Lab Session 3

Besides the LED matrix, Sense HAT has a lot of other build-in devices. In this session, you will learn how to write programs to make use of the joystick and three kinds of sensors, i.e., the air pressure, temperature, and humidity sensors.

You should work out the following individually.

By this lab session, you will learn:

- how to make use of the joystick;
- how to measure pressure, humidity, and temperature with your Pi;

What you will need:

- The basic Raspberry Pi and the usual peripherals
- A Sense HAT

Part I: Keyboard mapping and PyGame

1. Open Python 3 from a Terminal window as sudo by:
   ```
   sudo idle3 &
   ```
2. Write a script with following code:
   ```
   import pygame
   from pygame.locals import *
   from sense_hat import SenseHat
   pygame.init()
   pygame.display.set_mode((640, 480))
   sense = SenseHat()
   sense.clear()
   running = True
   while running:
      for event in pygame.event.get():
          print(event)
          if event.type == QUIT:
              running = False
              print("GOODBYE")
   ```
3. Run this Python module. Note that we are using the pygame Python module to detect the key presses. It must be executed within a proper X window environment. *SSH would not make it work*.
4. A blank window will appear. Use the mouse to move it to one side of your screen so that the Python Shell window is also visible.
5. Keep the blank window selected but move the mouse over it, press, and release some keys on the keyboard and waggle the Sense HAT joystick. Try pressing and holding a key for a moment and then releasing it a few seconds later. You should notice that two events occur when you do this: one for the key going down, and another for the key being released. In the following, you will only use the **KEY_DOWN** event.
6. Close this blank pygame window. You should see the GOODBYE message appear in the Python Shell window but the blank window does not close.

Remark:

We are consuming the pygame event queue using the **`for event in pygame.event.get():`** syntax. This will loop through all keyboard and mouse events that occur. Inside the loop, we display what the event was and then test to see if the event type is `QUIT`. If so, we set running to `False` which causes the while loop to end and the program to finish. The program should print a line of text in the Python Shell window whenever we move the mouse, click the mouse, and press or release a keyboard key.

Part II: Detect movements of the joystick

The Sense HAT joystick is mapped to the four keyboard cursor keys, with the joystick middle-click being mapped to the Return key. This means that moving the joystick has exactly the same effect as pressing those keys on the keyboard. Remember that the down direction is with the HDMI port facing downwards.

Consider how a joystick might work. You can use the LED matrix to help you think about it. Let's make a pixel white and use the joystick to move the pixel around the 8x8 matrix. To do this you can use an event to detect a keypress. For example, if a key is pressed DOWN what steps need to happen to move the pixel?

1. Set the initial pixel by using the following:
```
x = 3
y = 3
sense.set_pixel(x, y, 255, 255, 255)
```
2. Comment the **`print(event)`** line that you used in the previous section and add the code for the DOWN key. Insert the code below at the same indentation level:
```
if event.type == KEYDOWN:
    if event.key == K_DOWN:
        sense.set_pixel(x, y, 0, 0, 0)
        y = y + 1
        sense.set_pixel(x, y, 255, 255, 255)
```
3. Save and run the code. You should be able to move the pixel point down using the DOWN key or the joystick. If you keep going, you will eventually see this error:
```
ValueError: Y position must be between 0 and 7
```
4. Our y value can only be between 0-7, otherwise it is off the edge of the matrix and into empty space. So that is the reason why the error happens; the Sense HAT does not understand values outside this range. Our code just keeps adding 1 to y every time the DOWN key is pressed. Thus we need to stop y going above 7. To avoid this, adding the syntax **`and y < 7`** (and y is less than 7) to the key direction test:
```
if event.key == K_DOWN and y < 7:
```
Save and run the code again; this time, the code should not allow the point to go beyond the edge of the screen.
5. Now that this works, you will need to add the other directions for the joystick. Where you have **`if event.key == K_DOWN:`** in your code, you can also use:

   a. **`K_UP`**
   b. **`K_LEFT`**
   c. **`K_RIGHT`**
   d. **`K_RETURN`** (reset to the initial position of the pixel)

6. We can add a section for each direction. When you run your code, you should now be able to move the pixel point anywhere on the matrix. Try this out.

Part III: Measure the pressure and humidity

1. Write a new script with following code:

```
from sense_hat import SenseHat
sense = SenseHat()
sense.clear()
pressure = sense.get_pressure()
print(pressure)
```

2. If you see the error **Pressure Init Failed, please run as root / use sudo** on the last line in red, it means you have not followed the instructions above. You should see something like this:

```
Pressure sensor Init Succeeded
1013.40380859
```

   If you get 0 just run the code again. This sometimes happens when you use the pressure sensor for the first time.

3. Just before the print(pressure) line, add this line and re-run:

```
pressure = round(pressure, 1)
```

4. Add the following lines for humidity check:

```
humidity = sense.get_humidity()
humidity = round(humidity, 1)
print(humidity)
```

   You should see something like this:

```
Humidity sensor Init Succeeded
34.6
```

5. Think about how you might use the matrix to display the humidity. One way is to divide 64 (the number of LEDs in the matrix) by 100, then multiply that by the percentage of relative humidity, which gives you the number of LEDs you should turn on. 100%, for example, would be all 64 LEDs turned on. To do this, you need to build up a pixel list of the right number of light vs. dark pixels, and then call the **set_pixels** function:

```
from sense_hat import SenseHat
sense = SenseHat()
sense.clear()
on_pixel = [255, 0, 0]
off_pixel = [0, 0, 0]
while True:
   humidity = sense.get_humidity()
   humidity = round(humidity, 1)
   if humidity > 100:
       humidity = 100.0
   pixels = []
   on_count = int((64 / 100.0) * humidity)
   off_count = 64 - on_count
   pixels.extend([on_pixel] * on_count)
   pixels.extend([off_pixel] * off_count)
   sense.set_pixels(pixels)
```

Part IV: Measure the temperature

1. Temperature sensing can be executed with a similar syntax: **`temperature = sense.get_temperature()`**. Try to print it out.
2. The Sense HAT has two temperature sensors. One is built into the humidity sensor and the other is built into the pressure sensor. You can choose either one to use, or you could use both and average the results. Try the following functions instead of **`get_temperature`**:
   - **`get_temperature_from_humidity`** (uses the humidity sensor, get_temperature is a short version of this)
   - **`get_temperature_from_pressure`** (uses the pressure sensor)
3. Think about how you could show the temperature information on the LED matrix in some way. The obvious choice would be to use the **`show_message`** function, but, while this would work, there are probably better ways to do it. For example, you can:
   a. Use **`set_pixels`** to display some predefined colors based on ranges that the temperature falls in. For example,
      i. 10 to 15 degrees -> blue;
      ii. 15 to 20 degrees -> green;
      iii. 20 to 25 degrees -> yellow;
      iv. etc.
   b. Use **`set_pixels`** to display a single color but change the brightness of red (0 to 255) based on the measured temperature. For example, 10 degrees corresponds to (100, 0, 0), and 20 degrees corresponds to (200, 0, 0).
   c. Use the **`set_pixel`** function to display a bar that moves up and down similar to a thermometer.
4. Below is some starter code for the final suggestion above. This code will display a bar that has a range of 8 degrees Celsius. The maximum it can display is 31 (hardcoded; feel free to edit this) and so the minimum is 31 - 8 which is 23. If the measured temperature goes outside of that range then errors can occur. You can add code to clamp the measured temperature to prevent these errors if you like.

```
from sense_hat import SenseHat
sense = SenseHat()
sense.clear()
tmax = 31
tmin = tmax - 8
while True:
      temp = sense.get_temperature()
      print(temp)
      temp = int(temp) - tmin
      for x in range(0, 8):
            for y in range(0, temp):
                  sense.set_pixel(x, y, 255, 0, 0)
            for y in range(temp, 8):
                  sense.set_pixel(x, y, 0, 0, 0)
```

5. Write code to display either of the first two methods of displaying the temperature.

Assignment

Demonstrate your implementation to the course instructor or TA before you go.