Auprès de mon arbre – Annexe

Partie programmation :

Généralités sur le langage Arduino :

Le langage Arduino est proche du C, il se décompose, de façon générale, en 2 grandes parties :

- une partie *void setup()* qui sert à initialiser la communication entre la carte et les capteurs ainsi que la vitesse à laquelle ils vont envoyer des informations à la carte

-une partie *void loop()* qui va être exécuté en boucle par la carte, d'où son nom. On y retrouve la partie traitement du programme, à savoir les différentes boucles et fonctions qui vont former le programme.

Création du programme :

Le programme doit nous permettre de stocker les 42 mesures, ainsi que l'emplacement correspondant du capteur et le temps où la mesure fut faite.

Le problème rencontré ici est qu'il s'avère complexe de stocker des milliers de valeurs, et ce, de façon ordonnée directement dans un tableur par l'intermédiaire d'Arduino seul. On modifie alors le problème : on fait appel à *PuTTy*, un logiciel capable d'enregistrer dans un fichier texte ce qu'envoie Arduino par le port série. Ainsi, il suffit d'ordonner dans le programme l'affichage de chaque température et d'un temps correspondant à l'ensemble d'une mesure, c'est à dire qu'il nous faudra un témoin temps à chaque fois que nos capteurs auront fini d'envoyer leur valeur. Voici comment nous avons procédé pour obtenir cela :

Tout d'abord, nous nous servons d'une bibliothèque : *« DHT.h »* qui va permettre de traduire les octets reçus de chaque capteur en température.

Dans un premier temps, on va définir l'existence de nos capteurs ainsi que la vitesse de communication, qui est de 115200 bauds, pour des raisons de cohérence avec la bibliothèque utilisée.

Ensuite nous acquérons toutes les températures de tous les capteurs, en définissant à chaque fois une variable correspondant au relevé d'un capteur.

Enfin, nous affichons ces mêmes variables, en ajoutant une condition pour éviter les valeurs aberrantes, ce qui permettra d'avoir des résultats cohérents et de savoir tout de suite au post-traitement quels capteurs auraient été défaillants.

Remarque : Le code complet ne peut pas être affiché dans cette annexe car trop long.

Post-traitement, utilisation de GNU octave.

Les relevés sur le terrain étant faits, il est temps d'effectuer le post-traitement. La tâche est ici de réussir à visualiser, à partir de données brutes qui sont dans le fichier texte créée lors de

l'expérience, une série de cartes isotherme qui nous permettra de visualiser la température sous l'arbre durant toute la durée de l'expérience.

Pour cela, on fait appel à un logiciel de calcul matriciel : GNU Octave. Il possède une fonction, *contourf*, permettant de faire la moyenne des températures entre chaque nœud, ce qui lui permet d'obtenir des isothermes à l'image des cartes météo. Le problème est alors de transformer une matrice de 1 colonne pour des milliers de lignes en une série de matrices de 7 lignes pour 6 colonnes. Pour cela, on fait appel à une fonction, *reshape* qui permet de passer de cette énorme matrice à ce que l'on appelle une matrice 3d c'est à dire que la matrice est tel un livre : à chaque page du livre nous voyons une matrice de 7 lignes par 6 colonnes, et tourner une page de ce livre revient à avancer de 11 secondes dans l'expérience.

Code du post-traitement :

```
clear;clc;close all

load tomino_2_final.txt;

A=reshape(tomino_2_final,7,6,1703);

figure

contourf(A(:,:,300))

set(gca,'ydir','reverse')

colorbar
```

Mesure du vent :

Au cours de notre expérimentation visant à mettre en évidence la convection à l'échelle d'un arbre, nous avons tenté de mesurer la vitesse du vent en réalisant un anémomètre « maison ».

On monte, sur un roulement à billes, un rotor doté de 3 soucoupes. Il est aussi équipé d'un disque perforé de 3 orifices permettant d'éclairer ou non une photodiode se trouvant dans un compartiment situé en dessous de celui-ci. Lorsque la photodiode est éclairée, elle laisse passer le courant, ce qui nous permet, en la reliant à une arduino, grâce à la fonction *analogRead()* de détecter des pics d'intensités, ce qui nous permet d'en déduire la vitesse du vent en utilisant la formule du périmètre du disque.

Lorsque les brises sont trop faibles, la force d'entrainement du vent n'est pas suffisante pour vaincre l'inertie de notre dispositif.