1 NSI Interface Homme Machine

Fichier séance

Activité « Utilisation d'un joystick dans un programme en Python »

Captation de données analogiques - Interface Carte/Python

1 Objectifs

1.1 Objectif principal du TP :

Réaliser une interface entre la carte Arduino et Python.

1.2 Objectifs secondaires :

- Importer des données analogiques émises par la carte Arduino vers Python : import serial
- Envisager et mettre en œuvre des exploitations de ses données via Python et via une interface graphique pygame
- Insister sur l'importance du questionnement sur le type des données que l'ordinateur renvoie à l'homme
- Avantages/Inconvénients/Différences entre un appareil sans OS avec un petit processeur (la carte Arduino) et le PC (via Python)

2 Pré requis / place de la séance dans la progression :

2.1 Progression

	Progression NSI 1ère 2019/2020	
semaine	cours	TD
1	machine, OS, programmes – histoire	Démontage-montage machine- OS Debian – préparation des machines pour l'année
2	Html + css	Html + css
3	numérisation binaire	TD binaire+ table
4	, int, float, ascii/utf	PONG
5	Variables – if else	Variables – if else
6	for while	for while
7	PIGAME	PIGAME
8	fonctions	fonctions
9	tableaux	tableaux
10	son, image format	son, image format
11	recherche d'erreurs – entrainement sur petits exos	eval
12	Projet stegano	Projet stegano
13	Projet stegano	listes
14	listes et associées	compréhension de liste
15	html	html+javascript
16	html+javascript	POST GET
17	Réseau TCP/IP	Arduino I
18	Arduino II	Arduino III
19	robot	robot
20	projet sudoku	projet sudoku
21	projet sudoku - bibliotheque	projet sudoku - bibliotheque
22	Tris	Tris + complexité
23	Assembleur	Assembleur

2.2 Prérequis

- bonne maîtrise de Python (boucles, listes, fonctions)
- interface graphique pygame.
- codage des données : binaire, norme utf-8.
- Arduino : montages simples pour découvrir entrée/sortie digitales et analogiques + affichage des données sur l'IDE de la carte + importation vers Python des données émises par le potentiomètre

3 Articulation de la séance :

1. Montage d'un joystick sur la carte Arduino.
2. Extraction des données émises par Serial vers Python : travail sur le type des données renvoyées.
3. Travail sur l'interface graphique py.game avec les données analogiques émises par un joystick, récupérées et « digérées » par Python.
4. Extensions possibles pour un jeu.
5. Bilan de la séance.
6. QCM fin de séquence.

4 Points du programme abordées lors de ce TP :

- Interactions entre l'homme et la machine :
  - « Périphériques d'entrée et de sortie : identifier le rôle des capteurs et actionneurs »
  - « IHM : réaliser par programmation une IHM répondant à un cahier des charges donnés »
- Langage de programmation :
  - « Repérer dans un nouveau langage de programmation (ici C) les traits communs et traits particuliers à ce langage »
  - « Utilisation de bibliothèques : utiliser la documentation d'une bibliothèque (ici pygame et serial) »
- Représentation des données : types et valeurs de base (typ bytes) – importance de la norme d'encodage utf-8 sur Python

Correction des codes Python : voir document .py ci joint

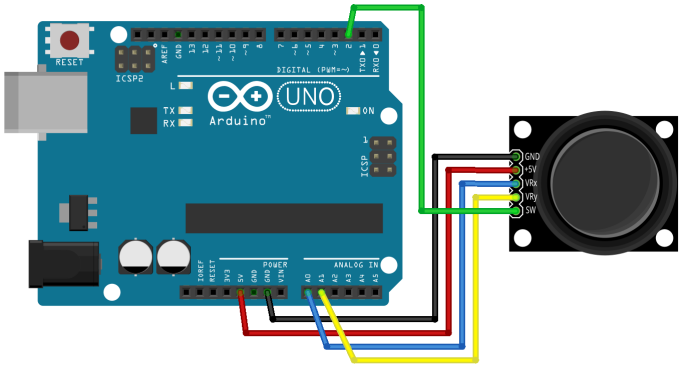
1 NSI Interface Homme Machine

Fichier élève

1 Le hardware : Arduino

Dans cette partie, nous allons connecter le joystick à la carte Arduino puis la faire transmettre à l'ordinateur les valeurs renvoyées par ce joystick.

1. Raccordez le joystick en utilisant le schéma ci-dessous :



2. . Les informations envoyées via les sorties VRx et VRy correspondent à la position du joystick. Déclarez ces différentes entrées dans le code Arduino.
3. Ajoutez les commandes nécessaires à l'envoie via le port série des valeurs VRx VRy puis observez ce qui est reçu.
4. Pour rendre l'envoie plus lisible ajouter le caractère d'espacement " " entre les valeurs puis un retour à la ligne à la fin.
5. Notez différentes valeurs de VRx, VRy suivant quelques positions dont la position neutre. Vous pourrez faire des schémas.
6. Arrêtez le défilement et observez le code de la première ligne. Comment expliquez-vous cela ?

2 Le software : Python

L'objet de cette partie est de récupérer les informations envoyées par Arduino pour les utiliser dans python.

1. Voici code permettant de lire ce qui est envoyé via le port série avec python et la bibliothèque serial :

```
from serial import *

port=Serial('Port',9600) # Placez le port o   est connect  Arduino    la place de Port
port.readline()          # Lit la premi re ligne
while True:
    line=port.readline() #Passe    la ligne suivante et la stocke
    print(line)
```

2. Notez une ligne lue par python. Quel est le type de la variable line ?
3. Expliquez en langage courant une procédure pour récupérer les informations attendues.
4. Voici quelques lignes permettant de le faire.
   Expliquez le rôle de chaque ligne

```
line=port.readline()
chaine=line.decode('utf-8')
liste=chaine.split(' ')
```

5. Récupérer les entiers x et y envoyés par Arduino.

3 Ajout du bouton

Nous n'avons pas utilisé la sortie SW du joystick. Cette partie est l'occasion de le faire avec quelques précision :

- SW correspond au bouton poussoir intégré au joystick.
- SW envoie 0 ou 1
- L'entrée SW se déclare via les commandes `int SW=2` et `pinMode(SW, INPUT_PULLUP);`.

1. Déclarez cette entrée et faite en sorte que sa valeur soit envoyée dans le port série avec VRx et VRy.
2. Récupérez-la via python.

4 Joystick et jeu vidéo

Dans cette partie, nous allons utiliser le joystick pour déplacer un objet graphique.

1. Voici un code python qui utilise la bibliothèque pygame.

```
import pygame

black = [0, 0, 0]
white = [255, 255, 255]

pygame.init()
fenetre=pygame.display.set_mode((1024,1024))        # creer fenetre de taille 1024x1024

x=300
y=300

while True:
    fenetre.fill(black)                             # fond noir

    pygame.draw.circle(fenetre, white, (x, y), 20) # balle blanche de rayon 20
                                                    # aux coordonn es x, y

    pygame.display.update()                         #rafraichit la fenetre
```

   Copiez ce code.

2. Ajoutez-y le code obtenu précédemment pour que le disque s'affiche aux coordonnées envoyées par le joystick.
3. Le but est que l'objet se déplace dans la direction donnée par le joystick mais ne revienne pas quand on le relâche.

   a) Dans ce contexte, on propose la fonction suivante :

```
1 def deplacementBalle(z,vz):          #z une coordonnee, vz valeur joystick
2       """                                                              """
3     if z <=0:
4         z=1
5     elif z>=1024:
6         z=1023
7     else:
8         z = z + int((vz - 515) / 40)
```

   Commentez les lignes 3 à 6 puis la ligne 8. Ajoutez une documentation.

   b) Intégrez cette fonction au code .

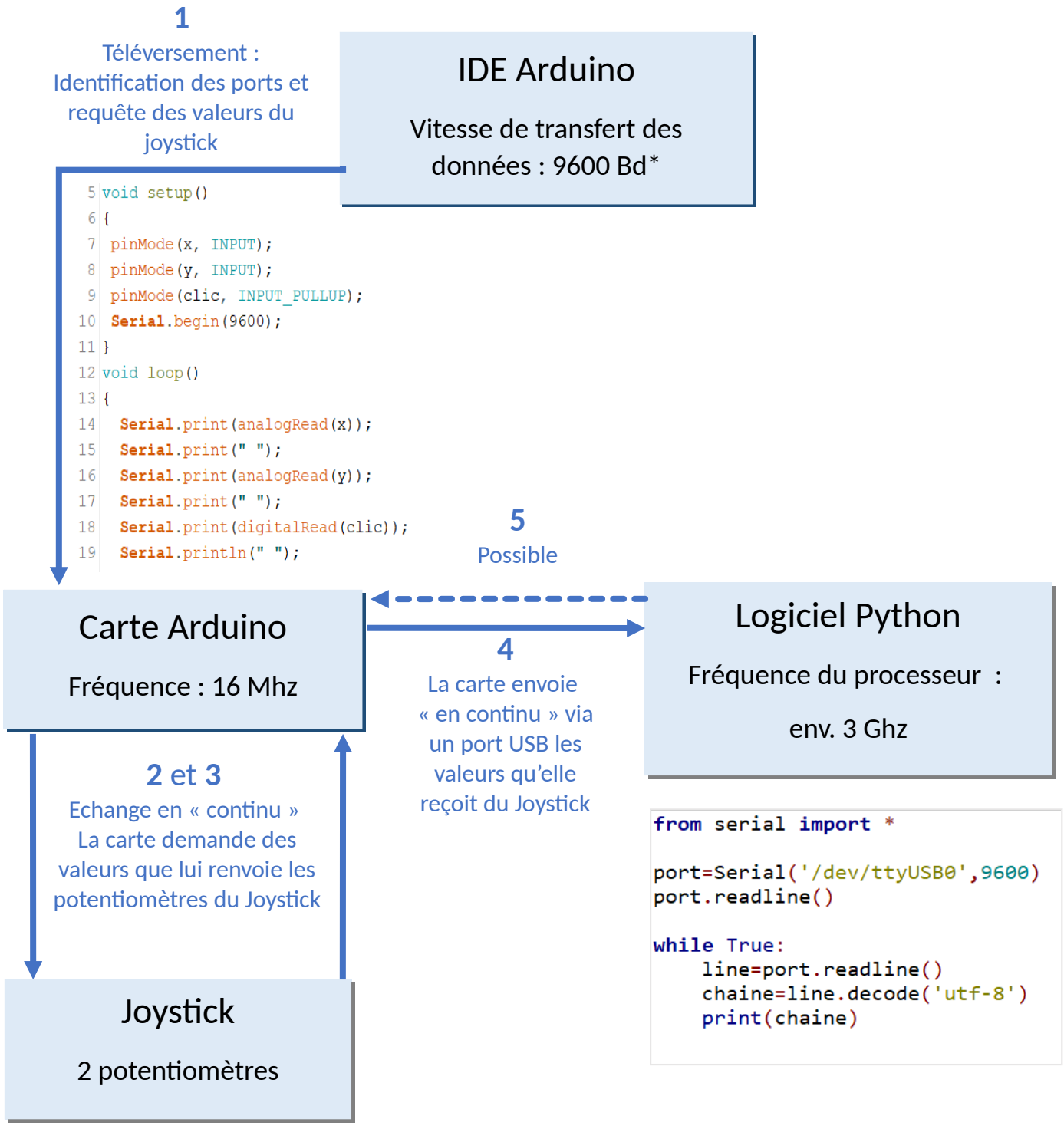
4. Ajouter un code qui permet de changer la couleur du disque de façon aléatoire quand on clique sur le bouton du joystick.

5 Un jeu

Proposez des améliorations côté hardware et/ou côté software. A vous de jouer.

Bilan de l'activité :

Utilisation d'un joystick dans un programme en langage Python




```arduino
void setup()
{
 pinMode(x, INPUT);
 pinMode(y, INPUT);
 pinMode(clic, INPUT_PULLUP);
 Serial.begin(9600);
}
void loop()
{
  Serial.print(analogRead(x));
  Serial.print(" ");
  Serial.print(analogRead(y));
  Serial.print(" ");
  Serial.print(digitalRead(clic));
  Serial.println(" ");
```

```python
from serial import *

port=Serial('/dev/ttyUSB0',9600)
port.readline()

while True:
    line=port.readline()
    chaine=line.decode('utf-8')
    print(chaine)
```

* Bd pour « Bauds » : unité de vitesse de transmission de signaux, dite aussi rapidité de modulation

L'ordinateur s'adapte à la vitesse de transfert de données de la carte Arduino qui est bien inférieure à la sienne.

1 NSI Interface Homme Machine : QCM

Fichier élève

1. Quelles sont les ordres de grandeur des vitesses d'horloge des CPU (Central Processing Unit = processeurs) pour un Arduino et un ordinateur portable standard ?

   a) de l'ordre du MHz pour Arduino et GHz pour un ordinateur portable

   b) de l'ordre du GHz pour Arduino et MHz pour un ordinateur portable

   c) 9600 pour les deux

   d) de l'ordre du THz pour les deux

2. Le convertisseur analogique-numérique (CAN) d'un Arduino donne une valeur numérique entre 0 et 1023.

   a) C'est un convertisseur 8 bits

   b) C'est un convertisseur 10 bits

   c) C'est un convertisseur 64 bits

   d) C'est un convertisseur 1024 bits

3. La fonction `digitalWrite()` permet :

   a) de programmer une E/S en sortie

   b) de changer l'état logique d'une sortie

   c) de moduler la tension de sortie d'une E/S

   d) d'écrire vers la liaison série

4. La commande `serial.print()` permet de :

   a) d'écrire sur l'Arduino

   b) d'envoyer sur la liaison série un texte

   c) d'écrire dans un tableau

   d) d'imprimer le programme

5. Que fait le code ci contre sur un Arduino qui possède une LED sur le port 13 ?

```
int LED=13;
void setup(){
    pinMode(LED, INPUT);
}

void loop(){
    digitaWrite(LED, HIGH);
    delay(100);
    digitaWrite(LED, LOW);
    delay(100);
}
```

   a) Le programme ne compile pas, il y a des erreurs.

   b) La led clignote

   c) La led monte et descend

   d) La led ne peut pas fonctionner, son mode d'entrée/-sortie n'est pas le bon

6. Que fait le code ci contre sur un Arduino ?

```
int LED=13;
void setup(){
    serial.begin(9600);
}

void loop(){
    serial.println("Bonjour ");
}
```

a) Le programme ne compile pas, il y a des erreurs de syntaxe.

b) Il est très poli.

c) Il répète 9600 fois Bonjour !

d) Il communique par wifi avec l'ordinateur à une vitesse de 9600.

7. Que fait le code ci contre sur un Arduino ?

```
//interrupteur en mode pull-up
int LED = 7;
int INTERRUPTEUR = 8;
int Bascule=0;
void setup() {
    pinMode(INTERRUPTEUR, INPUT);
    pinMode(LED, OUTPUT);
}
void loop() {
    if (digitalRead(INTERRUPTEUR)==0){
        if (Bascule==0) {
            Bascule=1;
            digitalWrite(LED,HIGH);
        }
    else {
        Bascule=0;
        digitalWrite(LED,LOW);
        }
    delay(100);
    }
}
```

a) Le programme ne compile pas, il y a des erreurs.

b) La LED s'allume uniquement lorsque l'interrupteur est poussé.

c) La LED s'eteint uniquement lorsque l'interrupteur est poussé.

d) La LED s'allume/s'éteint lorsque l'interrupteur est poussé.

8. Que fait le code ci contre sur un Arduino ?

```
int LED = 13;
void setup() {
    pinMode(LED, OUTPUT);
    for (int i=0;i<30;i++){
        digitalWrite(LED,HIGH);
        delay((30-i)*10);
        digitalWrite(LED,LOW);
        delay((30-i)*10);
    }
}
void loop(){}
```

a) Le programme ne compile pas, il y a des erreurs.

b) La LED clignote de plus en plus vite puis s'éteint.

c) La LED clignote toutes les secondes.

d) La LED clignote indéfiniment.

9. Où peut-on trouver un système d'exploitation ?

a) Sur Arduino.

b) Sur votre ordinateur.

c) Dans python.

d) Dans une entreprise.