haute école
neuchâtel berne jura
ingénierie
www.he-arc.ch

Rapport de Projet P3

Arc3D

Auteur(s):	Droxler Arnaud
Encadrant(s):	Julien Marchand
Mandant:	Stephane Gobron
Date:	27 janvier 2017
Version Doc.:	V1.0

Hes·so
Haute Ecole Spécialisée
de Suisse occidentale
Fachhochschule Westschweiz
University of Applied Sciences
Western Switzerland

Résumé

Ce rapport a pour but de présenter le projet d'étudiants en informatique de 3e année de l'He-arc ingénierie. Ce projet se déroule durant le premier semestre de l'année académique. Ce rapport synthétise les objectifs, travail à effectuer, les problèmes rencontrés ainsi que les solutions trouvées et les améliorations possibles.

Table des matières

Chapitre 1 Introduction

Le Campus ARC 2 de la He-ARC de Neuchâtel est un bâtiment de grande taille dont certaines zones sont restreintes au public. Les open-spaces du 2e étage ont un accès réglementé et limité, ils engendrent des problèmes tant pour les visiteurs que pour les étudiants. Le but du projet et de réaliser une application permettant de visualiser le bâtiment de la HE-Arc et ses alentours pour permettre au visiteur/étudiant de se repérer. Elle propose aussi une fonction de guidage pour aller d'un point A à un point B,et une fonction de géolocalisation a l'intérieur du bâtiment.

Chapitre 2 Contexte et besoin

Le projet a été en premier réaliser pas Thomas Roulin en tant que travail de bachelor. Il avait réalisé l'application en WebGL avec Tree.js, elle fonctionnait dans le navigateur web. Une recherche avait été faite, pour déterminer qu'elle fût, le meilleur moyen de localiser la personne dans le bâtiment. La trilatération wifi restait la meilleure alternative, car l'infrastructure est déjà existante. Mais un problème se posé, car le navigateur n'a pas un accès suffisant aux donnes nécessaires. Pour résoudre ce problème, il faut écrire une application native a la plateforme pour pouvoir récupère les informations des accès points wifi.

Un autre aspect du projet qui méritait d'être amélioré est le contenu de la scène. Il faut ajouter du contexte pour aider l'utilisateur à ce repère dans le bâtiment et ses alentours.Ce n n'avait pas été fait, car cela rajoutait beaucoup d'objets à rendre et cela avait un impact sur le framerate de l'application. C'est à cause de toutes ces contraintes que le choix de porter l'application existant sur Unity a été fait [**?**]. Unity est un moteur de jeux multiplate-forme très répandu dans l'industrie du jeu vidéo. Il permet de créer rapidement des applications 3d qui sont déployables sur de nombreuses plateformes. Les plateformes qui nous intéressent ici sont les plateformes mobiles (Android et IOS). En effet, une application de guidage comme celle-ci a plus de sens de se trouver sur mobile que sur desktop.

Chapitre 3 Principe de l'application

Le But du projet est de porter l'application de Thomas Roulin sur le moteur Unity, toutes les fonctionnalités de l'application doivent être implémentées dans ce portage. L'application doit permettre de visualiser le bâtiment de l' He-arc en vue a la première personne, avoir un framerate constant entre 60 fps. L'utilisateur pourra rentrer sa position et sa destination, l'application calculera le chemin le plus court et l'affichera à l'utilisateur. La caméra se déplacera dans le bâtiment en suivant le chemin calculé. UN mode pour les personnes Handicape est mis à disposition pour celle-ci. Le trajet passera par les l'ascenseur au lieu des escaliers. En plus de la fonction de guidage, l'application permet de trouver les toilettes les plus proches de notre position (pour les femmes et les hommes). L'autre fonctionnalité de l'application est de localiser l'utilisateur dans le bâtiment. Cela sera réaliser avec l'aide des access points wifi qui sont dispersés au sein du bâtiment. L'application doit aussi implémenter des objectifs plus secondaires, comme ajouter du mobilier au sein de l'école pour rendre le bâtiment plus vivant. Améliorer les textures du bâtiment et l'éclairage de la scène qui est inexistant dans le projet existant. Le bâtiment est vide et les alentours aussi, c'est pourquoi il faudrait rajouter des éléments distinctifs comme des arbres ou certaines textures sur les bâtiments. Un autre point important est la gestion des lieux, si on décide d'ouvrir ou de ferme une classe dans le bâtiment il serait intéressant de pouvoir modifier la liste des lieux visitable facilement. Un fichier externe devra contenir toutes ces informations et être facilement éditable. Enfin l'application devra pouvoir être déployée sur plusieurs plateformes comme Android, IOS et WebGL.

Chapitre 4 Choix techniques

Le choix a été fait de développer sur le moteur Unuty, il permet de créer notre scène et nos assets de manière graphique. Il permet d'aussi de définir des scripts d'actions, qui seront écrire en C#. Après avoir passé plusieurs semaines à prendre en main du moteur Unity et avoir pris de conscience des possibilités qu'il offre. Le pathfinding est une composante qui est très souvent utilisée dans les jeux vidéo, il existe un module qui permet de gère le pathfinding dans Unity. Après quelque test il a été décider d'utiliser ce module plutôt que de porter l'algorithme existant. Un autre problème à adapter l'algorithme existant est que les lieux étaient rentrés sous forme de graphe, mais les systèmes de coordonnées n'étaient pas le même que dans la scène Unity. Il aurait été difficile de transforme les coordonne des points du repère existant au repaire actuel. Concernant la trilatération il nous faillait plusieurs informations sur les access points de l'école. Il faillait connaitre notamment la postions des access points dans le bâtiment, un rendez-vous a été pris avec le service info pour obtenir ces informations et savoir s' il coopérerait sur le projet. Le rendez-vous c'est avéré favorable, car le service info nous autorisait a utilisé le réseau wifi de l école, mais nous avons aussi reçu un plan de l'école avec la position des access point dans le bâtiment.l'application a aussi besoin récupérer le SSID (Nom du Reseau), le BSSID (Adresse qui identifier les access points dans le réseau), RSSI (la puissance du signal de l'access point) pour réaliser la trilatération. Ces informations ne sont pas disponibles nativement avec les classes de base de Unity. Unity propose un asset store, il s'agit d'un magasin ou la communauté publie des assets, des scripts, des plug-ins. Après une quelque recherche, une solution a été trouvée. Android Info Scanner est un plugin qui permet récupère des informations du device Android. Les informations qui nous intéressent sont celles sur le wifi (SSID,BSSID,RSSI), le plugin fournit aussi des informations sur la batterie et le GPS. Aucun plugin semblable n'a été trouvé pour IOS. IL a été décidé de laisse le portage sur IOS de cote pour le moment.

Chapitre 5 Partie techniques

Cette partie détaille les différentes parties du projet

5.1 Création de la scène

Après avoir passé plusieurs semaines à découvrir l'environnement Unity et réaliser des tutoriels disponibles sur leur site. J'ai commence par importer le modèle 3d de l'école et des bâtiments au toure de la gare. Le modèle existant comprend le bâtiment campus Arc 2, le bâtiment de l'OFS, la gare et les bâtiments au tout de celle-ci, ce modèle a été réalisé avec le logiciel 3dsMax. Unity a un système d'Unity qui représente 1 mètre pour une unité, lors de l'importation du modèle dans Unity celui-ci était beaucoup trop grand par rapport a cette échelle. Il a donc fallut trouve le coefficient pour scaler les modelés pour qu'il corresponde a l'échelle d' Unity.

Une fois que le modèle fut importé, j'ai pus commence a travail sur les textures de celui-ci. Unity permet de créer des matériaux qui sont applicables sur des modèles vierges. Ces matériaux ont de nombreux paramètres, toutes les textures d'infographie sont présentes (albedo,spacular,normal,heightmap,occlusion). Les textures du bâtiment avaient été réalisées lors du projet précédent, j'ai donc pus les récurer pour créer les matériaux.

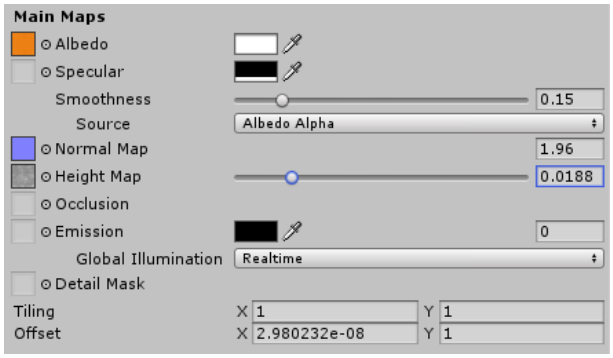

Figure 5.1: Liste des paramètres d'un matériau

La sky box est aussi considérer comme un matériau par Unity, pour les textures de celle-ci j'ai utilisé des assets gratuits disponibles dans Unity. Après l'avoir ajouté, j'ai ajouté une source de lumière directionnelle orientée comme le soleil présent sur skybox. Les ombres générer par le source sont calculé en temps réel dans la scène, elles sont donc réduites a la plus basse qualité pour éviter de perdre en performance, mais cela peut être améliore.(voir le chapitre concernant).

Malgré cela a scène reste triste et vide, car seul le bâtiment d' He-arc possédé des textures, les autres reste blanc pour le moment. Or sur la place de la gare et au tour de l'école de nombreux arbres sont présents. Il était donc important de les rajouter, car

permettre d'améliorer le réaliser de la scène et d'aider l'utilisateur a ce repérer. Unity propose un module pour créer des arbres de manière simplifier. On peut commence par diffinire la structure simplifier l' arbre (un tronc et des feuilles ou un trou plusieurs grosses branches puis des feuilles). Ensuite de nombreux paramètres permettent de créer l'arbre que l'on souhaite(taille, épaisseur, position des feuille et orientation des feuilles sur le tronc, etc.).Ces paramètres sont soumis a une seed random, cela permet de créer des arbres différant, mais avec les mêmes paramètres. Une fois les paramètres de l'arbre trouver, il reste plus qu'a plus placé dans la scène. Pour les placer les arbres j'ai créé un script qui permet de définir le nombre d'asset que l'on souhaite placer entre 2 points de la scène. Les arbres ont été placés comme ils sont disposés sur la place de la gare. Malgré le grand nombre d'arbres présents dans la scène, les performances sont restées constantes.

Figure 5.2: Point de vue de la scène

5.2 User Interface

L'étape suivante dans le projet est la création de l'UI, Unity permet de créer des UI simples et fournit touts les widgets fréquemment utiliser. Après avoir réfléchie sur la disposition des boutons et des fonctionnalités, j'ai choisi de m'inspirer de google map pour la disposition et la relation entre les différentes pages. La vue principale de l'application permet de voir la scène 3d et de lance les 2 fonctions principales de l'application, le guidage et la localisation. C'est aussi la vue que l'utilisateur aura lors que le guidage est effectué. L'autre vue de l'application est la vue de navigation elle permet de définir le point de départ, la destination, si on veut activer le mode handicape. Elle permet aussi de lance la recherche des toilettes la plus proche. La dernière vue est seulement afficher lors que l on veut ce localiser, mais que le wifi et le GPS ne sont pas actifs pour signifier a l'utilisateur qu'il doit les activer pour être localisé. Enfin on peut tourner le téléphone pour passer en mode portrait l'application gère ce changement est adapte les vues en conséquence.

Un widget custom a été créé pour cette interface, il s'agit de champ de texte pour rentre le départ et la destination. Il va proposer des noms de salle contenant les caractères précédemment tape dans le champ.

Figure 5.3: Interface principale de l'application

Figure 5.4: Interface permettant de définir la navigation

5.3 Navigation

Avant d'implémenter le pathfinding et la fonction de navigation, il faillait trouver un moyen de stocker le nom des lieux et leur position dans leur scène. Le format devait être facilement lisible par un humain pour qu'il puisse le compléter à la main et qu'il soit facile à parser pour l'ordinateur. Le choix a été fait d'écrire les données en JSON. JSON possède une structure simple qui est compressible pour tout le monde et une bibliothèque existait pour parser du JSON en C#.

La structure du fichier qui contient les informations sur les lieux est simple, les lieux sont stockés dans une table, chaque lieu est un objet qui contient un nom, une position[x,y,z] et une catégorie. La catégorie permet des differencier les toilés des salles de cours normales.

```
"place":[
    {
        "name":"043",
        "position": {
          "x": 98,
          "y": 1,
          "z": 4,
        },
        "category":"classroom"
    },
```

Figure 5.5: Exemple de déclaration d'un lieu en JSON

Le parsage du fichier JSON est fait au début de l'exécution de l'application,les donnes sont stockées dans des dictionnaires statiques. Cela nous permet d'y avoir accès dans toutes les autres classes du programme et pouvoir utiliser les informations parses.

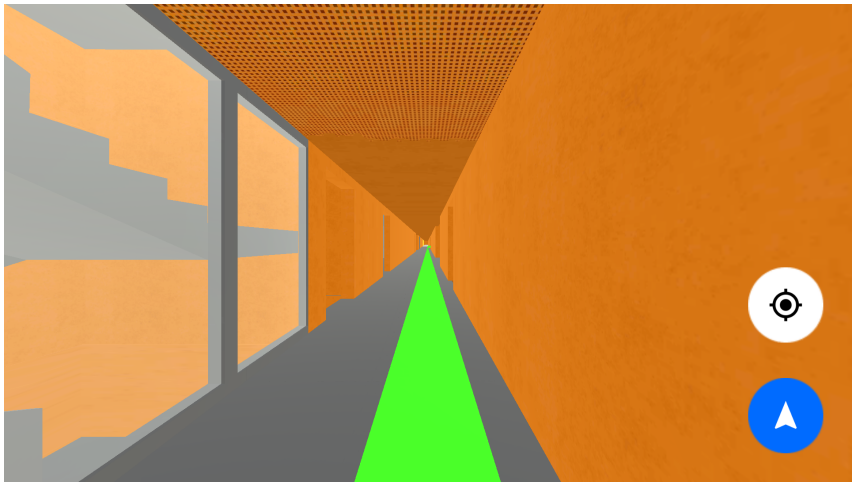

Figure 5.6: Exemple du tracer entre la salle 142 et 128

Le pathfinding dans Unity est gérée avec 2 composants, la navMap et le navMeshAgent. La navMap définie la zone ou le navMeshAgent va pouvoir ce déplacer d'un point a a B. La navMap est calculée à partir du mess des objets de la scène. En plus de cela on peut définir des zones avec des couts différents, pour favoriser ou défavoriser des zones ne notre scène. J'ai donc généré 2 zones dans ma scène avec 2 couts différents, la zone de marche normale avec un cout moyen et les assensceur avec cout plus faible. Le navMeshAgent

va calculer le chemin le plus court entre sa position et la destination donne sur une zone donne. J'ai donc ajouté ce composant au GameObject sur qui est placée la caméra. Pour le mode handicape, il suffit de dire a l'agent qu'il peut passer par la zone des assensceur, vue qu'elle a cout moindre il va automatiquement passer par la.

La dernière fonctionnalité a implémenté est le tracer du chemin, en effet il est plus agréable pour l'utilisateur de visualiser le chemin qu'il va parcourir, car il peut anticiper ces déplacements. Pour ce faire on récupérer le chemin calculer par l'agent,on va ensuite iterrerer sur les points du chemin pour construire une Line. Grâce a cela on va tracer la courbe qui representer le chemin emprunter par l'agent.

5.4 Localisation

La première chose a faire a été de ce balader dans l'école pour récupérée BSSID des access points du bâtiment et noter le positon exact dans le bâtiment. Une fois ces relevés fait ils ont était stocker dans un fichier JSON et sont parser de la même manière que pour les lieux du bâtiment. Pour récupérer les informations du téléphone sur les access points, j'utilise le plugin Android info scanner. Après l'avoir installé. Il c'est avéré qu'il ne fournissait uniquement le SSID,BSSID et RSSI auquel le téléphone l'access point est connecté. Il a donc fallu modifier le plugin pour qu'il fournisse les informations nessecaire(détail dans la partie suivant),qui est la liste de tous les access points qu'il détecte, en donnant a chaque fois le SSID,BSSID et RSSI pour chaque. Avec ces informations récupère depuis le plugin on peut commence a filtré uniquement les access points possédant le bon SSID "HE-ARC-secure". On va ensuite compare les adresses BSSID de la liste avec celle relever dans l'école, on a maintenant toutes les informations pour commencer la trilatération.

La première chose a faire est de convertir la puissance qui est en db en une distance pour ce faire une utilise cette formule :

$distance = pow(10, (RssiAtOneMeter - ReceivedRSSI)/20)$

On utilise la puissance à 1 mètre pour calculer la distance par rapport au signal. Après plusieurs tests il c'est avère qu'il impossible d'obtenir une distance correcte, en effet la puissance a 1 mètre varie selon les access point.

La fonction de localisation n'est pas allée plus loin, car les valeurs de distance obtenues étaient beaucoup trop hétéroclite pour permettre une localisation un minimum réaliste.

Chapitre 6 Probleme rencontrer

Ce chapitre détaille les problèmes recentrés et donne l'analyse qui en a été faite.
Le pathfinding implémenter dans Unity fonctionne très bien pour le projet, le problème est qu'il fonctionne trop bien. Il va toujours essayer de minimiser le chemin à parcourir, ce qui n'est pas toujours la meilleure solution au niveau du rendu. En effet il serait préférable que le chemin passe par le centre du couloir, a la place le chemin calculer frôler les murs, car cela fait parcourir moins de chemin a l'agent. Ce point pourrait être amélioré en implémentant un système de pathfinding comme cela avait été fait dans le premier projet.

Le deuxième gros problème fut le plugin Android qui ne fournissait pas les informations suffisant. Après avoir remarqué cela, j'ai tout de suite ouvert les fichiers pour essayer de comprendre comment il fonctionnait pour savoir si je pouvais le modifier. Le plugin est composé de classe C# que l on peut instancier dans nos scripts, de classe réalisant des appels sur classe java. Les classes java sont stocker dans un jar, les classes sont découpées en 2 une partie implanter la fonction qui sont appelé par les classes C# et d autres réaliser les fonctions Android. J'ai commencé par décompiler le jar grâce un outil disponible sur le web qui transforme du byte code java en fichier java. Après avoir compris comment fonctionnait la hiérarchie du plungin, j'ai pus modifier les classes Java. Le problème est venu de la recompilation du code java modifier, en effet la classe avait besoin de classe fournit par java, Android, le jar Unity et le jar en lui même. Le moyen de compile la classe modifier était d'utiliser la commande javac en spécifiant les jars cite précédemment. une fois la classe modifier on peut la replacer dans le jar du plugin l'utiliser dans nos classes C#.

La trilatération est impossible dans l'état actuel du projet, en effet après plusieurs tests, plusieurs causes ont été trouvées. La puissance n'est pas la même pour tous les access points du bâtiment, cela a été testé avec le calcul de la puissance a 1 mètre. Une solution peut être apporté a ce problème, il faudrait mesure la puissance de chaque access points du bâtiment et la note dans le fichier JSON. Comme cela on calculer la distance avec la puissance de chaque access points. Ensuite la disposition dans le bâtiment, le bâtiment est en long donc les access points le sont aussi ils sont aligne entre les couloirs et les salles de cours. Pour obtenir des résultats précis il faut que les access points soit disposer en triangle et sur le même plan que le téléphone, ce qui n'est pas du tout le cas dans le bâtiment de l'école, car ils sont accrochés au plafond. Pour toutes ces raisons, la trilatération est impossible dans l'état actuel des choses. À titre d'exemple un access point qui étatise a 22 mètres du téléphone étatise détecter a plus de 50 mètres. Plusieurs soluciton pourraient être envisage comme de couple le positon avec celle du GPS ou utiliser un autre réseau comme Galileo.

Chapitre 7 Amélioration et développement futures

7.1 Amélioration

Le premier point a améliore est le framerate, pour l'instant il oscille entre 40 et 50 fps, il sera agréable pour les yeux qu'il reste constant a 60 fps. Pour ce faire il faut réduire les paramètres graphiques de l'application(distance d'affichage, ombre temps réel,qualité des textures).Queque optimisation pourrait avoir lieu a ce niveau. Précalculer les ombres est les afficher en temps que texture serait une bonne optimisation a faire(cette amélioration a été teste, mais était très long donc abandonner).Une page tutoriel existe sur Unity pour optimiser les performances de notre application sur les plateformes mobiles.[8]

Un autre point a améliore sont les indications sur les salles et les zones interdites, en effet la fonction de guidage nous amène a la bonne destination, mais il serait intéressant d'avoir un texte devant la porte de la salle qui indique sont numéro. Cette fonctionnalité avait commencé d'être implémentée, mais des problèmes d'orientation du texte sont apparus. Pour l'instant l'application bloque l'accès aux open spaces du 2e étage, il serait intéressant de créer du mode un mode étudiant/public et un mode personnel qui pourrait avoir accès aux open spaces. L'implémentation de cette fonctionnalité peut être facilement réalisé dans l'état d' actuel du projet.

Enfin la localisation, plusieurs pistes peuvent être explorées pour essayer le l inplementer. Une étude plus approfondit sur la puissance des access points pourrait être faite pour améliore le calcul de la distance, car les access points n'ont pas tout la même puissance. Coupler la trilatération avec le GPS pour améliorer la position. Ces pistes peuvent être exploitées, mais elles doivent être testées et cela prend du temps.

7.2 Conclusion

Le bilan du projet est plutôt positif, le projet c'est bien dérouler la planif initiale a été respecter. Le projet a permis de découvrir le moteur Unity et plus particulièrement le développement Unity sur Android. La plupart des objectifs ont été remplis sauf pour la localisation de l'utilisateur. La localisation était un point important du projet qui n'a pus être réalisé, cet échec vient de la disposition des access points dans le bâtiment qui rende la trilatération quasiment impossible. Le résultat final est plutôt satisfaisant, l'application permet de trouver son chemin entre deux salles du Campus Arc 2 de la Haute-École Arc. La localisation de l'utilisateur serait vraiment une plus-value pour le projet, mais au vu des rechercher faite, il s'avère difficile a maitre en place. Ce projet est une preuve de concept, tous les objectifs n'ont pus être réalisés . Cela dit, le projet mériterait d'être approfondi pour être terminé et disponible au public.

Bibliographie

[1] Android info scanner. https://www.assetstore.unity3d.com/en/#!/content/44553.

[2] C documentation. https://msdn.microsoft.com/library.

[3] Determine accurate distance od signal. https://forums.estimote.com/t/determine-accurate-distance-of-signal/2858/5.

[4] Log cat. http://answers.unity3d.com/questions/492681/how-to-use-adb-logcat.html.

[5] Stackoverflow. http://stackoverflow.com/.

[6] Unity documentation. https://docs.unity3d.com/Manual/index.html.

[7] Unity forum. https://forum.unity3d.com/.

[8] Unity mobile optimization. https://docs.unity3d.com/Manual/MobileOptimizationPracticalGuide.html.

[9] Unity navigation. https://docs.unity3d.com/Manual/Navigation.html.

[10] Wireless ssid bssid essid. http://www.juniper.net/techpubs/en_US/network-director1.5/topics/concept/wireless-ssid-bssid-essid.html.

Table des figures