



UNIVERSITÉ DE TECHNOLOGIE DE  
COMPIÈGNE

RV01

## Size Doesn't Matter



Automne 2017

Virgile Vançon   Pierre Bathellier   Nicolas Marcadet

# Sommaire

<b>1</b>	<b>Présentation</b>	<b>2</b>
1.1	Principe . . . . .	2
1.2	Contexte . . . . .	3
1.3	Equipe . . . . .	3
1.4	Ressources . . . . .	3
<b>2</b>	<b>Réalisation</b>	<b>5</b>
2.1	L'expérience <i>Size Doesn't Matter</i> . . . . .	5
2.2	Gameplay . . . . .	5
2.3	Mécaniques de jeu . . . . .	6
2.4	Niveaux de réalisations . . . . .	8
2.4.1	Réalisation minimale : ✓ . . . . .	8
2.4.2	Réalisation intermédiaire : ✓ . . . . .	9
2.4.3	Réalisation optimale : ✓ . . . . .	9
2.4.4	Encore plus loin . . . . .	9
2.5	Quelques difficultés rencontrées . . . . .	10

# Présentation

## 1.1 Principe

*Size Doesn't Matter* est une expérience en réalité virtuelle confrontant l'utilisateur à une illusion d'optique peu connue, pourtant expérimentée en permanence dans la vie de tous les jours : la taille apparente (ou taille angulaire). C'est le phénomène qui fait que plus un objet est loin, plus il paraîtra petit à l'œil humain (et inversement).

Selon Wikipédia :

*“La taille apparente d'un objet vu à distance est la distance angulaire entre ses points extrêmes au point d'observation, c'est-à-dire l'angle entre les droites qui relient les extrémités de l'objet et l'observateur.”*



FIGURE 1.1 – Illustration de la taille apparente dans la vie réelle

Dans notre application, l'utilisateur peut attraper tous les objets qui l'entourent, et ce, quelle que soit la distance qui les sépare. Lorsqu'un objet est sélectionné, il conserve pour l'œil de l'utilisateur sa taille apparente et non sa taille réelle. Dans l'exemple ci-dessous, la maison est attrapée et elle conserve à l'écran

la même taille qu'à l'origine. La taille réelle de l'objet une fois relâchée est la projection de la taille apparente sur l'arrière-plan. On a ici visé le sol un peu plus loin, la taille réelle de la maison a donc augmentée pour que la taille apparente reste inchangée.



FIGURE 1.2 – Illustration de la taille apparente dans *Size Doesn't Matter*

## 1.2 Contexte

Ce projet est développé dans le cadre de l'Unité de Valeur (UV) de réalité virtuelle RV01 enseignée à l'Université de Technologie de Compiègne (UTC) par Mme Thouvenin au semestre d'Automne 2017.

## 1.3 Equipe

- Pierre BATHELLIER, étudiant en 5ème semestre de Génie Informatique à l'UTC et spécialisé en systèmes et réseaux informatiques.
- Virgile VANCON, étudiant en 5ème semestre de Génie Informatique à l'UTC et spécialisé en systèmes et réseaux informatiques.
- Nicolas MARCADET étudiant en 5ème semestre de Génie Informatique à l'UTC et spécialisé dans les systèmes en temps réels et l'informatique embarquée.

## 1.4 Ressources

Le projet a été développé grâce au moteur de jeu 3D Unity, et a été porté pour le casque de réalité virtuelle HTC Vive comme périphérique de visualisation

ainsi que deux contrôleurs pour déterminer les mouvements des mains du joueurs. Les mouvements de ses mains permettent de déterminer la direction des rayons servant à la fois à la sélection et à la manipulation des objets virtuels ainsi que le déplacement du joueur. Le casque HTC Vive permet d'offrir une expérience immersive particulièrement intéressante pour l'utilisateur.



FIGURE 1.3 – Casque HTC Vive ainsi que ses deux capteurs et contrôleurs

# Réalisation

## 2.1 L'expérience *Size Doesn't Matter*

*Size Doesn't Matter* se déroule dans un univers fantastique. A l'origine, le joueur se trouve en prison. Une baguette apparaît, une voix grave le presse de la saisir et le guide à travers un niveau de tutoriel lui permettant de s'évader de sa cellule.

La voix, omniprésente au cours du scénario que nous avons choisi de développer, permet un guidage efficace de l'utilisateur tout en préservant son immersion dans notre monde (la voix faisant partie intégrante de l'intrigue).

Après son évasion, un monde ouvert se présente à l'utilisateur dans lequel il lui est possible d'expérimenter plus en profondeur le phénomène de la taille apparente avec des objets plus complexes et variés que ceux présents dans la prison : des maisons, une tour, un avion, une pyramide... Ainsi que des animaux. Si le joueur choisi de suivre le scénario, il sera même confronté à une araignée monstrueuse et devra jouer de ses pouvoirs pour changer sa taille, toujours en jouant avec le principe de la taille apparente.

Une fois la dernière baguette récupérée (qui dispose d'une portée plus grande que celle proposée à l'origine), la voix impose au joueur de sélectionner le soleil, objectif ultime de la voix qui manipulait le joueur depuis le début pour voler la lumière afin de répandre l'obscurité sur le monde.

## 2.2 Gameplay

### Mouvements du personnage

La rotation de la tête se fait évidemment par le biais du casque. L'utilisateur peut ainsi regarder dans toutes les directions.

Pour déplacer le personnage, nous avons opté pour une navigation au touchpad du contrôleur droit. Tactile, il permet un déplacement efficace dans un univers étendu tout en préservant une certaine immersion, ce qui n'est pas le cas de la technique de téléportation "pointée cliquée".



FIGURE 2.1 – Le touchpad (bouton rond au centre du controlleur)

Il est également possible de sauter en appuyant sur le bouton central du touchpad.

### **Sélection d'objets**

Le premier type d'interaction est la sélection et la manipulation des objets de l'environnement grâce à la technique de RayCasting. Les objets présents dans l'environnement virtuel peuvent être déplacés dans les trois directions de l'espace par l'utilisateur. Pour ce faire, le joueur doit pointer sa baguette (contrôleur droit) vers l'objet et cliquer la gâchette arrière. Pour relâcher l'objet, il suffit de pointer dans n'importe quelle direction et de recliquer sur la gâchette. L'objet sélectionné sera projeté vers l'arrière-plan.

## **2.3 Mécaniques de jeu**

### **Préserver la taille apparente des objets**

Lorsqu'un objet est sélectionné par l'utilisateur, sa taille est actualisée en temps réel à chaque frame en fonction de sa taille et distance précédentes par rapport à l'utilisateur. Cette taille est préservée à tout instant.

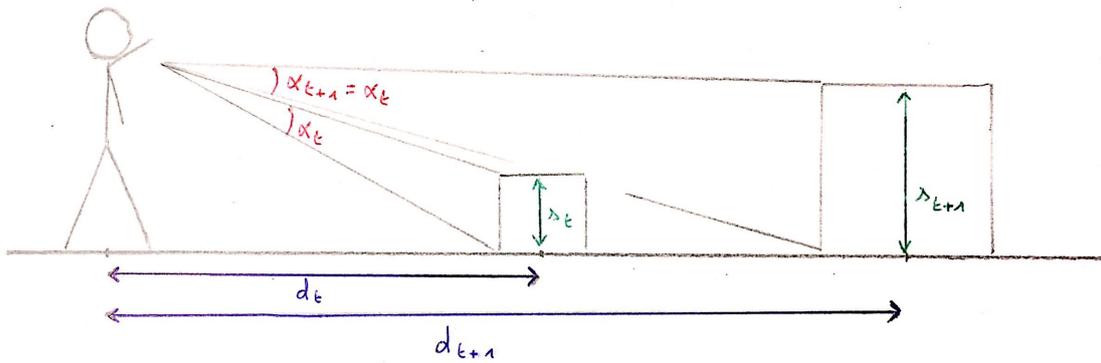


FIGURE 2.2 – Actualisation de la taille  $s(t+1)$  de l'objet sélectionné

Avec  $\left\{ \begin{array}{l} \text{Taille apparente : } \alpha_{t+1} = \alpha_t \\ \text{Taille réelle de l'objet à l'instant } i : s_i \\ \text{Distance par rapport à l'utilisateur : } d_i \end{array} \right.$

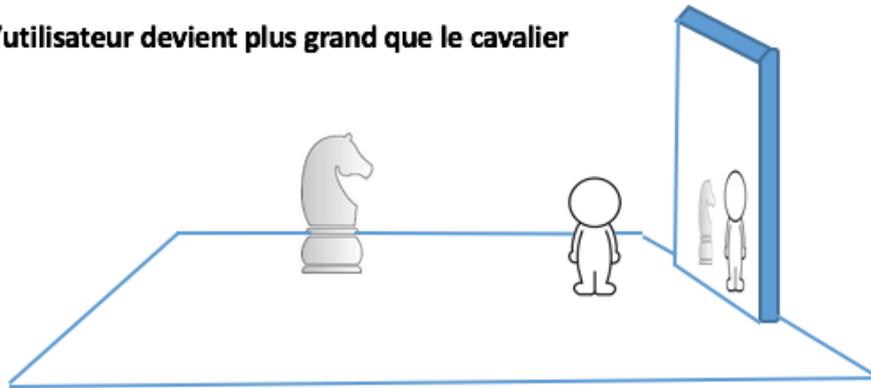
On en déduit facilement en appliquant le théorème de Thalès :

$$s_{t+1} = s_t * \frac{d_{t+1}}{d_t}$$

### Changement de la taille du personnage avec les miroirs

L'utilisateur peut également modifier sa propre taille en jouant avec la taille de son reflet dans un miroir vis à vis du reflet d'un autre objet dans le miroir. Sa taille est modifiée uniquement lorsqu'il sélectionne le miroir.

**Cas 1 : L'utilisateur devient plus grand que le cavalier**



**Cas 2 : L'utilisateur devient plus petit que le cavalier**

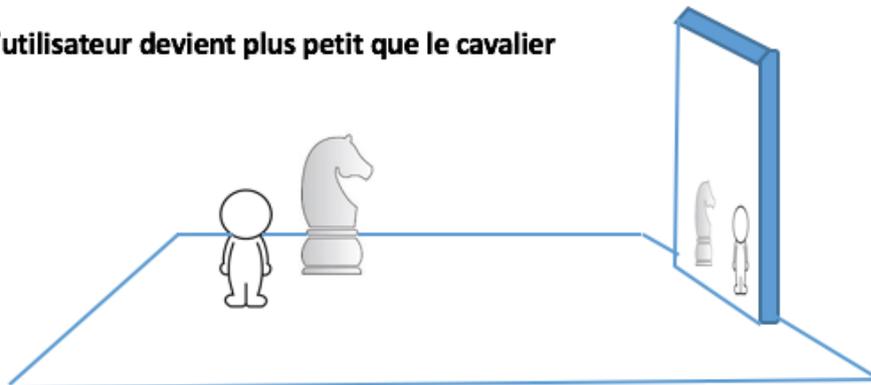


FIGURE 2.3 – Modification de la taille du personnage

## 2.4 Niveaux de réalisations

### 2.4.1 Réalisation minimale : ✓

La réalisation minimale pour ce projet correspondait au développement des interactions de base ainsi que de l'environnement. L'utilisateur devait pouvoir sélectionner les objets grâce aux contrôleurs du casque HTC Vive, les déplacer dans l'espace ainsi que modifier leurs tailles. L'application devait également permettre à l'utilisateur de déplacer son avatar ainsi que de modifier sa taille.

## 2.4.2 Réalisation intermédiaire : ✓

La réalisation intermédiaire correspondait à l'amélioration de l'environnement virtuelle ainsi que des graphiques et des objets à la disposition de l'utilisateur. L'utilisateur peut jouer sur sa propre taille en interagissant avec son reflet dans des miroirs. Les mouvements, les déplacements d'objets ainsi que les changements de taille se devaient d'être fluide et précis. L'utilisateur devait également avoir la possibilité de réaliser d'autres actions avec son avatar, comme sauter ou activer certains objets par exemple.

Nous avons eu à coeur, dès le début du développement, de proposer des graphismes de qualité et un environnement riche. Nous avons pour cela utilisé, entre autres, l'asset *Nature Starter Kit 2*. A l'origine, il était uniquement possible d'interagir avec des objets simples (cubiques ou sphériques). Nous avons travaillé pour que la mécanique de la taille apparente soit également possible avec des objets beaucoup plus complexes provenant de l'Asset Store (maisons, pyramides, animaux...).

## 2.4.3 Réalisation optimale : ✓

La réalisation optimale pour ce projet a été de parvenir à créer une application parfaitement intuitive et accessible même aux plus novices. L'objectif a donc été de parvenir à guider l'utilisateur en permanence pour que cette expérience immersive soit profitable et productive. Le tutoriel de jeu lui permet d'appréhender les commandes et la mécanique du jeu, et des aides visuelles et sonores lui indique le chemin à suivre tout au long de son parcours. A ce stade les illusions d'optique se devaient également d'être parfaitement réalisées.

La scène de tutoriel, centrée autour de l'évasion de la prison, correspond à ce qui était attendu de ce niveau de réalisation avec un guidage à la voix permettant à l'utilisateur d'être orienté et de mieux comprendre son environnement. C'est en effet un vrai challenge lorsqu'il utilise le jeu pour la première fois : la surcharge cognitive est alors importante, du fait de la nouveauté de l'environnement et il est donc nécessaire d'être parfaitement clair et de proposer une aventure interface intuitive.

## 2.4.4 Encore plus loin

Nous avons poussé un peu plus la réalisation avec des détails permettant d'enrichir l'expérience utilisateur :

- Lorsqu'un objet est sélectionné, au lieu de simplement changer la couleur du rayon de la baguette, nous avons créé un système de particules ressemblant

- à un laser. Un bruit sonore de laser est également rajouté.
- Les objets sélectionnés deviennent plus lumineux.
- Lors de leurs sélections, les objets entament une rotation douce pour faire face à l'utilisateur, puis sont tout le temps tournés vers l'utilisateur jusqu'à relâchement. Cela améliore l'effet visuel de la taille apparente.
- Des systèmes de particules autour des objets avec lesquels on peut interagir (le coffre, le portail...)

## 2.5 Quelques difficultés rencontrées

### Déterminer tous les cas possibles

Une des difficultés majeures lors de la réalisation de ce projet fût la projection des objets sélectionnés sur le premier plan rencontré par le raycast derrière notre objet.

La position (et donc la taille) de l'objet est actualisée à chaque frame. On positionne l'objet à la première intersection entre le raycast de la baguette et un objet différent de l'objet attaché (l'arrière plan). Hors, si on place notre objet à cette position, la moitié de l'objet attaché sera encastré dans l'arrière plan. Par exemple, lorsque l'on vise le sol, l'intersection est située en un point  $(x, 0, z)$ . Or, si l'on place un cube à ce point, sa moitié inférieure sera encastrée dans le sol. Il a donc fallu identifier un maximum de cas possibles où la position de l'objet doit être ramenée vers l'utilisateur ou vers le haut, ce qui s'est avéré difficile surtout lorsque l'objet est projeté sur des surfaces complexes.

Il fallut également identifier d'autres cas, comme par exemple lorsque la première intersection du raycast n'est pas l'objet sélectionné (quand l'utilisateur bouge trop vite la baguette et que l'objet attaché n'a pas le temps de suivre), ou bien lorsqu'il n'existe tout simplement pas de point d'intersection...

### Lâcher l'objet sélectionné sur un autre objet et éviter les forces de collision

Bien que nous essayons de prendre en compte tous les cas possibles, il arrive parfois que l'utilisateur lâche l'objet alors que celui-ci est encastré dans un autre (la surface de l'arrière plan n'est pas plane). En raison des rigidbodies et colliders présents sur les objets, l'énergie dégagée lorsque l'utilisateur lâche un objet encastré dans un autre projette les objets dans des directions différentes à très grande vitesse. Nous avons réglé ce problème en instanciant un timer dans un thread

chargé de réduire les vitesses des objets à 0 après quelques millisecondes (le temps de se désencastrier).

## **Personnage**

Nous avons choisi de créer notre propre first person controller. Comme l'utilisateur peut se voir dans les miroirs et doit pouvoir avoir une idée de sa taille, celui-ci est doté d'un chapeau ainsi que d'une paire de bottes. Nous avons également voulu doté le personnage d'un rigidbody pour que la gravité lui soit appliquée ainsi que d'un collider pour qu'il ne puisse pas passer à travers les objets. Un trigger est également situé au niveau des pieds du personnage pour que nous sachions quand celui touche le sol, et ainsi lui permettre de sauter à nouveau. Un dernier détail était de rajouter des bruits de pas si l'utilisateur se déplaçait au sol.

Avec tous ces nouveaux paramètres, un problème s'est créé. A cause du collider, le personnage rebondissait contre les autres colliders qu'il touchait et cela créait une sensation très désagréable pour l'utilisateur. Cela l'empêchait également de sauter sur des objets (il rebondissait dessus). La solution fût de réduire à zéro la vitesse du personnage lorsque son collider sortait d'une collision. Hélas, lorsque le personnage saute, son collider n'est plus en contact avec le collider qui était sous ses pieds avant de sauter, la vitesse était donc réduite à zéro et le personnage ne sautait que de quelques centimètres en l'air. Il fallut donc redoubler d'ingéniosité pour différencier tous ces cas, en vérifiant notamment que l'utilisateur a décollé car il a appuyé sur la touche de saut puis est retombé au sol : alors seulement la vitesse n'est pas réduite à nulle.

## **Miroir**

Le premier miroir que nous avons implémenté fonctionnait avec un shader que nous avons trouvé sur internet. Malheureusement, cette version ne fonctionnait pas en stéréo : en VR, le reflet était dédoublé dans le miroir. Nous avons finalement trouvé une solution avec un asset permettant de créer un miroir à l'aide d'une caméra.