Projet de Fin d'Etudes 2016-2017

Description of Work

CHESSPROJECT

Participants :

Tom DALL'AGNOL
Arnaud GARNIER
Lisa JOANNO
Simon PARIS

Encadrants :

Christophe PAPAZIAN

1. Résumé exécutif

En PNSInnov, nous avions commencé le ChessProject. Nous souhaitons approfondir ce projet car nous sommes tous les 4 motivés à l'idée de créer un jeu à présenter au Festival International du Jeu de Cannes en février 2017. Nous avions, en juin dernier, débuté un framework de création de jeux en réalité augmentée (RA). Pour cette deuxième étape, nous voulons converger vers un jeu innovant mettant à profit les possibilités que nous offre la RA.

Ce sera donc un jeu type jeu de plateau (echec, jeu de dame…). Nous souhaitons décupler sa jouabilité en proposant par exemple un plateau cubique. Le jeu de plateau étant le plus simple à appréhender cela nous permettra de proposer aussi un graphisme convainquant.

Dans ce rapport, nous présenterons une description détaillée du projet. Nous ferons par la suite un état de l'art, puis nous parlerons de l'organisation que nous prévoyons pour la mise en oeuvre de ce projet. Nous terminerons par une description des participants.

Table des Matières

I. Description du projet

Contexte

L'arrivée de la Réalité Virtuelle (RV) dans nos foyers a révolutionné l'univers du jeu vidéo : non plus passif, le joueur rentre physiquement dans son jeu pour interagir avec. Le problème se posant est que, dans le jeu, tout n'est que virtuel : les objets virtuels n'ont aucune consistance et les objets du réel ne sont pas pris en compte dans le jeu. Les deux mondes ne sont donc pas liés ce qui provoque nausée d'un côté (à cause de notre oreille interne) et frustration par un orteil blessé (par le meuble du salon que nous ne pouvions voir) de l'autre.

C'est là que se place la Réalité Augmentée (RA). Celle-ci permet d'ajouter au monde que nous percevons, et avec lequel nous interagissons au quotidien, des entités virtuelles. Cela peut aller de simples indicateurs (un popup "Vous avez 3 mails") à des éléments plus poussés que nous pouvons impacter (l'apparition d'une créature virtuelle que nous pouvons capturer dans notre jardin). Le succès de Pokémon GO de cet été (7,5 millions de téléchargements en moins d'une semaine de disponibilité aux Etats-Unis, source : Sensor Tower) a permi à la RA de faire une première avancée vers le grand public. De même, Microsoft propose dès à présent un kit de développement pour un casque dédié à la Réalité Augmenté (le Microsoft HoloLens) qui devrait sortir dans un futur proche. Le public comme l'industrie est intéressé par la RA mais pourtant beaucoup reste à faire.

En effet, l'une des principales critiques faites à Pokémon GO est le manque d'interaction entre les joueurs, un vrai multijoueur : pouvoir échanger ses créatures avec d'autres personnes, pouvoir défier des amis. Ce n'est pas étonnant, avec la RA nous rajoutons une couche virtuelle sur le réel, or les autres personnes font parti de cette réalité et nous souhaitons pouvoir interagir à plusieurs sur le même "virtuel", ou tout du moins pouvoir partager le virtuel sur lequel nous pouvons interagir avec d'autres personnes Par exemple, un échange de créature dans le cas de Pokémon GO, ou bien des parties de tir au pigeon virtuel que nous pourrions jouer qu'avec des personnes choisies (une sorte de lobby de jeu dans la réalité, où seules 8 amis choisis peuvent voir le fameux pigeon). Des problèmes de passage à l'échelle apparaissent alors : comment proposer à 8 personnes un jeu "privé" quand nous avons déjà 20 millions[1] d'autres personnes jouant déjà et que personne ne souhaite avoir des problèmes d'attente entre deux parties.

[1] Nombre d'utilisateur quotidien de Pokémon GO en Août 2016.
Source : http://www.businessofapps.com/pokemon-go-usage-revenue-statistics/

Motivations

Tout d'abord le fait que notre projet vise à développer un jeu est assez motivant mais cela ne s'arrête bien sûr pas là ; c'est le fait d'avoir et d'être encore témoin de la monté des jeux en réalité virtuelle, et d'anticiper la montée de la réalité augmentée qui est très motivant. C'est un monde nouveau où les perspectives de créations et d'innovations sont vastes. C'est un nouveau marché qui s'ouvre, une clientèle à atteindre. Le fait que le jeu en réalité augmentée ne soit pas encore grand public nous place dans une position d'explorateurs ! Si l'on considère que la concurrence est faible dans ce domaine, il faut aussi considérer que la clientèle est encore à convaincre. Notre jeu pourrait contribuer à la démocratisation de cette nouvelle façon de jouer.

Côté technique, nous nous servirons de smartphone pour avoir des lunettes de réalité augmentée ; notre jeu devra donc être suffisamment optimisé pour être déployé sur de tels engins. C'est surtout côté IHM que se trouve le challenge : les interactions en réalité augmentée font encore partie, pour beaucoup, de la science fiction. Il faudra proposer des contrôles adaptés et qui convainc nos utilisateurs tout en s'appuyant sur très peu d'expérience.

Objectifs

Concernant les objectifs que nous envisageons pour ce projet, nous en avons plusieurs. On peut tout d'abord citer les objectifs que nous nous étions fixés pour le projet de PNSInnov et qui n'ont pas pu être atteints. Pour nous ces objectifs sont les plus importants, et il nous est impératif de les avoir atteints pour la fin de ce projet. Ces objectifs comprennent l'ajout du mode multijoueur, qui est l'intérêt principal pour les joueurs, l'interaction en réalité augmentée (à l'aide du Leap motion par exemple), ainsi que l'adaptation de notre application au cardboard. Ces trois objectifs sont prioritaires pour nous car ils sont nécessaires à l'obtention d'un jeu présentable au festival des jeux de Cannes.
Cependant, ce ne seront les les seuls objectifs que nous souhaitons nous fixer. En effet, nous envisageons également d'ajouter de nouvelles fonctionnalités, afin de rendre notre jeu unique et innovant. Nous envisageons d'ajouter des comportements particuliers pour nos pièces, du type déplacement et déterminer les pièces à tuer. Mais l'idée ne s'arrête pas là : le joueur adverse ne serait pas au courant de la pièce à éliminer dans le camp adverse, ce qui fait que n'importe quelle pièce est suspecte. S'en suit un jeu d'esprits qui vise à bluffer pour induire son adversaire en erreur. Par la suite, nous comptons ajouter une nouvelle fonctionnalité de cases piégées. Ces cases, en apparence des cases normales, auraient des comportements prédéfinis, qui auraient pour but de corser un peu la partie, en ajoutant une dimension "aléatoire" dans le jeu. Nous avions pensés à quelques pièges types :

- Une case *craquelée*, qui se détériore dès qu'une pièce s'y déplace, jusqu'à ce que la case se casse et qu'on ne puisse plus s'y déplacer ;
- Une case *incendiée*, où les pièces ne peuvent pas se déplacer (on peut même imaginer que le feu se propage, rendant les cases adjacentes risquées…) ;

- Une case "*convertisseuse"*, qui convertit en une pièce de la couleur du camp adverse (si on convertit la pièce à tuer, il faudra que le joueur qui perd sa pièce reconfigure une pièce à tuer) ;
- Une case *téléporteuse*, qui déplace toute pièce la franchissant sur une autre case aléatoire non occupée ;
- Une case *passeuse de tour*, qui n'a pas d'effet sur le plateau mais permet à l'adversaire de jouer deux fois de suite.

Afin de mieux exploiter la dimension de réalité augmentée, nous avons imaginé l'ajout du relief sur notre plateau. Certaines cases seraient en fait perçues comme des obstacles, qu'il faudrait escalader. Cela inclut donc l'ajout d'une capacité d'escalade pour les pions.

Concernant les objectifs que nous percevons comme des défis, on pourra citer le changement de face des modèles, qui sera sans doutes la partie la plus difficile à implémenter, l'ajout du relief ainsi que les pièges.

Risques

Utilisant des plugins liés à la réalité augmentée, nous avons certains choix imposés, comme l'OS sur lequel nous pourront déployer notre jeu. Cela peut donc induire un problème matériel : il nous faut obtenir des smartphones suffisamment performant, avec le bon OS et trouver des lunettes adaptées. Toujours du côté matériel il nous faudra nous tourner vers un imprimante 3D pour fabriquer notre objet traqué (notre plateau sera projeté sur un objet prédéfini). Nous ne pouvons pas esquiver ce problème et il est important de rechercher le matériel et de se renseigner sur l'utilisation d'imprimante 3D le plus tôt possible. Cela permettra de faire les démarches nécessaires en cas de difficultés.

Nous souhaitons d'une part un résultat scientifique et en même proposer un jeu convaincant. Il nous faudra concilier les deux afin d'une part de pouvoir présenter notre travail au festival du jeu et recueillir un feedback primordial, et livrer un projet digne de dernière année de formation d'ingénieur. Il faut donc ne pas perdre de vu l'un des objectif au détriment de l'autre, cela pourrait mettre notre projet en échec. Pour garder le cap il nous faut demander le plus souvent que possible des avis extérieurs, notre encadrant en premier lieu, les responsables de spécialité AL et IHM afin de vérifier tout au long de notre travail que nous continuons à livrer du travail suffisamment technique.

Scenarios

Nous avons 3 scénarios principaux pour notre projet.

Scénario #1 : Multijoueur

Donald et Hillary sont des amis de longue date. Les deux sont très occupés par leur travail mais se souviennent de leur longues parties d'échec lorsqu'ils étaient à l'école. Très compétiteurs, ils aiment se retrouver autour d'une partie de jeu de plateau. Ils possèdent tous les deux une connexion internet et l'application installée (sur téléphone ou avec des lunettes de RA).
Ils lancent tous les deux l'application et s'équipent de leurs lunettes de réalité augmentée. Donald crée un salon et Hillary le rejoint. Une fois la connexion établie, ils peuvent commencer leur partie et ils voient les mouvements de l'autre en temps réel.

Critères d'acceptation :

- Un joueur a la possibilité de créer un salon et attendre un autre joueur
- Un joueur peut rejoindre un salon déjà créée
- Deux joueurs ont rejoint le salon : la partie commence
- Lorsqu'un joueur effectue un mouvement, l'autre joueur le voit et voit les effets du mouvement de l'autre joueur
- L'état du plateau de jeu est constamment le même chez les deux joueurs du salon

Scénario #2 : Jouer en réalité augmentée

Clémence possède des lunettes de réalité augmentée et aime les jeux de plateau. Elle lance l'application et place le cube de jeu en face d'elle. Une fois en jeu, elle voit le terrain de jeu là où se trouve le cube placé devant elle. Les pièces sont placées sur le terrain de jeu.
Elle sélectionne avec sa main une pièce, puis la déplace. Elle voit cette pièce changer de position sur le terrain de jeu. Elle souhaite visualiser le reste du terrain de jeu, donc elle tourne le cube physique, et elle voit le plateau de jeu et les pièces tourner avec lui.

Critères d'acceptation :

- Un joueur, muni de lunettes de réalité augmentée, lance la partie avec le cube devant lui. Il voit le plateau virtuel avec les pièces en relief.
- Le joueur bouge le plateau virtuel en bougeant le cube en face de lui.
- Le joueur voit les pièces bouger sur le plateau virtuel.

Scénario #3 : Choisir les pièces

Olivier a beaucoup joué à des jeux de plateau et il aime les jeux de stratégie. Il a des idées de mouvements pour ses pièces.

Il lance le jeu. Il a rejoint ou créé un salon et la partie commence. Il a du temps pour personnaliser les mouvements de ses pièces comme il en a envie. Pendant ce temps, son adversaire personnalise également ses pièces. Lorsque la partie commence, Olivier ne connaît pas les mouvements des pièces de son adversaire.

Critères d'acceptation :

- Lorsque le joueur lance la partie, une interface de personnalisation des pièces apparaît
- Le joueur peut, une à une, personnaliser les mouvements et coups possibles de ses pièces
- Le joueur peut signaler qu'il a terminé de personnaliser ses pièces
- Lorsque les deux joueurs l'ont signalé, la partie commence
- Les mouvements des pièces du joueur correspondent à ce qu'il a prédéfinis, et il ne connaît pas les mouvements de son adversaire.

II. Etat de l'art

Pour créer un jeu mobile employant des objets virtuels 3D (comme nous le souhaitons), plusieurs choix s'offrent à nous :

- Unreal Engine 4
- Unity3D

Les deux premiers sont des solutions assez similaires dans le fait qu'ils sont tous deux des moteurs de jeu mais aussi des outils complets (c'est à dire qu'ils proposent un outillage complet pour faciliter la création de jeu vidéo). Ainsi, sous la forme de "Scènes", un Concepteur de Jeu peut mettre en place les niveaux de jeux, paramétrer les différents éléments, mais aussi changer les modèles 3D des objets sans avoir à toucher au code. La logique de jeu proviendra par contre des développeurs. Ils devront coder les interactions et les comportements des différents objets. Un autre avantage est la présence d'un "marché" intégré, permettant à d'autres créateurs de proposer (gratuitement ou non) des modèles 3D ou des comportements pré faits. Tous deux permettent aussi de déployer notre jeu sur différents supports : android, IOS, Windows, OSX, Linux.
Le problème de ces deux outils étant que tous les deux produisent en sortie des jeux assez lourds car ils embarquent tout le moteur physique et tous les éléments "facilitateurs" directement dans l'exécutable final.

Au niveau des différences, Unreal Engine 4 possède une moins bonne communauté que Unity3D. Dans ce dernier, la documentation est à jour, précise et propose des exemples appréciables. Un simple tour sur les forums de Unity permettent d'avoir des informations lorsque l'on est bloqué lors de l'implémentation d'un comportement.
Par contre, nous n'avons pas accès à la base de code de Unity3D, certains comportements restent donc "cachés" voir "aléatoires", contrairement à Unreal Engine qui est ouvert. Autre avantage de l'Unreal Engine est de proposer des rendus très réalistes et donc plus beau que Unity.

Ainsi, nous avons opté pour Unity3D pour la communauté solide, qui est toujours la bienvenue lors de problème avec une technologie. Proposer de meilleurs graphismes (ou tout du moins plus réaliste) n'est pas à prendre en considération quand on sait que notre cible est un smartphone, support qui n'aurait pas les capacités de faire tourner un jeu avec des graphismes trop réalistes.

Côté Réalité Augmentée, on voit qu'il existe un certains nombre de cadriciels permettant de faciliter son utilisation. Parmi les plus utilisés, nous retrouvons Vuforia et ARToolkit, tous deux proposant de s'interfacer avec Unity3D. Ces technologies ont été choisies principalement par leur utilisation de "marqueurs", c'est à dire d'objets réels qui servent au logiciel de placer les objets virtuels dans le réel. Comme nous souhaitons proposer un jeu de plateau en Réalité

Augmentée, permettre d'avoir un plateau réel sur lequel ajouter des pièces et des animations en 3D est exactement ce que l'on souhaite.

L'avantage de Vuforia est de toujours faire apparaître l'objet virtuel alors même que le marqueur est en grande partie caché : nous avons un meilleur tracking ce qui permet une utilisation poussée de l'objet qu'on souhaite suivre. Vuforia propose aussi un support direct des nouveaux équipements pour la réalité augmentée (que ce soit les Cardboard comme nous souhaitons utiliser, ou bien les traditionnelles webcam). Le problème viens du côté boîte noir de la technologie : aucun moyen de savoir ce que fait vuforia exactement, ce qui fait qu'au moindre problème il n'y aura pas d'aide directe. Un autre désavantage est le fait que cette technologie n'est disponible que sur IOS, Android, Windows 8 et 10 et OSX. Cela force un environnement de travail.

L'avantage de ARToolkit est sa communauté bien plus importante que Vuforia : s'il y a un problème nous pouvons facilement le résoudre à l'aide de la communauté. Un autre point positif à ce niveau là est que le code est open source, on peut donc y accéder et vérifier des comportements pour s'assurer que c'est bien ce qu'on attend. De plus, il permet de déployer sur des lunettes de réalité augmenté ce qui est un grand plus. Le gros point négatif est le tracking de marqueur, ARToolKit perdant facilement l'objet que nous souhaitons suivre.

Nous avons choisi d'employer Vuforia, car notre plateau sera manipulé régulièrement et fait parti de l'expérience utilisateur que nous souhaitons proposer : nous ne pouvons pas laisser disparaître trop régulièrement les objets virtuels car nous cachons le marqueur nous servant de plateau de la main. L'expérience utilisateur étant un des points nous tenant à coeur, nous nous sommes donc dirigé vers cette technologie, et il nous reste entre autre à voir comment nous pouvons interagir avec.

Actuellement, nous pouvons voir plusieurs techniques traditionnellement mises en oeuvre pour interagir avec les objets virtuels de la RA.

Comme dans PokémonGO, l'écran tactile du téléphone peut permettre d'interagir avec. L'avantage est la mise en place aisée car il s'agit ni plus ni moins que la technologie de base du tactile comme pour n'importe quel jeu "normal". Le problème est que, à part voir des éléments virtuels s'ajouter au réel, nous nous retrouvons avec des interactions proche de ce que l'on peut déjà faire avec des jeux vidéos "normaux".

Un moyen plus en lien avec la réalité serait d'utiliser directement nos mains pour interagir avec les objets virtuels, et non plus par l'intermédiaire d'un écran. A l'aide d'une caméra à détection de mouvement telle que la Kinect de Microsoft, nous pouvons les retranscrire dans le monde virtuel et ainsi saisir et même jouer avec les objets virtuels. Dans notre cas de jeu de plateau, seule les mains étant importantes nous pouvons aussi employer un Leap Motion. Ce dispositif permet de capter seulement les mains et les avant-bras d'un utilisateur pour les retranscrire précisément dans le virtuel. Le problème de leur utilisation est qu'il force l'emploi de matériel en plus du plateau et du smartphone. Le leap motion a tout de même l'avantage d'être

portable et de pouvoir fonctionner directement connecté avec un smartphone, alors que le kinect nécessite un ordinateur.

Un dernier moyen est l'utilisation d'un autre objet que l'on trackerai grâce à la technologie de réalité augmentée. Servant de "baguette magique", nous aurions un objet physique qui permet d'interagir avec le virtuel, mais sans que l'on puisse toucher nous même le virtuel. Le problème est qu'en cas de problème de tracking de l'objet utilisé, nous ne pouvons plus agir sur le virtuel. De plus, on perd le côté manuel du jeu de plateau (bouger les pièces à la main par exemple), ce qui est une sensation que nous souhaitons conserver.

C'est pour cette raison que nous avons choisi l'utilisation du LeapMotion : pour garder le plaisir du jeu de plateau comme il existe normalement, nous souhaitons une interaction directe avec les mains.

Un autre des objectifs est de proposer un jeu à deux joueurs, bien que les deux soient extrêmement éloignés. Comme nous souhaitons proposer un service rapide et fluide (nous ne voulons pas attendre 10 minutes que l'action de notre adversaire soit finalement reçue), il nous faut pouvoir monter en charge et ainsi pouvoir se maintenir même avec un grand nombre de joueurs en même temps. Ce serveur permet aussi de vérifier qu'un des joueurs ne triche pas, et ne fait pas de coups non autorisés.

Une solution retenue est celle de Photon. Il s'agit d'une technologie qui peut s'interfacer avec Unity3D. Le serveur peut être hébergé sur le Cloud, et un grand nombre d'optimisations automatiques sont faites pour permettre une montée en charge conséquente. Le désavantage principal est qu'il s'agit d'une technologie centrée sur le synchrone, sur le jeu en temps réel. Or nous souhaitons faire un jeu au tour par tour, donc asynchrone. Une grosse partie des fonctionnalités et des optimisations ne nous concernent pas voir nous gênent (la transmission de donnée serait régulière et fait donc chuter la batterie de nos smartphones avec l'utilisation).

Une seconde solution est celle de Go, langage conçu par Google. Un de ses intérêts est un déploiement assuré sur le support de notre choix : le logiciel produit embarque avec lui tout ce dont il a besoin pour tourner. On peut donc déployer sur un serveur sans avoir un installer un grands nombre de dépendances et/ou de prérequis : la simple ligne "go run monProgramme" suffit au lancement. De plus, le concept de goroutines permettent de passer à l'échelle : il a été fait pour paralléliser des programmes (exécuter en parallèle des programmes pour éviter un trop grand temps d'attente alors qu'un grand nombre d'utilisateur essayent d'accéder à notre serveur). Le problème va être de devoir établir un protocole de communication avec notre serveur, ainsi que des messages datés (il ne faudrait pas qu'un des joueurs reçoivent des actions qui ont lieu après d'autres qu'il n'a pas encore reçu) mais aussi établir le modèle de jeu à la fois côté serveur dans un langage, et côté client dans un autre.

Nous avons donc choisi d'utiliser le langage Go et faire notre serveur nous même, pour permettre une montée en charge certaine pour notre cas, celui d'un jeu au tour par tour.

Une autre problématique liée au déploiement sur smartphone est la consommation de donnée. Dans le cas où nous souhaitons jouer en extérieur, il faut penser à réduire les échanges au maximum (pour les biens de la batterie mais aussi du forfait data du joueur). Il est donc inimaginable d'envoyer un état de jeu complet au serveur et que celui ci le transmettre aux joueurs. Dans l'industrie du jeu, ce qui est souvent fait est d'envoyer l'action effectuée (par exemple aux échecs, transmettre seulement "cavalier part de C7 pour arriver en D5"). Ainsi, nous n'aurons qu'à exécuter l'action transmise pour se retrouver dans le même état de jeu que l'adversaire. Nous pouvons aussi sauvegarder côté serveur les différents coups joués pour les renvoyer au joueur s'il a perdu la connexion. On parle ainsi de Programmation Basée sur des Évènements. Bien qu'il faille exécuter du coup trois fois le coup (une fois pour le joueur qui fait le coup, une fois pour le serveur pour vérifier la validité du coup et une fois par le joueur subissant le coup pour mettre à jour son côté), nous y gagnons en terme de nombre de données échangées.

III. Mise en oeuvre

Lots envisagés

- **Base du jeu (3 h/s)**
  Avoir une base de jeu sous Unity. Il faut avoir un système pour se déplacer de case en case, et un plateau de forme cubique d'où on peut se déplacer sur n'importe quelle case. La base sera faite en C#, langage que nous maîtrisons, aucun apprentissage n'est prévu à ce niveau là.

- **Multijoueur (2 h/s) : dépend de Base du jeu**
  Pouvoir jouer à notre jeu à deux avec deux dispositifs de RA. Il nous faudra apprendre le Go de manière à faire un serveur pouvant monter en charge, mais aussi récupérer et transmettre des évènements qui seront exécutés par les clients.

- **Cardboard (1 h/s) : dépend de Base du jeu**
  La démo de notre jeu était faite grâce à une webcam. Pour qu'il soit jouable il faut que nous puissions jouer avec des masques de réalité augmentée (cardboard, ou dispositifs de meilleure qualité...).

- **Interaction en RA (4 h/s)**
  Là encore nos mouvements lors des démos étaient faits avec la souris. Nous n'aurons que les cardboards pour le produit final, il faut donc implémenter des interactions adaptées pour pouvoir jouer avec les objets 3D. Il va falloir essayer d'intégrer le LeapMotion avec Unity pour pouvoir intéragir avec des objets virtuels. Il est donc nécessaire d'apprendre à l'utiliser.

- **Comportement (2 h/s) : dépend de Base du jeu**
  L'idée est d'avoir un nombre de pièce prédéfini avec pour toutes, un comportement de base (déplacement d'une case dans les 4 directions). On aurait ensuite d'autres comportements plus spécifiques, plus intéressants et en quantité limitée. Ces comportements seraient attribués comme on l'entend à nos pièces mais notre adversaire ne serait pas au courant de nos choix : ainsi s'instaurent des tactiques de bluff pour éviter que l'adversaire ne devine le comportement de chacunes de nos pièces.
- **Changement de face (3 h/s) : dépend de base de jeu**
  Notre plateau étant un cube, nous voulons ajouter la possibilité d'échanger les différentes faces. Il faut donc pouvoir sélectionner deux faces et pouvoir les échanger, que ce soit visuellement mais aussi au niveau du modèle de donnée.

- **Pièges (2 h/s) : dépend de changement de face**
  Notre idée est de pouvoir donner la possibilité au joueur de placer des *pièges* sur le plateau. Voici des idées de pièges :
    - Une case *craquelée*, qui se détériore dès qu'une pièce s'y déplace, jusqu'à ce que la case se casse et qu'on ne puisse plus s'y déplacer.
    - Une case *incendiée*, où les pièces ne peuvent pas se déplacer (on peut même imaginer que le feu se propage, rendant les cases adjacentes risquées...).
    - Une case *convertisseuse*, qui convertit en une pièce de la couleur du camp adverse (si on convertit la pièce à tuer, il faudra que le joueur qui perd sa pièce reconfigure une pièce à tuer).
    - Une case *téléporteuse*, qui déplace toute pièce la franchissant sur une autre case aléatoire non occupée.
    - Une case *passeuse de tour*, qui n'a pas d'effet sur le plateau mais permet à l'adversaire de jouer deux fois de suite.
    - ... plus d'idées à venir ...

  Ces pièges vont de pair avec une idée de changement de faces. En effet, si un joueur voit qu'une face du plateau est remplie de pièges, il aura la possibilité d'échanger deux faces, faisant, s'il se débrouille bien, tomber plusieurs pièces adverses dans des pièges (et peut-être même des siennes, dommages collatéraux).

- **Cases en relief (1 h/s)**
  Notre idée est d'introduire des cases en relief. Ces cases sont donc des obstacles. On associe aux pièces un niveau d'escalade, qui lui permet d'escalader les cases d'altitude inférieure ou égale à son niveau d'escalade.

Livrable :

- Multijoueur
- Cardboard + interaction en RA
- Pièges et faces
- Comportement
- Case en relief

Planning

Une estimation du plan que nous souhaiterions mettre en place :

# Semaine	Description
44	Développement base du jeu, réflexion sur comment interagir en réalité augmentée
45	Développement base du jeu, réflexion sur comment interagir en réalité augmentée
46	Développement base du jeu (à la fin de cette semaine, on devrait avoir un jeu 1 joueur minimal)
47	Intégration mode multijoueur, adaptation du code pour cardboard
48	Intégration mode multijoueur, adaptation du code pour cardboard
49	Multijoueur, interaction avec la réalité augmentée, ajout changement de faces
50	Interaction avec la réalité augmentée, ajout changement de faces
51	NOEL
52	NOEL
1	Interaction avec la réalité augmentée, ajout changement de faces
2	Ajout des cases pièges
3	Ajout des cases pièges, du relief
4	Ajout du relief et pièges
5	Ajout du relief et pièges
6	Ajout du relief
7	Préparation d'un jeu jouable (revue de l'interface, tests d'utilisation), préparation pour le FIJ
8	Préparation du jeu pour le FIJ le week end, récupération des feedback utilisateurs
9	Oraux
Légende	
	: Semaine de vacances
	: Semaine à temps plein sur le projet

Afin de maintenir ce planning à jour, à la fin de chaque semaine nous ferons un bilan des objectifs réalisés et non réalisés pendant la semaine. Nous analyserons les causes qui ont fait que nous n'avons pas pu réaliser ces objectifs, et proposerons une solution pour y remédier. L'utilisation d'outils de développement agile, tels que Jira, permettra de maintenir une bonne dynamique tout au long du développement du projet.

Outillage

Unity3D

C'est un moteur de jeu qui permet de créer des jeux vidéos en 3D. Il nous permet de simplifier le déploiement de jeux sur des supports différents, dans notre cas sur Android. Nous l'avons déjà utilisé lors du PSInnov et il est largement utilisé par la communauté des développeurs de jeux vidéo.

Aisance : toute l'équipe l'a utilisé lors du projet PNSInnov (3 semaines à temps pleins). Lisa l'a aussi utilisé lors de son stage et Tom pour un projet dans un cours à Montréal et pendant des projets personnels.

Gestionnaire de version

Nous voulons utiliser un gestionnaire de version interne à Unity afin de faciliter la résolution des conflits et le merge des fichiers. Nous allons utiliser Plastic SCM car il est recommandé par Unity3D et performant dans le développement de jeux vidéo.

Aisance : Nous avons tous déjà utilisé des gestionnaires de version, mais Plastic SCM nous est inconnu.

Go

Go est un langage qui devrait nous permettre de de créer facilement un serveur, indispensable à notre jeu. Là encore il nous offre une facilité de déploiement sur divers OS et une facilité de passage à l'échelle qui ne manquera pas nous aider.

Aisance : Simon et Tom l'ont utilisé lors d'un cours de Programmation Concurrente. Ils ont ainsi des notions dans ce langage mais un apprentissage supplémentaire sera nécessaire pour produire un serveur. Lisa et Arnaud ne l'ont jamais utilisé.

Vuforia RA

Vuforia est une plateforme de réalité augmentée qui met à disposition un SDK. Ce SDK est disponible comme plugin de Unity. Il nous permet ainsi d'outre passer les difficultés de la réalité augmentée, et d'espérer ainsi livrer un jeu convaincant. Cependant, il est à noter qu'il ne fonctionne pas sur Windows 7, et que cela rend le développement plus difficile, car 3 personnes sur 4 utilisent Windows 7 dans l'équipe.

Aisance : Simon s'en est occupé lors du PNSInnov, et Arnaud lors d'un cours d'Interaction. Une période d'apprentissage sera ainsi nécessaire.

LeapMotion

LeapMotion est un petit boitier permettant de détecter les mains et de les représenter pour l'ordinateur. La technologie est suffisamment précise pour détecter individuellement chaque doigt et comment ils se plient.

Aisance : Personne de l'équipe ne l'a encore utilisé, il va falloir donc apprendre à l'utiliser.

IV. Participants

Simon : Spécialité AL. J'ai travaillé sur la réalité augmentée par le biais du plugin Vuforia. J'ai aussi travaillé au développement sur Unity.

Arnaud : Spécialité IHM. Pendant le projet PNSInnov, j'ai utilisé l'outil Unity3D.

Lisa : Spécialisée AL. J'ai manipulé Unity3D pendant le PNSInnov et pendant mon stage de 4ème année, où j'ai rendu multijoueur un jeu préexistant en singleplayer.

Tom : Spécialité AL. J'ai pratiqué Unity3D pendant le PNSInnov mais aussi lors d'un cours de Conception de Jeux Vidéo à Montréal et sur des projets personnels. J'ai aussi utilisé Go dans le cadre d'un cours de Programmation Concurrente.

Graphiste (**Julien** Paris) : A déjà travaillé avec nous pendant le PNSInnov et nous a livré des visuels. Il a accepté de nous aider de nouveau pour ce projet.

Christophe PAPAZIAN est notre encadrant.

Nous pensons proposer notre jeu aux étudiants de la **promotion SI** dès que nous aurons les premières versions jouables de notre projet, dans le but de réaliser des essais.