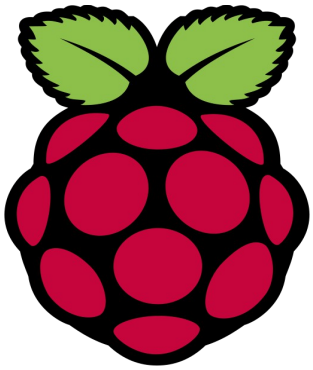
Rapport de projet

Borne de guidage pour le musée Fabre



Réalisé par
Kévin Giordani
Maxime Soustelle

Sous la direction de
David Delahaye
Vincent Berry

Année universitaire 2016-2017

Introduction

Nous avons réalisé un projet en collaboration avec le musée Fabre à Montpellier en réponse à un appel d'offre. Le projet consistait à développer une borne d'assistance au déplacement dans le musée contrôlable avec une interface. Cette interface est un Raspberry possédant un capteur ultrason, à l'aide duquel l'utilisateur peut choisir l'oeuvre vers laquelle il souhaite être dirigé. En bougeant simplement sa main au-dessus du capteur l'utilisateur peut se déplacer dans une liste d'oeuvres affichées sur un écran.

Les informations doivent ensuite pouvoir être envoyées vers un second Raspberry qui lui s'occupera de guider l'utilisateur jusqu'à l'oeuvre. Ceci est fait grâce à un écran composé de LEDs qui va indiquer, par l'intermédiaire de flèches, le chemin que l'utilisateur doit prendre. Une fois que l'utilisateur est arrivé devant l'oeuvre, l'écran va lui indiquer les flèches afin qu'il revienne au point de départ.

1 – Mode d'emploi

Afin de faire fonctionner le projet, il est tout d'abord nécessaire d'avoir deux Raspberry Pi, une qui sera utilisée pour faire le choix d'une oeuvre et l'autre vers laquelle les informations seront envoyées et qui guidera l'utilisateur.

Une connexion internet est nécessaire afin de faire communiquer les Raspberry entre elles. Ils font donc prévoir des câbles ethernet afin de les relier au réseau, ou des Raspberry 3 compatibles avec le Wifi si aucune liaison filaire n'est disponible. Il faut aussi alimenter les Raspberry, pour cela on peut utiliser un cable d'alimentation pour Raspberry classique pour la Raspberry contrôlant l'interface, et une batterie portable pour la Raspberry indiquant le chemin car elle devra être transportée par l'utilisateur.

Pour la Raspberry qui gère le choix de l'utilisateur il faut utiliser un shield GrovePi afin de pouvoir ensuite y insérer un capteur à ultrason branché sur le port D4 il sera utilisé pour le choix de l'utilisateur. La Raspberry qui gère le déplacement doit être équipée d'un Sense Hat qui possède tous les éléments nécessaires au fonctionnement tels que les LEDs et un bouton.

Enfin, pour ce qui est de la partie logicielle, il faut que les outils I2C, senseHat et python3.0 soient installés sur les Raspberry afin de pouvoir utiliser les capteurs. Les scripts commandeScript.exp et borneScript.exp doivent être modifiés avec les adresses IP des raspberry et les accès aux comptes. Les chemins doivent être enregistrés dans le fichier interfaceSenseHat.py. Les fichiers communications.txt doivent tous deux être initialisé a 0. Dans le fichier mainBorne.c il faut initialiser la variable distanceMax à la distance entre le capteur ultrasons et l’objet le plus proche lorsque qu’il n’y a pas la main de l’utilisateur au-dessus. Compiler les fichier mainBorne.c et mainCommande.c et enfin lancer les deux programmes sur les deux raspberry.

2 – Moyens Materiel

Pour ce projet nous utilisons deux Raspberry, une pour l'interface et la seconde pour le guidage de l'utilisateur. Nous avons choisi d'utiliser le shield GrovePi avec un capteur à ultrason car cela rend le déplacement dans la liste des oeuvre et le choix d'une oeuvre intuitif pour l'utilisateur. Nous avons également utilisé une carte Sense Hat pour la Raspberry gérant le déplacement car elle possède les capteurs nécessaires à notre projet. On y trouve les LEDs qui vont afficher les flèches, un bouton sur lequel l'utilisateur va appuyer pour indiquer sa progression et un accéléromètre couplé à un gyroscope qui vont permettre de détecter l'orientation de l'utilisateur.

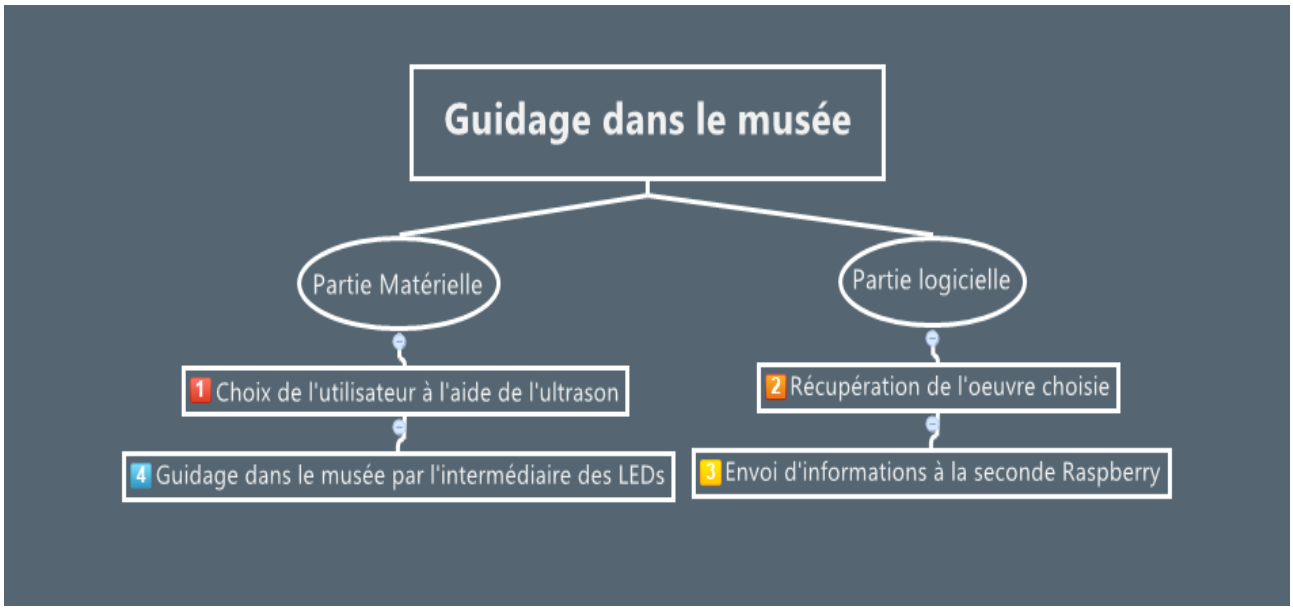
3 – Moyens Humains

Pour la réalisation du projet nous avons tout d'abord cherché une idée ensemble et nous nous sommes mis d'accord pour répondre à l'appel à projet du musée Fabre. Nous avons ensuite réalisé les spécifications fonctionnelles et la configuration du Raspberry ensemble afin d'avoir une base pour notre projet.

Par la suite Kévin Giordani s'est occupé des fonctionnalités concernant la Raspberry guidant l'utilisateur tandis que Maxime Soustelle a travaillé sur l'interface de choix d'une oeuvre.

Nous avons ensuite mis notre travail en commun et nous avons travaillé ensemble pour le reste du projet, c'est-à-dire la liaison des deux parties et la finalisation du projet.

4 – Architecture



5 – Le Code

Le projet se divise en sept fichiers, deux fichiers de communications entre les raspberry et cinq fichiers de code, dont deux en C, un en python 3.0 et deux scripts exp. Les fichiers C sont les fichiers principaux.

Dans le fichier mainBorne.c on trouve toutes les fonctions nécessaires à l'affichage de l'interface pour l'utilisateur, à savoir « show() » qui permet d'afficher l'interface, «menuUp() » qui permet de monter dans le menu afficher, « menuDown() » qui permet de descendre dans le menu afficher et enfin « pause() » qui permet de mettre en pause l'affichage du menu lorsque la commande est en train de guider quelqu'un. On trouve aussi la fonction « distance() » qui renvoie la distance entre le capteur ultrason et l'objet le plus proche de lui. On trouve aussi des fonctions d'écriture et de lecture dans un fichier, « writeFile() » et « readFile() ». Le « main() » est une boucle infinie qui affiche le menu en attendant que quelqu'un mettent sa main au-dessus du capteur, une fois que l'utilisateur a sa main au-dessus du capteur, la fonction va monter ou descendre dans l'affichage selon la distance actuelle de la main par rapport à sa distance précédente, une fois la main retirée l'indice de l'élément courant dans le menu est écrit dans un fichier et envoyer via SSH a la seconde raspberry. L'envoi par SSH se fait grâce au script borneScript.exp qui va exécuter un scp vers la seconde raspberry pour qu'elle puisse lire le fichier et réagir en fonction. Le main se met alors en attente d'une réponse de la commande pour recommencer son exécution.

Dans le fichier mainCommande.c on trouve une fonction « script() » qui permet l'exécution d'un programme python en donnant en paramètre un tableaux rempli avec le nom du programme courant, le programme python, la fonction à exécuter et les diffèrent paramètres de la fonction. Cette fonction est tirée de la doc python : https://docs.python.org/2/extending/embedding.html . Comme dans mainBorne.c on trouve les fonctions d'écriture et de lecture dans un fichier. La fonction « main() » est elle aussi une boucle infinie, elle attend d'abord que le fichier de communication soit initialisé avec l'indice d'une œuvre, ensuite il lance la fonction main du programme interfaceSenseHat.py avec en paramètre l'indice de l'œuvre. Il attend ensuite que le programme python ait fini son exécution pour enfin communiquer via SSH avec la première raspberry pour lui dire de recommencer son exécution et il se met en attente.

Le fichier interfaceSenseHat.py permet d'afficher les flèches à l'utilisateur, selon la valeur la flèche pointe dans une direction différente. Les chemins sont pré-enregistrés dans un tableaux de tableaux. Un tableaux cheminInverse contient les chemins à faire pour revenir au point de départ. Le tableaux arrowDirection quant à lui contient l'orientation dans laquelle doit pointer la flèche pour que même si l'utilisateur tourne sur lui-même la raspberry continue à pointer dans la même direction. On trouve la fonction « textScroll() » qui fait défiler un texte donné en paramètres, la fonction « arrow() » qui affiche une flèche pointant dans la direction donnée en paramètre, « getYaw() » renvoie la différence d'angle entre l'orientation initiale et l'orientation actuelle de la raspberry, « click () » qui renvoie 1 si le bouton a été appuyé. Une fonction permettant d'écrire dans un fichier. Et la fonction « displayNextDoor() » qui affiche la flèche dans la direction donnée en paramètre et se charge qu'elle pointe toujours dans la même direction. La fonction « main() » prend en paramètre l'indice de l'œuvre vers où elle doit guider. Elle va afficher les flèches une à une jusqu'à l'œuvre, l'utilisateur doit appuyer sur le bouton pour passer à l'indication suivante, une fois arrivée elle va afficher les flèches pour le retour, une fois finie la fonction écrit dans le fichier de communication pour permettre a mainCommande.c de reprendre son exécution.

6 – Perspectives

Le projet peut encore être amélioré, on peut notamment penser à intégrer un gps dans la commande pour permettre de guider sans que l'utilisateur ne doive appuyer sur un bouton a chaque porte. On pourrait aussi rajouter une interface vocale pour les personnes malvoyante. En ce qui concerne le code, il faudrait modifier la manière dont les chemins sont enregistrés pour permettre à l'utilisateur d'en ajouter, d'en supprimer de manière plus intuitive. De manière générale il faudrait que le système soit capable de prendre en compte les escaliers, les pièces possédant plus d'une porte du même côté. L'affichage du menu peut aussi être amélioré pour le rendre plus agréable. On pourrait imaginer que la commande durant le guidage donne des informations vocale sur l'œuvre vers la personne se dirige. Enfin Le système pourrait aussi créer des itinéraires plutôt que d'amener seulement vers une œuvres.

Pour mener a bien toutes ces améliorations, il faudrait sans doute un groupe de deux ou trois personnes travaillant pendant 20h chacun sur le projet.

La commercialisation serait possible car il aurait une réelle utilité dans les lieux comme les musées cependant les améliorations citées plus haut le rendraient bien plus attractif. Le marché pour ce type d'appareil est ouvert, peu de solutions sont offertes aujourd'hui pour répondre a ce besoin.

Le prix d'une borne et d'une commande actuellement serait au minimum de 140€ à cause des deux raspberry, du SenseHat, du shield et des capteurs. Cependant les coûts pourraientt être réduits en utilisant un gyroscope et un écran séparé à la place du SenseHat, on pourrait aussi remplacer la raspberry par un autre appareil moins coûteux. Une seule borne est nécessaire, elle coûterait minimum aux alentours de 40€ (sans l'écran) et elle pourrait contrôler plusieurs télécommandes qui coûteraient elles aussi minimum aux alentours de 40€ chacune.