Riley Tuttle
ELE 543
27 April 2018

Motion Detection using WiFi

Introduction/Motivation - With the growing popularity of IoT devices there are more and more sensors in our lives. Instead of adding more sensors, it might be beneficial to use existing devices to do the same sensing. Consider the wifi network. At this point in time it has become ubiquitous; free wifi at every shop and cafe. If we can use this huge pre-existing infrastructure that could be very beneficial. I propose to use it for motion detection. This is only the simple first step to fully utilizing wifi networks. In the future we could do more complicated tasks with it such as positioning of a person within a WiFi field or tracking of a person through it.

Design - The design of this is simple. Use a normal wireless access point (household router) and a wireless device (laptop or smartphone) to create a sort of tripwire to detect motion between them. Of course it would be radio waves doing the detection instead of a wire. From a networking point of view this solution lives somewhere near the Link layer (possibly adjacent to it). The two devices are already connected by the physical layer and we need to know something about that connection. Unlike most networking applications this one will actually skip over the first few layers and run almost directly on top of the physical layer connecting the two devices. Not exactly on top of because there are kernel hardware drivers that write network card information (like the connection status) to a file. My application reads that file and uses the information for the rest of the application.

Approach - The first task I had was to actually get some information about the signal strength. I found the easiest way to do that was through a terminal command (iwconfig). This terminal command returned a lot of information about the wireless devices. I again used terminal commands to grep and cut out the relevant information which was the signal strength of the access point in decibels. At this point I decided to use bash scripting throughout the entire project instead of trying to interface those scripting commands with another programming language like C or python. Next I had to devise a detection algorithm which will be discussed more thoroughly in the next section.

Detection Algorithm - First I started with an easy algorithm just to test feasibility. This initial detector was based on the amplitude of the signal strength compared to a threshold. If the strength passed over a threshold then motion would be detected:

$$X_i > thresh$$

It is important to note that since the signal strengths are decibels they come back as increasingly negative numbers as the strength gets weaker. I chose to omit the negative sign and work with all positive numbers. This was fine because I got to choose the threshold and comparison to be whatever was appropriate. Instead of testing when the

strength was below a certain threshold I could check to see if its positive was above a certain threshold.

The threshold would be found through inspection. I would look at the strength values when there was no motion and when there was motion, then I would select a suitable threshold value to delineate the values. The initial detection algorithm was not great. It did not account for noise. Any noise spike would trigger a motion detected state. In order to correct for the noise I took an average strength over time and compared that to a threshold:

$$\mu_n = \frac{1}{N}\sum_{i=0}^{N-1} x_i > thresh \qquad i = 0, 1, 2, ..., N-1 \quad N > 0$$

The idea being that the average would smooth out any spikes due to noise. The average would then be different under a motion state versus a no motion state. Again the threshold would be selected through inspection. I would look at the averages with and without motion then select an appropriate threshold value. This algorithm still had some problems. For instance it does not adjust to a new environment. The first case would be that there was no motion between the devices and then a body moved between them. You would expect that the strength or the average would go down triggering detection. If instead we started with a body between the devices that then moved out. The strength would actually increase and never fall below the threshold thus never triggering the motion. In fact the past two detectors were closer to object detectors that would be able to detect an object between the devices (assuming there was a calibration done with no object). In the hope of fixing this I made the average a moving average, averaging the last N samples:

$$\mu_n = \frac{1}{N}\sum_{i=n}^{n+N-1} x_i > thresh \qquad i = 0, 1, 2, ..., M \;\; n = 0, 1, 2, ..., M-N$$
$$N > 0 \qquad N + n - 1 < M$$

Here the hope was that the averager would eventually forget about its past and “adapt” to any current state. Unfortunately the results for this weren’t as good as I had hoped but I noticed that I was moving towards a change from the average: or a deviation from the mean. For the next detector I only needed to change the previous detector very slightly to get a variance detector. Again I decided to make this a moving average of the variance:

$$\sigma_n^2 = \frac{1}{N}\sum_{i=n}^{n+N-1} (x_i - \mu_n)^2 > thresh \qquad i = 0, 1, 2, ..., M \;\; n = 0, 1, 2, ..., M-N$$
$$N > 0 \qquad N + n - 1 < M$$

Note that the average μ_n can be taken over a different number of samples than the variance effectively providing me with two adjustable parameters not including the threshold.

This algorithm would smooth out noise while also adapting to changing initial conditions as with a body starting off between the devices. I decided to continue with this algorithm because it produced somewhat reliable results and developing more robust algorithms would require the ability to do more mathematics operations (bash scripting is limited in its mathematical operations).

Results - After experimenting with my three parameters, I can pretty accurately detect motion on the scale I was aiming for. A human sized movement between the access point and the device. For example it could detect a human walking through. It might detect a hand wave but it would not detect an insect flying through. Unfortunately this evaluation of the performance is purely anecdotal. I could not devise a way to conveniently test the performance. Even the metric of performance would be hard to define. I would say that it should be correct detections of motion over total motions, however I would then have to define what motion was. I started trying to do that but there were too many variables to account for. For instance I started to define it as human body walking between the devices at walking speed, but there are many different human bodies and different ways and speeds of walking. Not to mention what clothes the person is wearing. I think the only real way to test it would be to set it up across the doors of a mall so as to get a random sample of all the variables that I listed. However this would not fit the “convenient” qualifier that I was looking for.

What I learned - I learned a pretty good amount of bash scripting which has already been useful to me in some other work. I was also very surprised at how useful detection theory concepts could be and how well they worked in practice.

Conclusion - I consider my project to be a success in what I was attempting to accomplish. I can detect human scale motion between an access point and a device. I think with more time I could add this type of functionality to a smart lightbulb or something of the sort. Given even more time and resources I think I could achieve the positioning of a person using the same techniques. However I’m not sure I could do the device-less tracking that some are trying to achieve. Mainly because I think the processing would need to be done on the routers themselves and then I would have to find special routers.