Low-power motion detection and building control using Computer Vision

Kwabena W. Agyeman
Department Electrical and Computer Engineering, ECE
Carnegie Institute of Technology, CIT
Carnegie Mellon University, Pittsburgh, USA
kwa@andrew.cmu.edu
Low Power Motion Detection Project Webpage

Current motion detectors that control lighting and building HVAC systems use passive infrared (PIR) sensors to detect the presence of a person in a room. However, PIR sensors cannot see small amounts of motion in a room and will turn the lights and HVAC system off on the occupants of a room who are sitting and working in the room. To remedy this problem I propose that a low-power computer vision system is used along with a PIR sensor to perform variable frame rate image differencing to detect the presence of people sitting and working in the room.

Low-power, CMUcam4, PIR sensor, inexpensive

I. Introduction

Making systems smarter and more energy efficient is one of the many problems faced by our world today. Dumb building lighting systems and HVAC systems consume tremendous amounts of energy while performing unnecessary work by illuminating and cooling or heating unoccupied rooms. Efforts have been made to incorporate motion detector systems into the lighting and HVAC control loop that turn the lights and HVAC system on when motion is detected, and that turn the lights and HVAC system off after motion has not been detected for a long period of time.

These motion detection systems rely on inexpensive passive infrared (PIR) sensors to detect motion within a room. PIR sensors work by looking for large changes in the passive infrared light emitted by objects in the room. Different objects at different temperatures in the room emit different amounts of infrared light. PIR sensors are used in motion detection systems to detect when the amount of passive infrared light in a room changes when an object enters or exits the PIR sensor's field of view.

Thus, when a person, who is at a different temperature than the room, enters or exits the PIR sensor's field of view it detects that the scene has changed. However, the PIR sensor cannot detect if a person entered or exited the room. The PIR sensor only detects if something in its field of view changed. Additionally, PIR sensors cannot detect small amounts of motion within their field of view because small amounts of motion do not drastically change the amount of passive infrared light in the room.

Because of the limitations of PIR sensors, PIR based motion detectors cannot sense the presence of, for example, an office worker sitting at his or her desk who is in the line of sight of the PIR sensor. The officer worker will have to get up and move around to re-trigger the sensor to keep the lights and HVAC system on. Additionally, when the office worker leaves his or her room, the PIR sensor will not be able to detect the absence of the officer worker's presence and will instead turn the lights off based on a countdown timer from the last time the PIR sensor was triggered. This means that the lights and HVAC system will continue to run for a long while in the absence of the office worker.

To solve these problems I propose that a low-power and inexpensive computer vision system is used along with the PIR sensor to perform variable frame rate image differencing to detect the presence of the office worker sitting in the room when the PIR sensor cannot. Additionally, the low power and inexpensive computer vision system could also possibly be used to count the number of occupants in a room to control the lighting and HVAC system more precisely.

II. Proposed Implementation

The proposed motion detection system will consist of a PIR sensor and the CMUcam4 low-power computer vision system. The CMUcam4 is an embedded system development platform that pairs a microcontroller with a cell phone camera.

The CMUcam4 will use the PIR sensor to detect motion in a room when the CMUcam4 cannot see because the lights are off. When the PIR sensor detects motion, the CMUcam4 will then turn the lights on and wake up from sleeping. The CMUcam4 will then adjust to the lighting in the room and take a picture of the room for reference. After which the CMUcam4 will proceed to go back to sleep to conserve power while leaving the lights on.

After sleeping for a while (the optimal sleep time must be determined empirically) the CMUcam4 will then wake up and take another picture and compare this new picture to the reference picture that was taken previously. The differences between the two pictures will be thresholded using a look up table and the resulting binary bitmap will be stored to memory. Additionally, the CMUcam4 will compute the average of the new image and the reference image and replace the reference image with the average image. Finally, the CMUcam4 will also store the histogram of the differences to memory to adjust the threshold look up table in response to different distributions of the magnitude of differences in an image.

The CMUcam4 will then go back to sleep, wake up again after some amount of time sleeping, and repeat the above process again. The CMUcam4 will detect motion in the room by looking at the binary bitmap of all the blobs in the thresholded pixel differenced image. All binary blobs larger than a certain size and having a certain pixel density will be treated as differences. The presence of large and dense binary blobs will tell the CMUcam4 to leave the lights on in the room.

The CMUcam4 will then go back to sleep for an amount of time inversely related to the amount of motion detected for the current sample period and the previous sample periods. When the CMUcam4 detects a lack of motion over time, it will sleep for shorter periods of time increasing its sample rate. When the CMUcam4 detects motion over time it will sleep for longer periods of time decreasing its sample rate.

Because the CMUcam4 averages images of the room each time it samples, objects that stop moving will slowly disappear into the background of the image. By having a variable sample rate, the CMUcam4 will be able to more quickly react to the lack of the presence of motion in a room to turn the lights off faster by increasing the sample rate causing non-moving objects to fade into the background. More importantly, the CMUcam4 will also able to stay in low power sleep mode for longer to avoid drawing excessive amounts of power when motion is detected.

The reason for using the CMUcam4 or any other low power vision system over a fully-fledged computer for this application is that the CMUcam4 is able to draw about 100 uA sleeping and less than 100 mA while operating. Any sensor that controls the lights for the room should not draw an amount of energy comparable to the load it controls.

III. Actual Implementation

Due to time and resource constraints (because I did not have a partner for this project) the full-blown motion detection algorithm described previously was not actually implemented. Instead, I implemented a simpler version of the motion detection algorithm that just uses the histogram produced by the above process and does not use the binary bitmap.

All the steps the system goes through as described previously are the essentially the same. The system preforms image differencing on the average of frames captured at a variable frame rate that is inversely related to motion detected. However, instead of trying to detect blobs in a binary bitmap, the system just preforms a Chi-Squared test on the average of the previous histograms of pixel differences and the current histogram of pixel differences. If the Chi-Squared sum is greater than a significance level of some percentage, then motion is detected, otherwise motion is not detected. After motion is detected or not detected the algorithm will then adjust the significance level to make it harder to detect motion if motion was not detected and easier to detect motion if motion was detected. The algorithm does this to balance the effect of the variable sample rate on the motion detection process.

Once motion is detected, the algorithm will not turn the lights and the CMUcam4 off completely until motion has not been detected for at least 32 cycles. If motion is detected again in any on if these 32 cycles, another 32 cycles of no motion detected will need to pass before the algorithm will turn the lights and the CMUcam4 off completely. These 32 cycles, coupled with averaging frames, averaging difference histograms, and a variable sample rate that ranges from 10 seconds to 0.078125 seconds results in a no motion detected time of about 1 minute and 30 seconds on average before the algorithm will turn the lights and the CMUcam4 off.

IV. Results

Because the full-blown motion detection algorithm was not implemented, the CMUcam4 is not able to detect motion any better than a PIR sensor, however, it does detect motion just as good as a PIR sensor. Additionally, using the previously described algorithm the CMUcam4 draws current comparable to a PIR sensor. If the more complicated frame differencing algorithm were implemented, then the CMUcam4 would still draw power comparable to a PIR sensor while detecting motion better than a PIR sensor.

To obtain the above results, two PIR sensors from Parallax Inc. were tested. The first PIR sensor is revision A and the second PIR sensor is revision B. Both PIR sensors have two operating modes and were tested in both operating modes while being exposed to motion and not being exposed to motion. Additionally, the histogram motion detection algorithm was also tested in the same way as both PIR sensors. Listed below, are the power consumption graphs for all tests performed.

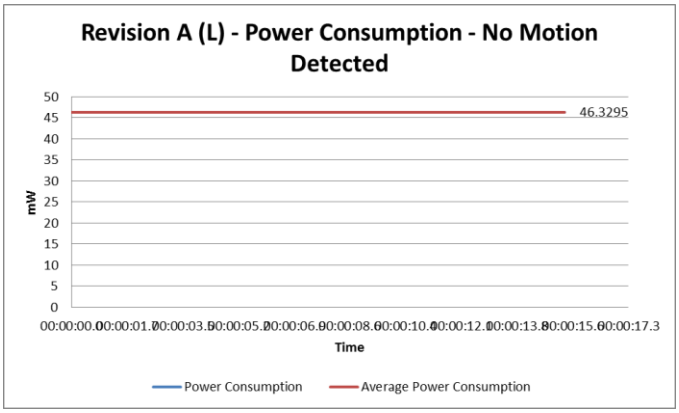

Figure 1 Revision A PIR sensor in low range mode

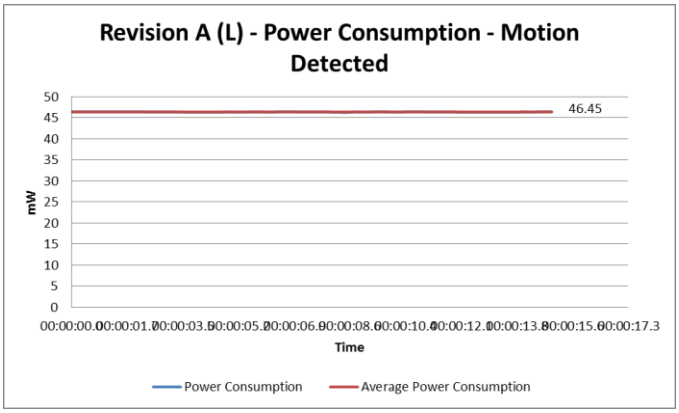

Figure 2 Revision A PIR sensor in low range mode

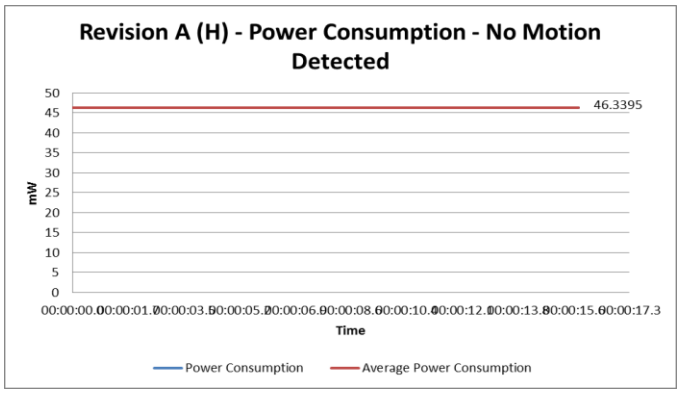

Figure 3 Revision A PIR sensor in high range mode

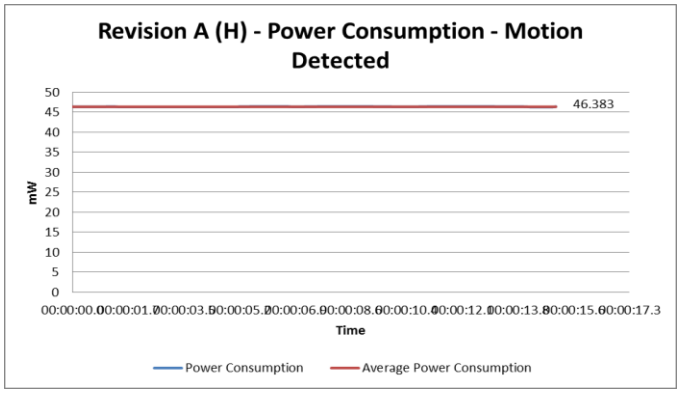

Figure 4 Revision A PIR sensor in high range mode

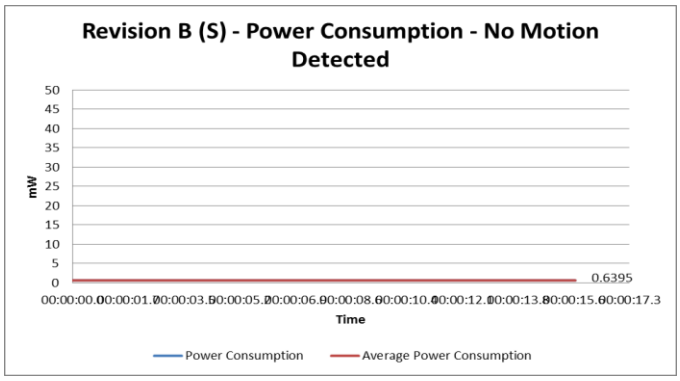

Figure 5 Revision B PIR sensor in short range mode

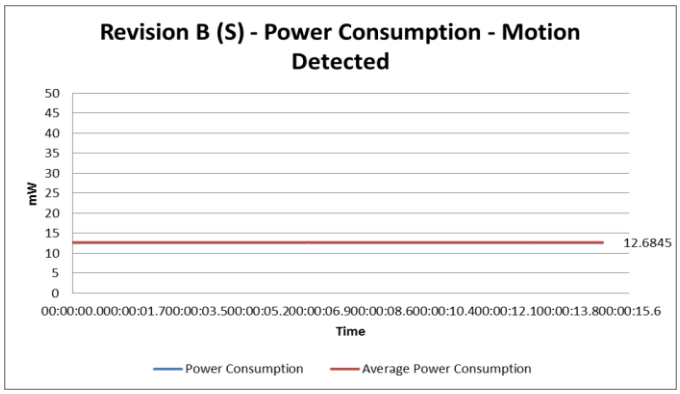

Figure 6 Revision B PIR sensor in short range mode

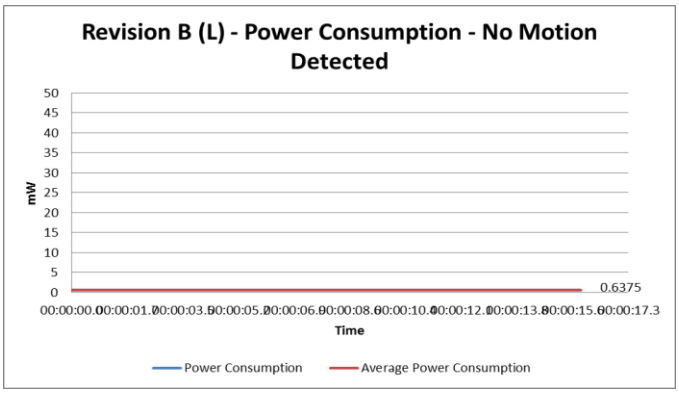

Figure 7 Revision B PIR sensor in long range mode

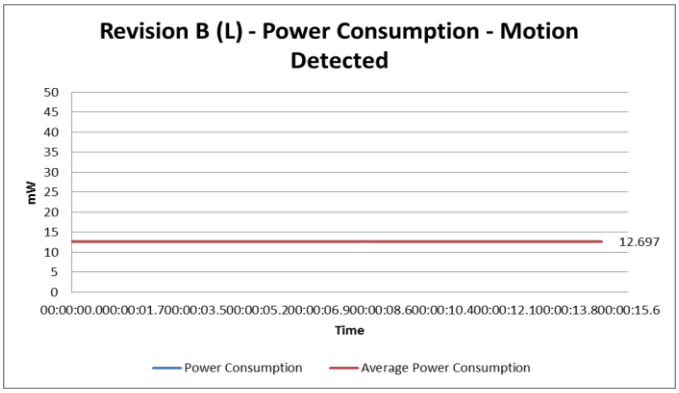

Figure 8 Revision B PIR sensor in long range mode

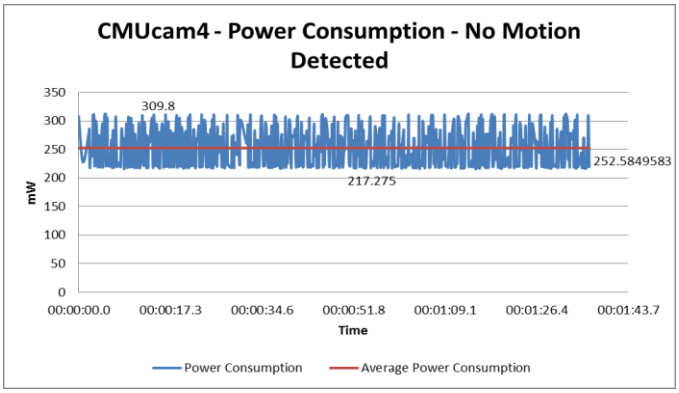

Figure 9 CMUcam4

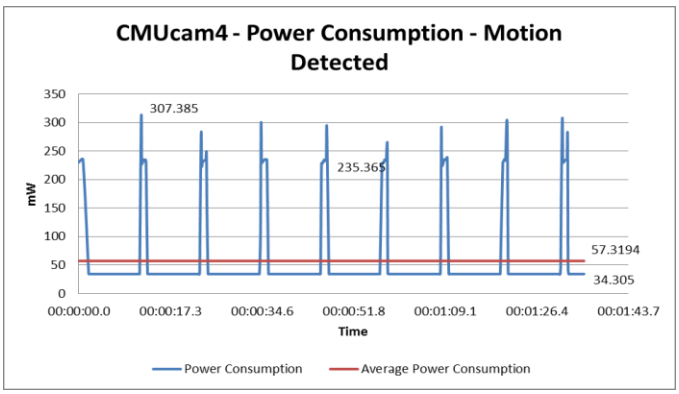

Figure 10 CMUcam4

As can be seen from the graphs Figure 1, Figure 2, Figure 3, Figure 4, Figure 5, Figure 6, Figure 7, and Figure 8 there is no power consumption difference between the two operating modes for both PIR sensors. For PIR sensor A, it draws the same amount of current whether motion is detected or not. PIR sensor A draws about 9.2 mA at 5V which is equal to 46 mW. For PIR sensor B, it draws 0.128 mA at 5V which is equal to 0.64 mW when motion is not detected and it draws 2.54 mA at 5V which is equal to 12.7 mW when motion is detected.

The CMUcam4 on the other hand in Figure 9 and Figure 10, draws about 50.4 mA at 5V when motion is not detected which is equal to 252 mW and 11.4 mA at 5V which is equal to 57 mW when motion is detected. This amount of power consumption is directly comparable to PIR sensor A's current consumption when motion is detected. Note that while the CMUcam4 draws 50.4 mA when motion is not detected, a complete motion detection system should be designed to use a PIR sensor to detect motion when the CMUcam4 does not detect motion so as to not require the CMUcam4 to operate looking for motion. This is because the purpose of the CMUcam4 in the motion detection system as described previously is to look for small amounts of motion that a PIR sensor cannot see. If the CMUcam4 decides that it detects no more motion, then the PIR sensor should be deployed to look for large changes in motion because it is likely that no one is in sight anymore sitting at their desk for example.

Additionally, the CMUcam4 can also consume power comparable to PIR sensor B if the processor clock speed was lowered from 96 MHz to 20 KHz during the sleep intervals. This was not done in the CMUcam4 code for testing purposes because dynamically changing the clock speed requires a stabilization time of about 20 ms on wakeup which would have complicated the motion detection algorithm.

V. Conclusion

Orwellian implications aside, it is possible to make a low-power camera based motion detection system that consumes similar power to a PIR sensor based detector and that can perform as well as a PIR based motion detector. With enough time and testing, an advanced motion detection algorithm could be designed that minimizes power consumption while detecting motion easily. More advanced cameras with higher resolutions and better processors in the future will further this effort to create a truly smart sensor that knows when a room is occupied and when a room is not occupied.

However, PIR sensors will be continued to be used in the near future, as a low-power camera sensor based solution is inherently more complex and costly than a simple PIR based motion detector. Until low-power processors with amazing capabilities and outstanding high resolution cameras are available for cheap, camera based motions detectors will not be as cost effective as a simple PIR based motion detector.

VI. Releated

I am the inventor of the CMUcam4 open source programmable embedded color vision sensor. I have spent two years working on the project so far. I designed the CMUcam4 hardware and software. Additionally, I wrote all the documentation for the CMUcam4 and provided Arduino Interface libraries for it. I also run the CMUcam4 website and provide help support for CMUcam4 users.

For this project, I had to rewrite the CMUcam4 firmware to make frame differencing possible. By default, the CMUcam4 performs none of the features needed for this project. The default CMUcam4 firmware performs color tracking and calculates basic color statistics among many other features, but not frame differencing due to the requirement of having to store a complete frame of the image in memory. I reused some basic camera initialization code from the CMUcam4 firmware, but beyond that, I had to rewrite the image processing pipeline code from scratch.

To implement the vision processing code on the CMUcam4, I created an interface library for the CMUcam4 and I have released the source code for the interface library on the website linked to at the top of this paper. The interface library offers all of the functionality talked about in this paper. It can perform frame differencing and averaging, histogram binning from either the average or difference, and it also produces a binary bitmap using a set of thresholds. In short, it can perform 256 different image processing operations at 30 FPS. Additionally, the interface library also has helper functions for access the above mentioned data structures and a few low level device control knobs exposed for the camera.

The interface library does not actually implement the algorithm however; I designed that functionality in a separate file that I have also released alongside the interface library on the website. This separate file also contains a debugging interface that I used to test the interface library code out and design the frame differencing algorithm.

References

[1] C. Stauffer, W.E.L. Grimson, "Adaptive background mixture models for real-time tracking", published.

[2] M. Piccardi, "Background subtraction techniques: a review", unpublished.

[3] A. Verdant, A. Dupret, H. Mathias, P. Villar, L. Lacassagne, "Low Power Motion Detection with Low Spatial and Temporal Resolution for CMOS Image Sensor", published.

[4] K. Agyeman, "CMUcam4: Open Source Programmable Embedded Color Vision Sensor", unpublished.

Carnegie Mellon University, Carnegie Institute of Technology, Department of Electrical and Computer Engineering, CMUcam