

Internet of Things (ECE:5550)

Spring 2023

Lab 01

Start Date: Wednesday, February 8, 2023

Due Date: Friday, February 17, 2023 by 11:59pm (ICON submission, TA check off)

This lab will cover the basics of the Raspberry Pi and Arduino platforms and their use for collecting basic sensor data. If you have any issues getting either of your devices set up, please refer to the Initial Setup guide on ICON and/or contact the instructors or the TA.

As discussed in class, we have subdivided each project team into two-person lab teams. **Please note that each lab team is expected to work independently of the other lab team from the same group.** The project and lab team assignments are listed on ICON. **Only one lab report is required per lab team (i.e., one submission is required between each team of two people).**

It is highly recommended that you carefully read this document as well as the material at the provided links to develop a first draft of your approach/code before starting the lab. This will greatly improve the efficiency of working with your lab partner.

Part 0: Raspberry Pi, First Boot Info

Changing Your Password

Please SSH into your Pi using PowerShell/Terminal as mentioned in the Initial Setup guide. Again, please let us know if you have issues with this, and we will help get you up and going! **If you did not already set up a password using the Raspberry Pi Imager**, before we get to using the Pi, you will want to change your Pi's password (if you did set one up you can skip this step):

1. In your SSH session window, type `passwd`
2. Change the password to a strong password that you'll remember
3. Reboot the Pi with `sudo reboot`
4. Connect via SSH again with your new password

When Shutting Down the Pi

Your SD card may become corrupted if you simply shut off the power to the Pi. To power down properly, please follow these steps:

1. In your SSH session window, type `sudo shutdown now`
2. Wait until the green light on the Pi shuts off and stops blinking
3. You can now remove the power cable, SD card, etc.

Part I: Raspberry Pi, Sense HAT

For this portion of the lab, you will be using a Raspberry Pi single-board computer running the Raspberry Pi operating system, formerly Raspbian OS. The Raspberry Pi is interfaced with an Adafruit Sense HAT board that contains a LSM9DS1 IMU chip, a LPS25H Pressure/Temperature sensor, and a HTS221 Humidity/Temperature sensor. The board also has a 9x9 RGB LED matrix and a joystick. Your task for this lab will involve using the sensor chips.

Begin by downloading and installing the **sense-hat** library on your pi:

1. SSH into the pi following the instructions from the *Initial Setup* guide.
2. Please ensure you have your own, non-default password (set either via the Raspberry Pi Imager or via Part 0 above)

3. After connecting to your pi via SSH with your new password, type:

```
sudo apt-get update
```
4. After the update is complete, type:

```
sudo apt-get install sense-hat
```
5. Optional: After this is completed you may need to reboot the Pi by typing

```
sudo reboot
```
6. If you did Step 5: Reconnect to the pi via SSH and log in.

Now the library to control the Sense HAT should have been successfully installed on your Pi. The library is installed in the directory `/usr/src/sense-hat`. Example code can be found in the `/usr/src/sense-hat/examples` directory. You can run the python examples in the `python-sense-hat` subdirectory to see some of the functionality of the board by cd-ing to the above directories and then running the python script. For example:

```
cd /usr/src/sense-hat/examples
ls
cd python-sense-hat
ls
python <filename>
```

where `<filename>` is the name of the python file containing the example program.

This lab will require you to program in C++. The C++ library to access the Sense HAT module is called *RTIMULib* and was automatically downloaded with the **sense-hat** library. This library is intended to work with a class of sensor modules known as Inertial Measurement Units (IMUs), including the Sense HAT board. An example program, *RTIMULibDrive11.cpp*, illustrates usage of the RTIMULib C++ library and can be found in the directory:

```
/usr/src/sense-hat/examples/RTIMULib/RTIMULibDrive11
```

To compile this program when you are in the *RTIMULibDrive11* directory listed above, type:

```
sudo g++ RTIMULibDrive11.cpp -o RTIMULibDrive11 -lRTIMULib
```

Run the program by typing:

```
./RTIMULibDrive11
```

This will pull sensor data from all of the onboard sensors and display it on the screen. Unfortunately, the *RTIMULibDrive11.cpp* example is a little difficult to follow. To assist you in understanding the *RTIMULib* library, we have provided a *slightly modified* version with more comments (denoted by "HINT"). You can find this program, named *IoTLab1Example.cpp*, on ICON. Note that when compiling the example program, you will need to link with the RTIMULib library. To do this from the command line, you can type:

```
sudo g++ IoTLab1Example.cpp -o IoTLab1Example -lRTIMULib
```

Your task for this portion of the lab is to write/modify a simple C++ program to collect and display some sensor data. Your program should collect data from the sense HAT sensors **once per second** and (1) print out the current values and (2) the average of the last 10 samples. You should collect data from the gyroscope and at least one other sensor: temperature, humidity, or pressure. Keep an average for each value you collect. Feel free to start with the provided code and modify from there.

For IMU data, please keep an average of the pitch, roll, and yaw separately. If you find you need to change the sample code, you can use the Linux function `usleep()` to control the rate at which the sensors are read. The function call `usleep(1000000);` will cause your program to sleep for one second, for example. Consult the example program posted on ICON for additional details/examples regarding use of the *RTIMULib* library to access the Sense HAT module. Another useful resource that (as of the time of writing this lab) may be useful is browsing the source code at <https://github.com/RPi-Distro/RTIMULib>.

To get yaw, pitch, and roll individually from FusionPose, do the following:

```
RTFLOAT roll = imuData.fusionPose.x() * RTMATH_RAD_TO_DEGREE
RTFLOAT pitch = imuData.fusionPose.y() * RTMATH_RAD_TO_DEGREE
RTFLOAT yaw = imuData.fusionPose.z() * RTMATH_RAD_TO_DEGREE
```

Some Hints for Programming on the Pi

- You can use VSCode to remote in to and easily create files, write/run your programs, etc. directly on your Pi. You can use the VSCode [Remote Development Extension Pack](#) to accomplish this. More details about this workflow have been posted on the ICON page.
- The Pi has Vi and Nano installed for editing text files in the command line. If you have never used either of these, the following tutorials will get you started:
 - <http://www.howtogeek.com/102468/a-beginners-guide-to-editing-text-files-with-vi/> (Vi)
 - <http://www.howtogeek.com/howto/42980/the-beginners-guide-to-nano-the-linux-command-line-text-editor/> (Nano)
- If you prefer Vim editor, you can install it by following the instructions at:
 - <https://www.raspberrypi.org/documentation/linux/usage/text-editors.md>
- **It is possible to copy-paste code from your computer into your SSH terminal window and the other way around.** This is one way you can get the sample code from your local machine onto your Pi, for example. To do so, you may just be able to copy and paste normally from your local machine into an open file editor on the Pi. If it does not work, a simple Google search should provide info for your chosen editor. For Vim, please follow the steps below:
 1. Copy the text on your local machine
 2. In your SSH session, open the file you want to copy into: e.g., `vim code.cpp`
 3. Type `:set paste` to enter Vim's paste mode
 4. Press the `i` key and then paste your code
 5. Exit paste mode, save and exit Vim by pressing ESC followed by `:x`
 6. See <https://stackoverflow.com/a/2514520> for more

The procedure to copy-paste the text from the SSH window back to your machine is similar. Just select the text in the SSH terminal session and use CTRL+C (Win) or CMD+C (Mac) to copy the information in the terminal window and paste to wherever you desire.



Part II: Arduino

This portion of the lab will involve using an Arduino Nano BLE Sense single-board computer interfaced with a HTS221 temperature and humidity sensor. Code for the Arduino must be developed and compiled on a host computer and downloaded to the Arduino via a USB connection. Please ensure you have followed all of the instructions in the Initial Setup guide to install the Arduino IDE, support for our board (via the Boards Manager), and the various libraries. Some basic Arduino tutorials can be found here:

<https://www.arduino.cc/en/Tutorial/HomePage>.

We can begin reading the temperature and humidity from the HTS221 on the Arduino by using its sample code. In the Arduino IDE, go to File > Examples > and under the header Examples for any board you will see Arduino_HTS221. Open either the ReadSensors or ReadSensorsImperial sketch.

Uploading code to an Arduino:

1. To upload code to an Arduino you must connect the Arduino to the host computer
2. Once connected, select Tools > Board and be sure your Arduino Nano 33 BLE is selected
3. Select Tools > Port and select the Arduino from the list of serial ports
4. Compile your code with the  button found on the top left toolbar
5. Upload the compiled code using the  button on the top left toolbar
6. View the console on the bottom of the Arduino IDE to verify code was successfully uploaded. If so, your code should be running on the Arduino
7. To view any Serial.print statements select Tools > Serial Monitor from the Arduino IDE

Tasks for this lab:

1. Collect data from the temperature and humidity sensor following ReadSensors example
2. Collect data from one of the other sensors on the Arduino following the code of another example (e.g., Examples > Arduino_LSM9DS1 > SimpleGyroscope). An overview of this Arduino device and its various sensors can be [found here](#).
3. Collect one sample every second
4. Display the current values and their averages over the last 10 samples

Lab Check-off

In addition to submitting the contents listed below you will need to demonstrate your completed lab to the TA (Brody Schwartz) and answer a few basic follow up questions to receive his check-off. All check offs will be completed during TA office hours or by appointment. **To claim a spot for your check off, please see the Files tab in the Lab Checkoffs channel on our Microsoft Team.** Prior to check offs there will be a file that the TA will populate with available times. Please put your lab members down in one of the time slots to claim it. **The entire lab team should join the checkoff if possible.** If you need to meet with Brody outside of these times, please make arrangements via Teams or email.

All lab check offs will ideally be completed during the week following the lab's due date in ICON. Please get in touch if there are any issues. It's a large class, so we will be as flexible as possible!

Questions or Issues?

We are here to help! Feel free to post a question on the course Teams Questions channel, send a DM in Teams, or send an email. Please feel free to message any questions about the content of this lab or any other technical difficulties you may be having (e.g., any issues with the equipment).

Submission

You should submit the following items (one submission per lab team) to the ICON drop box by **11:59pm on Friday, Feb. 17, 2023.**

- A copy of your team's C++ program for Part 1.
- A copy of your team's Arduino program for Part 2.
- A screenshot showing output displayed by your running C++ program
- A screenshot showing output displayed by your running Arduino program