vuforia). L'objectif de ce projet est de suivre un TP d'initiation à l'utilisation de Vuforia sur des exemples simples, puis de proposer une application personnalisée illustrant ce que l'on a retenu de Vuforia.

2 Retour sur le TP d'initiation

Le tutoriel proposé est extrêmement détaillé, et permet de se familiariser avec Vuforia sans grandes difficultés. On est vraiment pris par la main tout le long du TP. Toutefois, certains détails semblent inutiles, notamment de longs paragraphes sur le fonctionnement de Unity, et certaines manipulations très basiques n'ont, selon nous, pas besoin d'être décrites (comme dézipper le dossier *AugmentationAsssets*, en faisant attention d'être dans l'onglet *Models* de Unity, ou tout le détail sur la récupération du modèle de l'astronaute, avec le parcours des fichiers pour le trouver, le *drag and drop*, et faire attention qu'il soit bien en enfant de l'*Image target...*). En somme, il y a beaucoup de détails qui peuvent s'avérer utiles lorsque l'on n'est pas familier avec le fonctionnement de Unity, mais qui s'avèrent inutiles pour nous en cette fin d'année (surtout avec le projet Unity effectué en parallèle).

D'autre part, certaines précisions manquaient :

- Modifier l'image affichée dans un *Image target* (dans le script *Image Target Behaviour*).
- Accéder aux différentes caractéristiques des *GameObjects* attachés à une *Image target* (simplement en appelant le *GameObject* associé.
- Comment l'échelle influence les relations de distance entre 2 *GameObjects* issus de 2 *Image target* différentes. En effet, le rapport de taille entre le marqueur-image dans Unity et sa taille réelle sert à déterminer une échelle pour Vuforia. Ainsi, Vuforia travaille avec une échelle par objet plutôt qu'une échelle globale et cela crée souvent des comportements inattendus.

3 Notre application personnalisée

3.1 Objectif de l'application

Notre application s'inspire de plusieurs jeux orientés sur la collecte de minerais (*Deep Rock Galactic*, *Space Engineers*). La scène est constituée de 3 images correspondant à 3 minerais (un grolem, l'affiche du jeu *No Man's Sky* et l'acteur Dwayne Johnson ("the rock")), d'une image d'astronaute et d'une image de

quadcoptère (pour le drone).

FIGURE 1 – Les minerais à ramasser et leurs marqueurs.

L'objectif de l'application est que le drone ramasse les minerais grace à un ordre donné par l'astronaute.

En pratique, l'astronaute doit s'approcher d'un minerai qu'il veut que le drone ramasse (il faut donc manuellement amener son marquer proche de celui du minerai). Une fois suffisamment proche du minerai, l'astronaute pourra faire signe en direction du drone par pression de la touche 'C' du clavier. Dans ces conditions, le drone sera capable de récupérer le rocher, et s'approchera automatiquement de celui-ci (s'éloignant donc temporairement de son marqueur). Une fois le drone suffisamment proche du minerai, ce dernier disparaîtra et le score du joueur augmentera de 1. Le minerai est "ramassé" et le robot retourne à sa base (son marqueur). On peut remettre le jeu à zéro en appuyant sur la touche R' du clavier une fois tous les minerais ramassés.

Attention : Le joueur faisant signe au drone doit rester à proximité de celui-ci le temps qu'il récupère le minerai. Si le joueur s'éloigne du minerai ciblé, le drone sera "perdu" et incapable de récupérer le minerai.

Attention : Comme expliqué plus haut, Vuforia détermine une échelle pour chaque objet en fonction de la taille de son marqueur. C'est ce qui lui permet d'identifier qu'un objet est loin par exemple, et qu'il doit être affiché plus petit. Il faut donc faire attention à utiliser le même rapport de taille entre les marqueurs dans le monde réelle que dans Unity. C'est d'autant plus important que beaucoup de nos actions se déclenche en jugeant de la proximité de 2 objets, et donc des échelles différentes donneront l'impression que certains comportements ne se déclenchent pas comme voulu.

Pour le confort de jeu, nous vous recommandons fortement d'imprimer les marqueurs que nous avons utilisés, en essayant au mieux de respecter la même échelle de tailles que sur Unity.

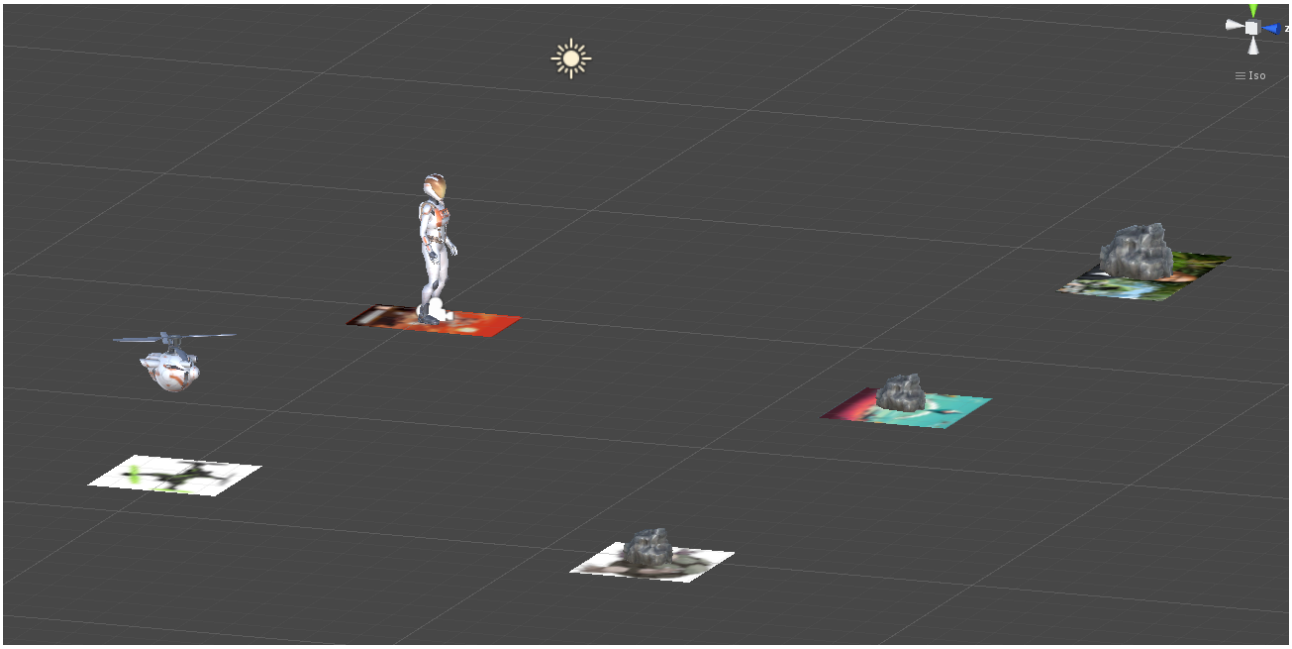

FIGURE 2 – La scène de notre application : un astronaute et son drone partent à la recherche de quelques minerais.

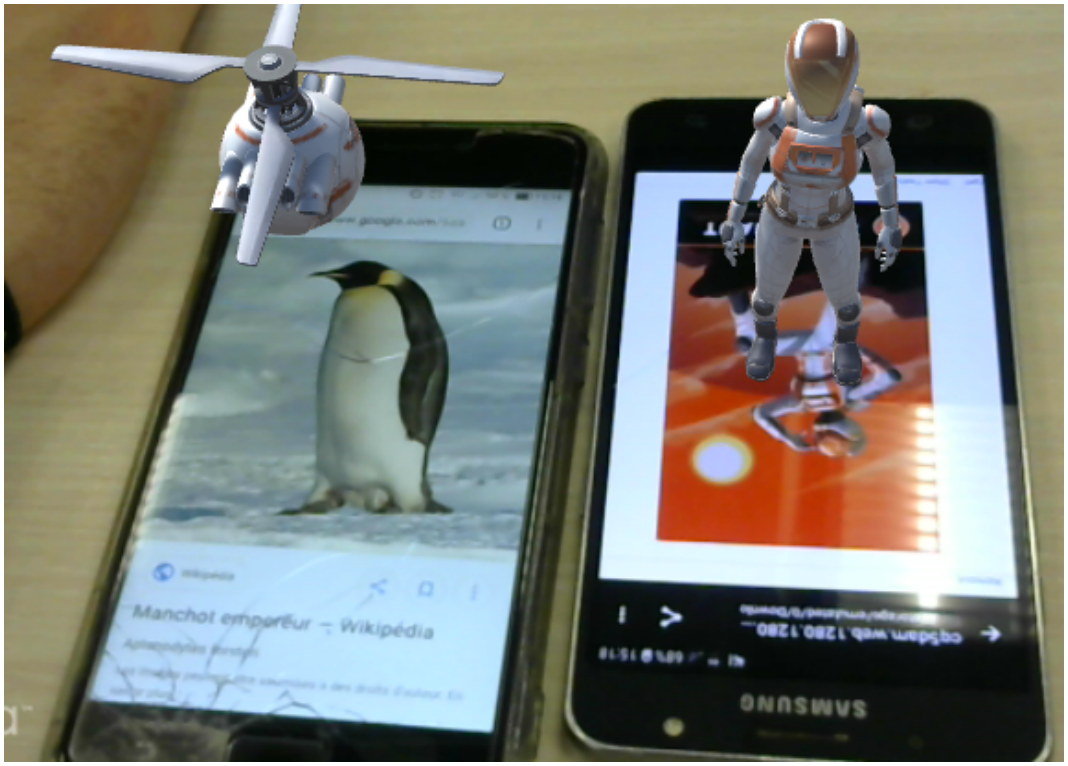

FIGURE 3 – Vuforia nous permet d'afficher nos protagonistes dans le monde réel. Remarquez que le marqueur du robot a changé en cours de projet car l'image n'était pas reconnue de manière optimale (manque de contraste).