Pour les futurs RICM qui travailleront sur le projet de Reconnaissance de Signatures, nous vous donnons dans ce paragraphe quelques notes et conseils qui sont ressortis lors de notre soutenance.

- L'algorithme construit gère l'homothétie, la translation, la rotation, il reste cependant quelques problèmes : pour l'homothétie, la signature doit quand même être assez grande et les points d'acquisition espacés, pour la rotation, nous avons calculé un angle de rotation entre les deux signatures en calculant l'angle entre les troisièmes vecteurs des deux signatures ce qui ne fonctionne pas toujours très bien.

- L'algorithme que nous avons construit se base sur une moyenne de similitude entre les deux signatures à comparer sans tenir compte de l'accélération du client à certains points de celle-ci, on nous a fait remarquer que cet algorithme ne permettait pas d'empêcher la copie de la signature par un imposteur, pour prendre en compte l'accélération de l'auteur de la signature, il faudrait considérer la norme des vecteurs calculés (en prenant par exemple le produit scalaire).

- De plus, notre algorithme ne fonctionne correctement pour un seuil important que pour de la reconnaissance de formes (et non pas pour des signatures plus complexes), il faudrait un seuil plus faible d'acceptation ce qui, étant donné les problèmes de considérer la moyenne des similitudes, augmenterait grandement le taux de fausses-acceptations. Par conséquent, une solution pourrait être de comparer la similitude vecteur par vecteur et de rejeter la comparaison si cette dernière est trop faible.

- En ce qui concerne l'échantillonnage des signatures, il est possible de le choisir à l'aide d'une variable globale dans la classe « Biometrie », cette dernière n'est pas réglable par l'IHM, seulement à travers le fichier.
  Pour comparer deux signatures, on doit s'assurer que ces dernières ont toutes deux le même nombre de points. Pour cela, on modifie la seconde signature selon deux cas :
  - Si elle possède plus de points que la première, on supprime des points à intervalle régulier afin de parvenir au même nombre de points que la première
  - Si elle possède moins de points que la première, la solution actuelle est de dupliquer certains points. La meilleure solution resterait cependant l'interpolation de points afin d'en rajouter à intervalle régulier et parvenir au même nombre de points que la première signature. Nous pensons que les résultats seraient bien plus intéressant (2 solutions d'interpolation : interpolation linéaire, ou par des splines).

  Par conséquent, il est évident que l'on aurait pu utiliser la suppression de points dans les deux cas, mais il était intéressant de traiter le second même si nous n'avons malheureusement pas eu le temps de le faire.

- Une des difficultés de ce projet est de parvenir à communiquer avec l'Arduino depuis le PC, pour cela, la configuration est différente suivant les machines :
  - Sur Windows, après avoir branché l'Arduino, vous devez installer le pilote manuellement à partir de l'interface « Gestionnaire de périphérique », le pilote se trouvant dans les dossiers d'installation de l'Arduino.

- Sur Linux, il faut ouvrir un port série et donner les droits administrateur. Il faut aussi penser à installer tous les drivers nécessaires.
- Pas d'essai sur Mac.

Ensuite, à l'intérieur du programme, vous devrez configurer le port à utiliser, dans la classe SerialTest, variable static « PORT_NAMES ». Le plus simple pour trouver le bon port est de lancer le programme arduino.exe, de mettre les sources du programme pde dans la fenêtre, uploadé le code si besoin sur l'Arduino, et lancer le Monitor. Si ça écrit des choses vous avez gagné ☺ Sinon, changez la configuration via le menu en haut.

- Une autre difficulté est la gestion du levé de stylo tout en gérant le bruit. En effet, nous avons pu constater que, pendant l'acquisition d'une signature, certains points ou autre informations apparaissaient dans les paquets sans avoir de rapport avec les points appartenant à la signature. Pour les supprimer, nous avons décidé de ne pas prendre en compte tous points à plus de x pixels de distance avec le précédent, le soucis est que cette solution devait au départ nous servir pour détecter si la signature comportait un ou des levés de stylos. Notre conseil serait de modifier le programme de récupération en Java, afin de contrôler la distance sur plusieurs points et non un seul. Cela permettrait de gérer un levé de stylo avec de nombreux points (ne marchera pas en théorie sur les points uniques, comme le point du « i »).

- Les courbes ROC construites dans le menu de Statistiques vous permettront de trouver la valeur idéale du seuil, il faut, pour effacer les données du programme initial supprimer tous les fichiers « .log » dans le dossier « logs ».

- Le montage provisoire que nous avons construit n'est pas très pratique pour la création de signature à cause de l'emplacement de l'écran.

- Faites également attention, le montage actuellement réalisé est assez fragile, j'ai eu un fil cassé, la nappe de l'écran qui s'est enlevé du composant rouge, et de nombreux courts circuits à cause des fils très proches les uns des autres. Mais ne désespérez pas, le montage marche ☺