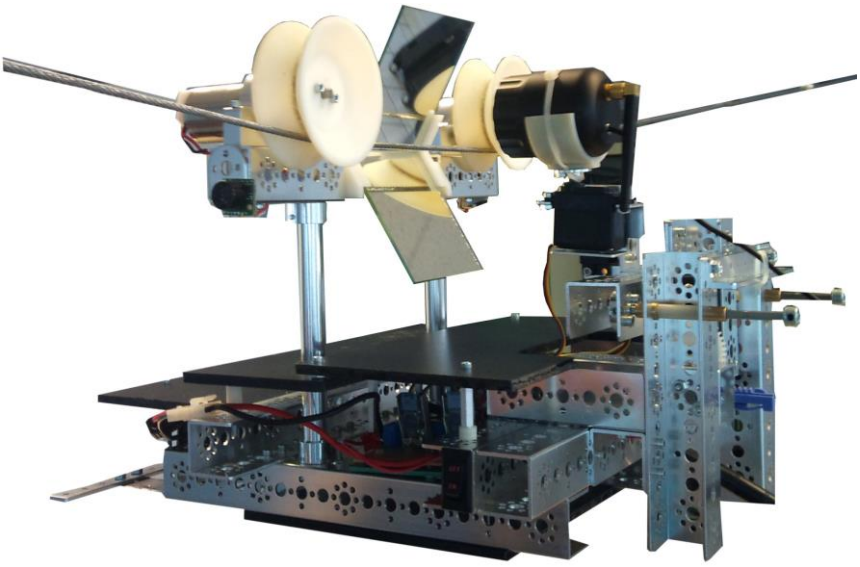
BOURY
Ludovic

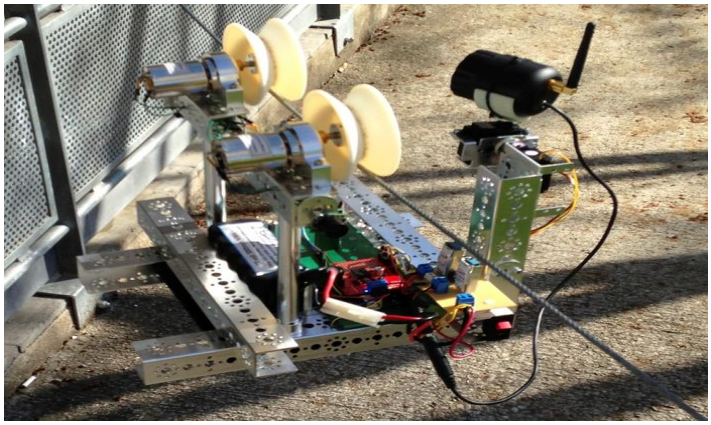
Robot inspecteur de Câbles

Spy Bird



En début d'année scolaire, Jérémy Barra, Robin Bertholet, Mathieu Delabays, Samuel Duval, Jérémy Villevieille et moi-même avons formé une équipe de six personnes pour réaliser le projet de SI. Après plusieurs propositions de nos professeurs, notre choix c'est rapidement tourné vers le «Robot inspecteur de câbles ». Ce projet nous unissait tous les six car il était pluridisciplinaire et chacun pouvait donc apporter son savoir aux autres.

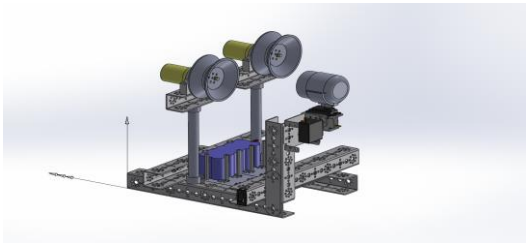
Notre but était de concevoir un robot qui, à l'aide d'une caméra pourrait inspecter tous types de câbles, et qui permettrait donc de réduire les effectifs de personnes qui risquent leurs vies chaque jour à inspecter des tyroliennes, câbles à haute tensions etc. Ce robot devait être commandé à distance par les mécaniciens ou ingénieurs grâce à une application androïd installée sur leurs téléphones ou tablettes. Le SpyBird est né :

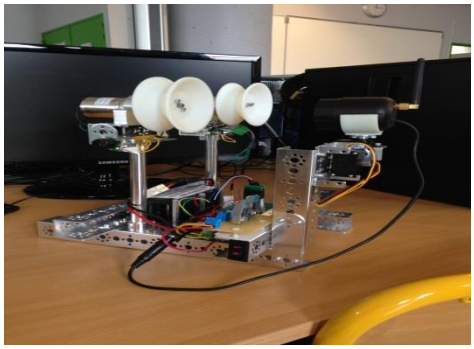


Pour ce faire, nous nous sommes répartis le travail en 3 groupes. Le premier groupe s'est chargé de tout ce qui est la partie mécanique. Le deuxième, lui, s'est interrogé sur la programmation des composants ainsi que l'électronique du système. Pour ce qui est du troisième, dans lequel je faisais équipe avec Robin, nous nous sommes chargés de la partie communication entre l'utilisateur et le robot, et avons donc réfléchit à une application androïd pour répondre à ce besoin.

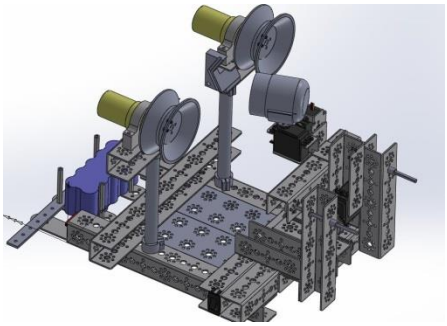
I. La Maquette

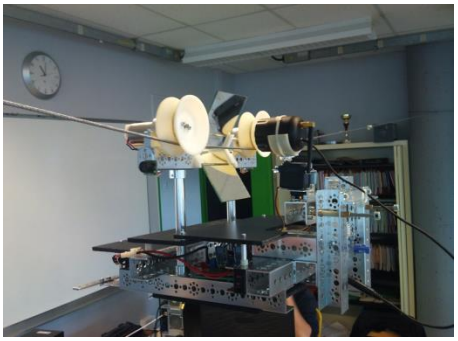
Samuel et Mathieu on beaucoup réfléchit pour élaborer une maquette qui peut avancer sur un câble où les poids sont correctement répartis, mais également sur l'emplacement des moteurs, des servomoteurs et des capteurs. Ils ont d'abord fait une première conception solidworks qui a tout de suite vu le jour en réel.





Mais, après la finale académique des Olympiades de Sciences de l'Ingénieur à Lyon, nous avons pensé à modifier cette maquette en y rajoutant un système de Mise au point qui permet, grâce à deux miroirs de voir l'intégralité du câble, le Robot que nous avons présenté à Paris était donc plus évolué :





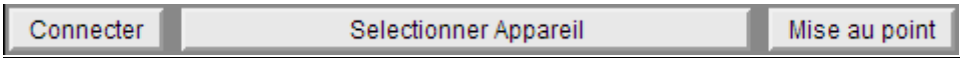
II. L'application

(Aperçu de l'application complète en Annexe 3)

Pour concevoir l'application, nous avons trouvé un site internet « MIT AppInventor » qui nous convenais parfaitement pour aboutir à ce que l'on souhaitait. Le plus dur a été la prise en main, car ce n'est pas comme processing par exemple. Le code est en fait sous forme de blocs qu'il faut emboiter les uns dans les autres. Nous avons mis beaucoup de temps et lu de documentation pour prendre le logiciel en main. Nous nous sommes entrainés en faisant à l'aide d'un didacticiel le jeu « MoleMash » où le but est en fait de taper sur la taupe qui se déplace aléatoirement, pour faire le plus de points. (Voir Annexe 1) Nous avons créé une application nous même pour s'habituer avec le site. Le but est de prendre une photo puis de la retoucher avec 5 couleurs différentes, soit grâce à des points, soit des traits. (Voir Annexe 1)

Après tout cela, nous nous sommes donc lancés dans la création de l'application qui allait nous permettre de diriger notre robot en avant et en arrière, d'orienter la caméra horizontalement et verticalement puis de faire une mise au point. Elle est constituée de 3 boutons, d'une glissière pour contrôler les moteurs et d'un Pad pour diriger la caméra.

Les Boutons :



- Le bouton « Sélection de l'appareil » qui est en fait une liste, nous permet de sélectionner l'adresse Mac de l'appareil Bluetooth auquel on veut se connecter. Une fois l'adresse Mac choisie, le bouton « Connecter » se déverrouille.
- Lorsque l'on clique sur le bouton connecter, on vérifie si le module Bluetooth auquel on veut se connecter est allumé et si le texte du bouton est « Connecter ». S'il l'est, on démarre un Compteur Clock1 : réglé à 1000ms qui, lorsqu'il est activé vérifie si Le nombre d'octets disponible pour recevoir du texte est supérieur à 0. Puis, tant qu'il est supérieur à 0, la variable recvtxt prend la valeur du texte reçu sur 7 octets, ensuite on l'affiche au-dessus du Canvas Moteur (Voir Annexe 1).

Et l'on appelle une fonction BTconnectDevice qui, si le BluetoothClient est connecter à l'adresse Mac que l'on a choisi dans la liste (Sinon, une notification s'affiche « Vous n'avez pas sélectionné un appareil à connecter »), change le texte du bouton en « Déconnecter » et met une notification « Appareil Connecté » à l'utilisateur.

Sinon, le compteur s'arrête, le BluetoothClient se déconnecte, le texte du bouton revient en « Connecter », et une notification « Appareil Déconnecté » apparait.

(Voir Annexe 2)

- Le dernier Bouton est le bouton « Mise au Point », il nous permet d'avancer la caméra pour observer de façon nette les reflets du câble qui étaient flou en position reculée, dans les deux miroirs grâce à un servomoteur, un pignon, deux glissières et une crémaillère. Lorsque l'on clique sur ce bouton, si la variable MiseAuPoint est égale à 40, alors, on la met à 120, et inversement. Ses deux nombres nous sont venus en faisant des tests à tâtons, on voulait que lorsque la caméra était avancée, les glissières de soient pas en butées mais qu'il y est un léger espace (de 1 à 2 mm) entre le bâtis et celles-ci. Et de même lorsqu'elle est reculée, car au début nous avions mis des valeurs trop élevées et, le fait que les glissières butent faisait que le pignon sortait à moitié de la crémaillère.

La Glissière Moteur :

Pour faire avancer le robot, nous avons opté pour une glissière, le principe est simple, plus vous aller à droite, plus le robot avance, plus vous aller à gauche, plus il recul.

Comment afficher une glissière qui suit notre doigt ?

- Après avoir insérer un Canvas dans l'application, on a créé une variable « Moteurs », qui prend la valeur « CurrentX » (Endroit sur l'axe X du canvas au moment où l'on le touche).
- On a ensuite créé une fonction « AfficheCouleurCanvasMoteur » qui nous permet de nous repérer lorsque l'on dirige notre robot. Lorsque on est entre 40 et 60% de la largeur de l'écran, la barre devient rouge cela veut donc dire que le robot est à l'arrêt. Lorsque l'on passe au-dessus des 60%, elle devient cyan, ce qui veut dire que l'on va à droite (ou à gauche, selon de quel côté on place le robot sur le câble). Puis, en dessous de 40% elle devient rose, on se dirige donc à gauche. Mais, pour qu'elle suive notre doigt et que ce ne soit pas tout le canvas qui change de couleur à chaque fois, il a fallu que l'on dessine une autre ligne grise de la largeur de l'écran qui remplit donc la partie à droite (selon Current X) du Canvas.

  Comment définir l'octet moteur ?

- Si l'on dit que l'octet moteur=Moteurs=Current X, on ne peut pas diriger nos moteurs car la largeur de l'écran de chaque appareil est différente et ne vaut donc quasiment jamais 255 (Pour les moteurs, on doit envoyer au programme une valeur entre 0 et 255). On a donc créé une fonction « determineroctetMoteurs » qui Test si le quotient entre (Moteurs*255) et La largeur du Canvas Moteur est supérieur à 255. Si oui, alors OctetMoteurs prend la valeur 255, sinon il prend la valeur (Moteurs*255)/(LargeurCanvas). Comme ça, pour n'importe quelle largeur d'écran, on envoi au module Bluetooth une valeur entre 0 et 255.

Le Pad Caméra :

Pour diriger notre caméra, nous avons tout d'abord pensé à faire deux glissières similaires à celle du moteur, une pour la variable PAN (horizontale) et une pour la variable TILT (verticale). Cela fonctionnait très bien, mais pour plus de facilité à la diriger nous avons pensé à faire un pad directionnel, ce qui est également plus design.

Comment faire pour que l'image de l'objectif suive notre doigt ?

- On a insérer une « ImageSprite » dans le canvas caméra, c'est une image qui peut bouger (avec la fonction image.MoveTo), comme dans le jeu MoleMash. Or ici, au lieu de la faire bouger aléatoirement, on a créé 2 variables PAN et TILT qui prennent respectivement les valeurs de CurrentX2 et CurrentY2.

Comment définir l'octet PAN ?

- Exactement pareil que pour celui du moteur or, comme ici on parle d'angles, on vas seulement de 0 à 180. Mais les calculs sont les mêmes, il suffit juste de remplacer 255 par 180 et LargeurCanvasMoteur par LargeurCanvasCaméra.

Comment définir l'octet TILT ?

- Avant, il était pareil que le PAN, or depuis que l'on a la mise au point, cela ne sert à rien d'aller de 0 à 180 pour faire haut-bas et, de plus si l'on inclinait trop la caméra, elle touchait le montage.
- On a donc fait le calcul : OctetTilt = 90-((60*TILT)/(HauteurCanvasCamera)) puis, on a dit que si il était supérieur à 90 alors, il prenait la valeur 90 et s'il était inférieur à 30 alors il prenait la valeur 30.

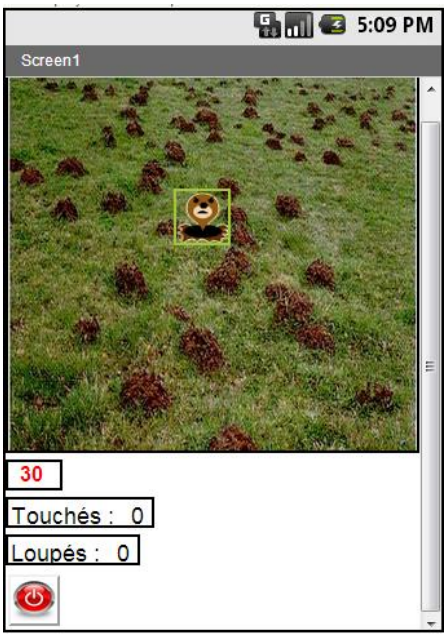
Pour envoyer nos 4 octets au programme Arduino, nous avons créé une fonction « Envoyer ». Dans celle-ci, on teste une variable ou plutôt a boolean nommée « Change » (à chaque fois que l'on touche l'un des deux Canvas où que l'on appui sur le bouton de la mise au point, change=True.), si elle est égale à True et que le Bluetooth est connecté alors, on envoie les 4 octets, sinon on affiche une notification « Appareil Bluetooth non connecté », dans tous les cas, change redevient égal à False.

III. Conclusion :

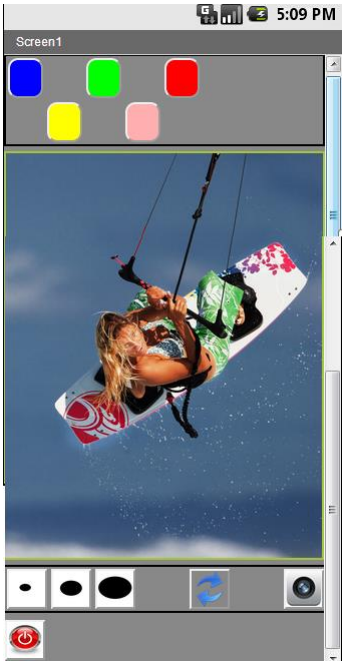
Pour conclure, je peux dire que je suis vraiment fière du travail réalisé par l'ensemble de notre équipe, car oui nous sommes une véritable équipe, soudée et déterminée à vouloir toujours essayer de faire de notre mieux pour que notre projet soit le plus aboutis possible. Déjà que de travailler ensemble était un plaisir, mais le fait que l'on ait en plus participer à un magnifique concours a été réellement enrichissant pour chacun d'entre nous, et même si nous n'avons pas gagner à Paris, le fait de se remettre en question après Lyon, de tout donner pour la Finale Nationale et d'apprendre de nos erreurs nous a fait grandir. Je remercie mes professeurs car, sans leur soutiens et leurs précieuses aides, notre projet ne serait pas ce qu'il est devenu.

Annexe 1

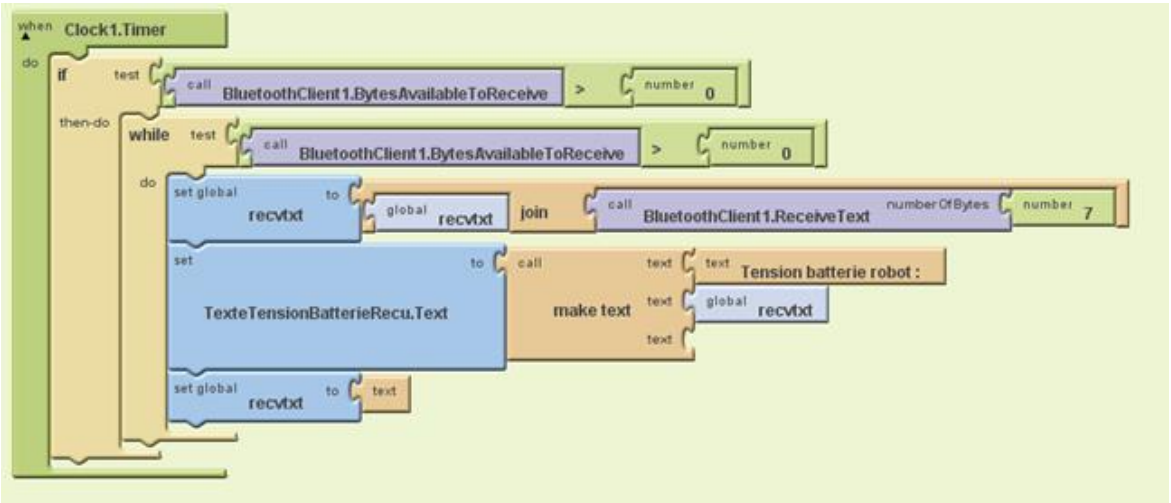
Mole Mash :



Retouche Photo :

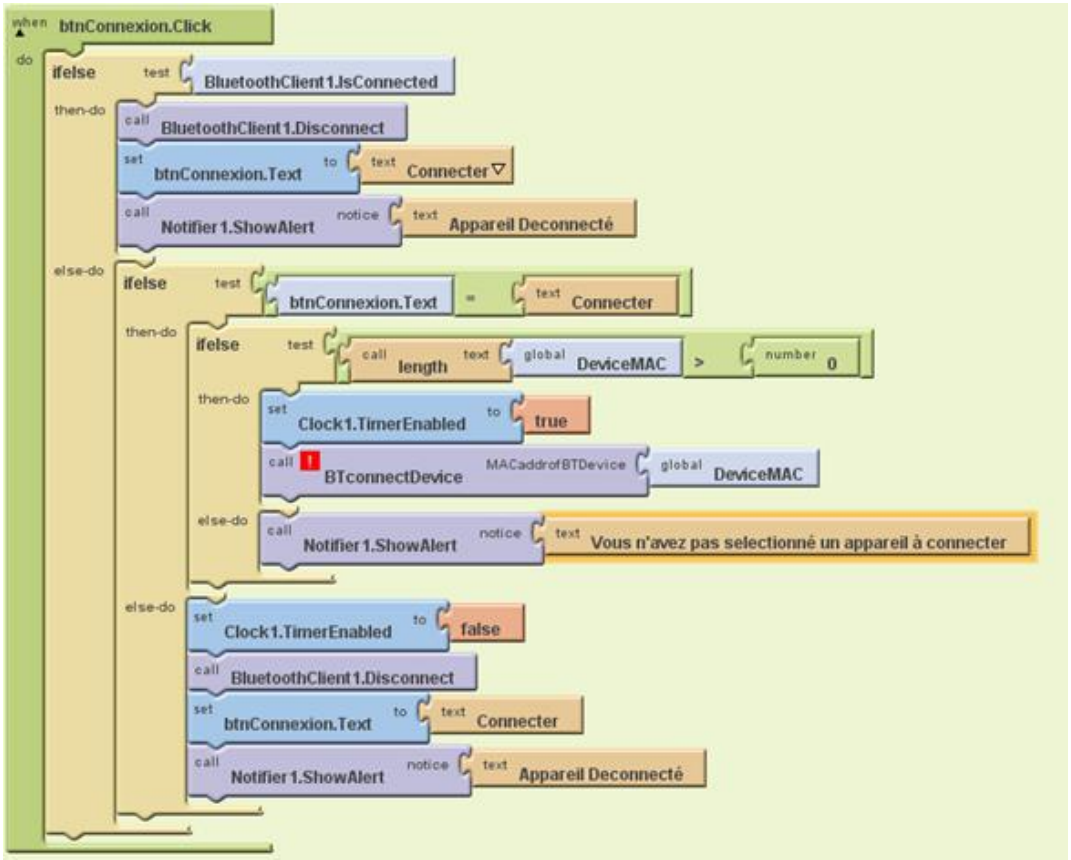


Clock1 :



Annexe 2

Bouton Connexion :



Annexe 3

Aperçu du BlockEditor de l'application complète :