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.

```python
import struct

def serialiser(joystickGauche, joystickDroit, boutons):
    gaucheX, gaucheY = joystickGauche
    droitX, droitY = joystickDroit

    # convertir les etats des boutons en un seul entier
    boutonsInt = 0
    for i, appuye in enumerate(boutons):
        if appuye:
            boutonsInt += 2**i

    # empaqueter les donnees en octets
    donnees = struct.pack("!4fH", gaucheX, gaucheY, droitX,
        ↪ droitY, boutonsInt)
    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.

```python
def deserialiser(donnees):
    gaucheX, gaucheY, droitX, droitY, boutonsInt =
        ↪ struct.unpack("!4fH", donnees)

    # convertir l'entier en une liste de booleens
    boutons = []
    for i in range(9):
        masqueBit = boutonsInt & 2**i
        boutons.append(bool(masqueBit != 0))

    joystickGauche = (gaucheX, gaucheY)
    joystickDroit = (droitX, droitY)
    return joystickGauche, joystickDroit, boutons
```

Listing 2: Fonction de Deserialisation

3.3 Exemple d'Utilisation

```
# exemple
joystickGauche = (0.2, -0.5)
joystickDroit = (-1.0, 1.0)
boutons = [True, False, True, False, ...]

donneesSer = serialiser(joystickGauche, joystickDroit,
    ↪ boutons)
print(f"Donnees serialisees: {donneesSer}")

joyGDes, joyDDes, boutDes = deserialiser(donneesSer)
print(f"Joystick gauche deserialise: {joyGDes}")
print(f"Joystick droit deserialisee: {joyDDes}")
print(f"Boutons deserialises: {boutDes}")
```

Listing 3: Exemple d'Utilisation