UNIVERSITY of DEBRECEN

Dr. József Sütő
Department of IT Systems and Networks

Embedded Systems Laboratory

Lab 7: Game development on Sense HAT

Project overview

In this lab, you will create some games using the Sense HAT and some Python code. One of them is the Pong game. Pong is one of the oldest graphical games ever created, and was originally played on an oscilloscope!

The following topics will be covered in this lab:

- Advanced Sense HAT LED matrix usage

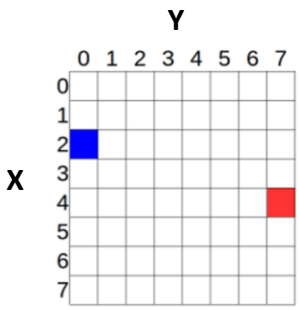
Technical requirements

The following components are required to complete this lab:

- Raspberry Pi 3 Model B+ (MicroSD card, power supply, keyboard, mouse)
- Sense HAT
- Or you can use the emulator!

1. Introduction

Games often use the coordinates **x and y** to determine where an object is on the display. x is used to set the horizontal position of an object, and y is used to set the vertical position of an object. However, when we represent a game space in Python as a list, **x and y directions will be interchanged!**



2. Catch the ball game

This game is similar to the very popular Pont game (next section), but its goal is to catch as many balls as possible. The player has a basket and the basket can be moved by the joystick. If a ball touches the basket, we suppose that the ball is in the basket. Balls are randomly arriving.

2.1. Import necessary modules:

```
from sense_hat import SenseHat

import random

from time import sleep
```

2.2. Create an object from the SenseHat class and define some auxiliary variables to seep control, basket posiotion (its rightmost point) and score tracking:

```
sense = SenseHat()

speed = 0.4

basket = [7,4]
```

2.3. Create and initialize game space:

```
w = (0,0,0)

r = (255,0,0)

b = (0,0,255)

game_space =    [w,w,w,w,w,w,w,w,
                 w,w,w,w,w,w,w,w,
                 w,w,w,w,w,w,w,w,
                 w,w,w,w,w,w,w,w,
                 w,w,w,w,w,w,w,w,
```

```
w,w,w,w,w,w,w,w,
w,w,w,w,w,w,w,w,
w,w,w,b,b,w,w,w]
```

2.4. Create a function which updates game space:

```
def update_space(x, y, colour):
    #index element from coordinate
    p = 8 * x + y
    game_space[p] = colour
    sense.set_pixels(game_space)
```

2.5. Create a handler function to the joystick left and right movements. Some data called **event** will be passed to this function. The event data the function will receive is information about has happened to the Sense HAT joystick. This will include the **time** that the stick was used, the **direction** it was pushed in, and whether it was **pressed, released, or held**. Below lines say: "When the Sense HAT joystick is moved left or right, call a handler function":

```
def left(event):
    if event.action == 'pressed':
        #the basket reached the left side
        if basket[0] – 1 == 0:
            pass
        #move basket one position left
        else:
            update_space(basket[0], basket[1], w)
            basket[1] -= 1
            update_space(basket[0], basket[1] - 1, b)

def right(event):
```

```
if event.action == 'pressed':
    #the basket reached the right side
    if basket[1] + 1 == 8:
        pass
    #move basket one position left
    else:
        update_space(basket[0], basket[1] - 1, w)
        basket[1] += 1
        update_space(basket[0], basket[1], b)
```

2.6. Joystick trigger events:

```
sense.stick.direction_left = left
sense.stick.direction_right = right
```

2.7. Initialize game:

```
sense.clear()
sense.set_pixels(game_space)
game_alive = True
```

2.8. Game loop. A ball moving in two dimensions has three essential properties: position, speed, and direction. The ball has a vertical and a horizontal coordinate on the grid:

```
while game_alive:
    #initialize position and direction of the ball
    #(x,y) – ball coordinate, d - direction
    x = 0
    y = random.randint(0,7)
    # random.choice() – randomly selects a value from a list
    d = random.choice([-1,1])
```

```
#put the ball into the game space
update_space(x, y, r)

#while the ball is in the game space
while True:
    sleep(speed)
    update_space(x, y, w)

    #ball is on the edge of  x dimension
    if x == 7:
        if y == basket[1] – 1 or y == basket[1]:
            #ball is in the basket
            update_space(x, y, b)
            score += 1
            break
        else:
            #ball is out of the space
            game_alive = False
            break

    #ball reached the right side of the space
    if y == 7 and d == 1:
        d = -1
    #ball reached the left side of the space
```

```
elif y == 0 and d == -1:
        d = 1


y += d
x += 1
update_space(x, y, r)
```

2.9. At the end of the game, print out the "Game over!" message and the score:

```
sense.clear()
sense.show_message('Game over!', scroll_speed=0.05, back_colour=w)
sense.show_message('Score: ' + str(score), scroll_speed=0.01,
back_colour=w)
```

3. Simplified Pong game (task to be solved alone)

Pong is one of the first computer games that ever created. Today, the Pong game is considered to be the game which started the video games industry, as it proved that the video games market can produce significant revenues.

It can be seen as a simple "tennis like" game. Its simplified version consists of one bat and the ball recoiling from the wall.

3.1. Import necessary modules (same as 2.1)

3.2. Create an object from the SenseHat class and define some auxiliary variables to seed control, bat position and score tracking. This part of code is similar to 2.2 but includes 1 extra variable which keeps track the ball's up/down direction:

```
up_down = -1
```

3.3. Create and initialize game space (same as 2.3)

3.4. Create a function which updates game space (same as 2.4)

3.5. Define joystick event handlers (same as 2.5)

3.6. Define joystick trigger events (same as 2.6)

3.7. Initialize game (same as 2.7)

3.8. Define the initial position and direction of the ball

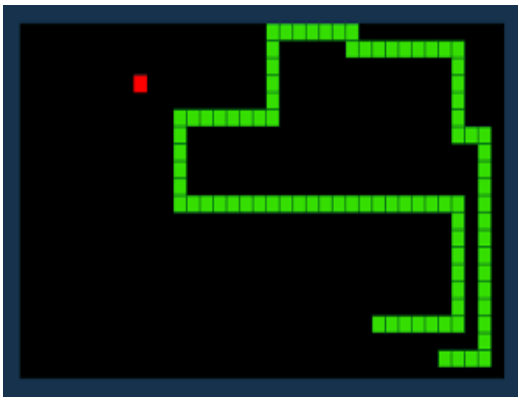
```
x = 0
y = random.randint(0,7)
d = random.choice([-1, 1])
update_space(x, y, r)
```

3.9. Define the game loop according to 2.8

3.10. At the end of the game, print out the "Game over!" message and the score. (same as 2.9)

4. Snake game (task to be solved alone)

Your task is to implement a simple version of the Snake game on the Sense HAT's LED matrix! Snake is the common name for a video game concept where the player maneuvers a line which grows in length, with the line itself being a primary obstacle. After a variant has been implemented on Nokia phones in 1998, there was a resurgence of interest in the snake concept. In this concept, a sole player attempts to eat items by running into them with the head of the snake. Each item eaten makes the snake longer, so controlling is progressively more difficult. The player loses when the snake runs into the screen border or itself.



References

[1] Raspberry Pi projects, https://projects.raspberrypi.org/en. Accessed on 10/01/2020.

This work was supported by the construction EFOP-3.4.3-16-2016-00021. The project was supported by the European Union, co-financed by the European Social Fund.