Purpose

The main goal of this lab is to give you some experience dealing with real sensor data, and to let you get some more experience writing Python code. You won't use all the parts of Python you know in this lab, but you should think carefully about what data structures and algorithms you do use. This is a two-week lab, since it's pretty complex and we want you to have time to think about it properly. It's also worth twice the amount of a normal lab.

Preparation

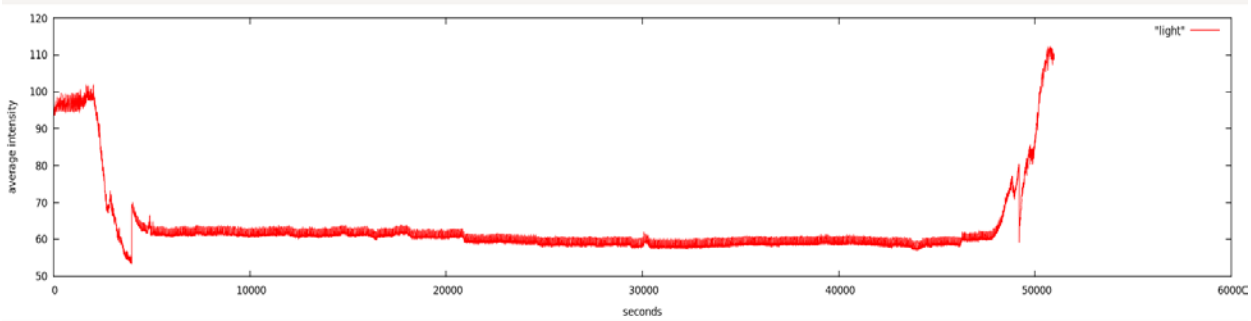
Go to the University webcam page, and look at the feed from the MU webcam. We're going to be using data from this webcam (Links to an external site.) for the lab. Watch it for a couple of minutes, and try to get a feel for what's happening (which parts of the image are static, which are moving, and how the image changes over time).

Assignment

1. Take a look at grabber.py for grabbing images from the webcam. See if you can work out what it's doing, and how to run the code from the command line. Save a couple of images, and make sure that they look the way you expect them to. For the rest of this lab, *don't modify this code*. Rather, `import` it into your own code, and use the class.

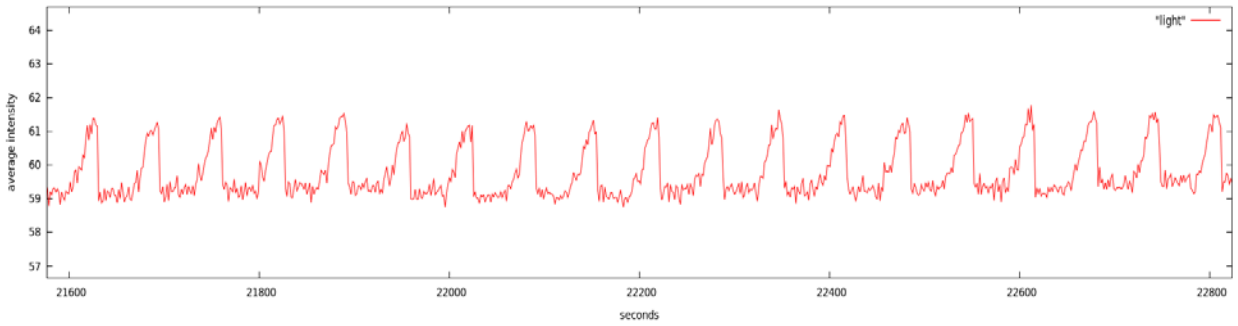
2. Write some code to calculate the average intensity of the camera image. It should grab an image, and return a single number with the average intensity, as a floating point number.

3. Make a plot of the average image intensity over time. It should look something like this



   Take a reading about once a second, and plot it against time, in seconds. Your plot doesn't have to run this long, but it should show enough time to convince the TA that the code does the right thing.

4. Wrap up your code in a class, and wrap up the code from the previous question in a function of this class.



5. If you zoom in on these data, you'll notice that they're a bit noisyWrite some more code that returns the *filtered* average intensity. You can use any filter you like, and any size, but you should pick one that causes the graph to be somewhat smooth. Provide a function in your class that returns the filtered average intensity.

6. Write a class function, called `daytime` that returns `True` if it's daytime, and `False` if it's night. Base the output of this function on the average intensity of the image, not on the actual time.

7. Write a function that grabs an image and calculates the most common color in it. The function should take no arguments, and return the most common (r, g, b) tuple. What color is this? What proportion of pixels have this color?

8. Write a function that detects motion in the image stream. This function should take no arguments. It should take two images, with a short delay between them (to make sure that they're actually different images). Then, it should calculate the differences of the corresponding pixel values in the two images, and return the average pixel difference. Write a function called `motion` that returns `True` if there's movement on the quad, and `False` otherwise.

9. **599 students (and extra credit for 499 students):** Write a function called `event` that returns `True` if there's an event happening on the quad, and `False` otherwise.

Grading

There are 15 points available in this lab for 499 students (plus 4 extra credit), and 19 for 599 students.

1. Calculate average intensity of image: 2 points
2. Correct graph: 2 points
3. Wrapped up in a class in a reasonable way: 1 point
4. Filter function: 3 points
5. Daytime function: 1 point
6. Most common color: 2 points for code, 1 point for answering questions.

7. Motion function: 2 points for writing the function correctly, and an extra point for it being robust to sensor noise.

8. Event function: 2 points for writing the function correctly, one point for it working as expected, and one point for showing it detects events.

Thoughts

1. To calculate the average intensity of a camera image, you can either use the Image class from the Python Imaging Library, or the `grab_image_data` function from the `Webcam` class. If you're new to Python, the second option is easier. If you're a seasoned Python programmer, I'd suggest trying the first. This applies to any of the questions that deal with analysis of the image.

2. `grab_image_data` returns a list of tuples, where each tuple contains the (red, green, blue) intensity values for a pixel. The pixel intensity is the average of these three values, and the average intensity of the whole image is the average of these averages.

3. The light intensity graph was on this page was started in the evening, and ran overnight. The long, low intensity is the nighttime intensity.

4. The filtered output is a bit tricky, since you're going to have to store some of the previous values in the class instance.

5. If you already know about inheritance, you can make your class inherit from `Webcam`. If not, you can create an instance of `Webcam` inside your class. We'll cover inheritance in class on Wednesday.

6. For the `daytime` function, you'll need to pick a threshold. If the average light levels are above this, then it's daytime, otherwise it's nighttime. Pick this threshold after looking at the actual values you calculate from the image.

7. To calculate the pixel differences, calculate the Euclidean distance between each pair of corresponding pixel values in the two images, then calculate the average of these values. This average is the number you want to return. You'll need `sqrt` from the `math` library for this.

8. To determine if there's an event on the quad, assume that an event will involve a tent on the central square. Check the pixels in this area (this can include some of the grass, and doesn't have to be precise) to see if they match an empty square. If not, then you can declare that there's something going on, and have the function return `True`

The Rules

Everything you do for this lab should be your own work. Don't look up the answers on the web, or copy them from any other source. You can look up general information about Python on the web, but no copying code you find there. Read the code, close the browser, then write your own code. Don't forget to submit your code!