

Exercice 5 — Mini-projet de conception protocole

Sanna Thomas, L3STI

September 16, 2024

1 Introduction

L'objectif de cet exercice est de concevoir un protocole de sérialisation utilisant le moins de mémoire possible pour enregistrer l'entrée d'une manette pour une frame. Les informations à enregistrer sont:

- La position de deux joysticks analogiques (deux coordonnées à valeurs dans $[-1, 1]$ sur les axes x et y).
- L'appui de 9 boutons d'action différents.

La fonction de sérialisation retournera une instance de `Bytes`, résultat de la fonction `struct.pack`, et la fonction de désérialisation prendra en entrée une instance de `Bytes`. Le choix de la structure de données pour représenter la manette est libre.

2 Protocole de Sérialisation

On peut représenter les données de cette façon:

- Chaque coordonnée des joysticks est représentée par un float (4 octets), sachant que chaque joystick a deux coordonnées entre -1 et 1.
- Les états des 9 boutons sont représentés par un entier non signé de 2 octets (16 bits), où chaque bit représente l'état d'un bouton (1 pour appuyé, 0 pour non appuyé).

Ainsi, la structure totale occupe $4 \times 4 + 2 = 18$ octets.

3 Implémentation en Python

3.1 Fonction de Sérialisation

La fonction de sérialisation convertit les états des boutons en un entier et empaquette les données en octets.

```
1 import struct
2
3 def serialiser(joystickGauche, joystickDroit, boutons):
4     gaucheX, gaucheY = joystickGauche
5     droitX, droitY = joystickDroit
6
7     # convertir les etats des boutons en un seul entier
8     boutonsInt = 0
9     for i, appuye in enumerate(boutons):
10         if appuye:
11             boutonsInt += 2**i
12
13     # empaqueter les donnees en octets
14     donnees = struct.pack("!4fH", gaucheX, gaucheY, droitX,
15                             ↪ droitY, boutonsInt)
16     return donnees
```

Listing 1: Fonction de Serialisation

3.2 Fonction de Désérialisation

La fonction de désérialisation dépaquette les données des octets et convertit l'entier en une liste de booléens représentant l'état des boutons.

```
1 def deserialiser(donnees):
2     gaucheX, gaucheY, droitX, droitY, boutonsInt =
3         ↪ struct.unpack("!4fH", donnees)
4
5     # convertir l'entier en une liste de booléens
6     boutons = []
7     for i in range(9):
8         masqueBit = boutonsInt & 2**i
9         boutons.append(bool(masqueBit != 0))
10
11     joystickGauche = (gaucheX, gaucheY)
12     joystickDroit = (droitX, droitY)
13     return joystickGauche, joystickDroit, boutons
```

Listing 2: Fonction de Deserialisation

3.3 Exemple d'Utilisation

```
1 # exemple
2 joystickGauche = (0.2, -0.5)
3 joystickDroit = (-1.0, 1.0)
4 boutons = [True, False, True, False, ...]
5
6 donneesSer = serialiser(joystickGauche, joystickDroit,
7 ↪ boutons)
8
9 print(f"Donnees serialisees: {donneesSer}")
10
11 joyGDes, joyDDes, boutDes = deserialiser(donneesSer)
12 print(f"Joystick gauche deserialise: {joyGDes}")
13 print(f"Joystick droit deserialisee: {joyDDes}")
14 print(f"Boutons deserialises: {boutDes}")
```

Listing 3: Exemple d'Utilisation