Université Pierre et Marie Curie Faculté d'Ingénierie

Rapport de Projet STL

A.Fernandez & S.Ung

Encadré par
V.Botbol & G.Ziat

Le 25 avril 2017

Table des matières

1 Introduction

1.1 Sujet

Nous présentons ci-dessous le sujet du projet tel qu'il est décrit aux étudiants lors de la sélection :

Dans le cadre des activités robotiques au sein du M2 STL, nous aurions besoin d'explorer la programmation sur les drones. A l'heure actuelle, toutes les réalisations robotiques du M2 STL se basent sur des plateformes mobiles terrestres. Il est souhaitable d'explorer d'autres possibilités de support. Nous proposons de nous attaquer à ce projet sous un angle "Kid games", en récoltant des données de jeux multi-joueurs et en envoyant des commandes de déplacement selon ces données sur un drone baptisé "RockAn'Dron".

Le premier de tels jeux serait un Tambour-Hero où de multiples joueurs jouent en parallèle un jeu de rythme sur des smartphones Android en mode "client". Les résultats de chaque joueur sont évalués dynamiquement sur chaque support "client" avant d'être synchronisés sur un smartphone Android en mode "serveur". Les évaluations de chaque joueur doivent refléter sa capacité à imiter un batteur dans un morceau de musique folle. A chaque synchronisation sur le "serveur"(de l'ordre de la seconde par exemple), le smartphone "serveur" décide le gagnant actuel et envoie le drone faire un pas vers le gagnant en question. Le jeu termine quand un des joueurs arrive à attirer vers lui le drone, porteur de gros paquets de bonbons, ou de nouilles, voire d'autres choses selon des envies du moment.

Ce PSTL s'adresse à des étudiants en M1 STL curieux des réalisations robotiques ouvertes au grand publique de type "tout en kit" [1]. Dépendant du niveau de chaque participant au PSTL, on commencerait avec une partie de programmation purement Android, puis, on réfléchirait à l'intégrer dans les actionneurs du drone ; ou l'inverse ; voire on ferait des choses complètement imprévues. Une réalisation, cependant, est attendue au terme du PSTL, avec poster de présentation et montage de vidéo-clip.

1. Les étudiants curieux des réalisations robotiques à partir de (quasiment) rien pourraient s'intéresser au PSTL "Prototype de robot mobile".

1.2 Enjeu

Il va principalement s'agir dans ce projet de réussir à manipuler un drone, le jeu ne sert en effet que de prétexte pour l'utilisation de ce dernier. Ainsi, la forme finale de l'application mobile peut changer selon les limites et difficultés rencontrées ; l'un des objectifs est de présenter une démonstration du travail effectué à l'occasion de la *Fête de la science*[2]. Une étude des ressources à disposition dont le kit de développement logiciel[3] fourni par le constructeur sera donc nécessaire ; le modèle du drone utilisé est le Parrot BEBOP 2[4].

Le développement est décomposé en 3 grandes parties que nous présenterons plus en détails dans ce rapport :

1. Conception du jeu mobile
2. Développement du serveur de jeu multijoueur
3. Manipulation du drone

Comme on peut le voir, c'est un environnement client-serveur que nous allons mettre en place. En effet, les joueurs, à partir de leur application mobile, représentent les clients et se connecteront à un serveur hébergé sur un ordinateur dont le rôle sera de centraliser les scores des joueurs. A partir de ces données, le serveur communiquera avec le drone pour le piloter vers l'un des joueurs.

2. du 12 au 15 octobre 2017
3. SDK : Software Development Kit
4. `https://www.parrot.com/fr/Drones/Parrot-bebop-2`

2 Développement

2.1 Client - Serveur

La première chose qu'il fallait faire était de mettre en place une architecture client-serveur le plus rapidement possible et établir un protocole de communication. Pour des raisons de simplicité nous avons opté pour un protocole textuel, dont la définition est la suivante :

Pour identifier d'où proviens les paquets tout les message (sauf le premier message "CONNECT pseudo") seront structurés par le client comme ceci :

`Paquet = [idClient] [MESSAGE]`

- **Connection** :
  Client envoie :
  - `CONNECT pseudo:string`
    Premier message envoyé au serveur.

  Serveur envoie :
  - `CONNECTOK`
    S'il reçoit ce message ce cas Client envoie : READYRECEIVE Cela permet au serveur d'attendre que le client ait bien initialisé sa socket (s'il ne reçoit pas ce message il renvoie le CONNECTOK et au bout de 4 tentatives il supprime le client)
  - CONNECTBAD Renvoyé si l'indentifiant est déjà pris.
- **Initialisation/gestion de la partie** :
  Client envoie :
  - `AULIST`
    Demande au serveur la liste des utilisateurs

  Serveur envoie :
  - `ULIST nom1 nom2 ...`
    Renvoie la liste des utilisateurs au client

  Client envoie :
  - `STARTGAMEOK`

  Serveur envoie :
  - `STARTGAME nbsec:int`
    Envoyé quand tout les clients de la partie ont envoyé STARTGAMEOK. La partie commence pour le client et le serveur au bout des nbSec
- **Partie Jeu** :
  Client envoie :
  - `SCORETICK score:byte`
    Par exemple toutes les 500 ms. Met à jour le score du joueur côté serveur, et leur avancement relatif par rapport a la piste (si jeu musical)
- **Partie Fin** :
  Serveur envoie :
  - `GAMEEND pseudoWinner:string`
    Envoyé lorsqu'un gagnant a été décidé, plusieurs idées pour définir les règles : Plus on approche de la fin de la piste, plus le drone va avancer vers le jouer au meilleur score. Ou bien, sans prendre en compte la fin

de la piste (peut finir avant, ou reboucler sur la piste mais plus vite) demande plus d'équilibrage sur le gameplay. Déconnection

Client envoie :

— `DISCONNECT`
Si jamais le client n'est pas déconnecté, nettoyer les clients qui ne jouent plus depuis un certain temps.

2.2 Jeu mobile

2.3 Serveur - Drone

3 Conclusion

4 Bibliographie