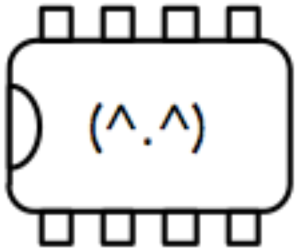
# LED Experiment ch1



**www.ZENMCU.com**

Draft 1 2022-12-13

## LED Experiment #1

This document is a supplement to com.zenmcu.0031-0_(zenai-community-of-robots).pdf.

The purpose of the test is determine if I can use an LEDs as light sensors and measure the approximate brightness of the incident light. The intended use case is to have two of them, one on each side of a robot, to detect relative brightness on both sides, and then turn towards (or away from) the light.

### Background

So RGB LEDs aren't difficult to operate, although driving them directly from an MCU takes 3 pins (4 if you want to use them as light sensors). And I have collected several articles about building short range low bandwidth half duplex LED to LED communications protocols (see the Mitsubishi Electric Research Laboratories paper, etc). The idea struck me that maybe I could just use RGB LEDs to shine each robot's color, sense the colors of other robots, and possibly even communicate via these LEDs. I'm not going to explore the use of audio for this yet because I think it would be annoying to listen to them, but its on the list for later.

(I did buy some KTD2026 I2C CRGB LED driver chips to save pins, but I have not tried them yet.)

So I did some experiments.

First off I wanted to use software PWM to drive an RGB LED. Done, no major issues, except that the brightness of the colors isn't well matched since I am not using current limiting resistors, so I had to account for that in software.

Next, I wanted to see if LEDs could effectively sense other LEDs. And, in general, they can. When reverse biased they discharge more quickly when exposed to light (of the same wavelength they emit) than they do when in darkness.

So you can time how long they take to discharge, and then decide that the discharge time is somehow proportional to the amount of light it senses. Larger LEDs take longer to discharge than smaller LEDS, so I figured smaller would be better.

I implemented light sensing using an LED as the sensor per "Very Low-Cost Sensing and Communication Using Bidirectional LEDs".

A plain old red LED (not RGB) is connected to two digital pins. The anode only needs a digital out pin, and the cathode needs an output / input-no-pullup pin.

anode --->|--- cathode ==> input/ADC

1 --->|--- 0 == forward bias == emit

0 --->|--- 1 == reverse bias == charge

0 --->|--- float == sensing timing until the pin reads a 0

Observations:

I seemed to get inconsistent readings using the same LED on two different boards under the same conditions.

In ambient lighting conditions the emitting LED must be fairly close (a few inches) to be detected. That is fine for the robot community.

But there are multiple robots and they must each have their own unique Inigo Montoya, so I switched to the common anode RGB LED.

RED - Interestingly the red junction is more sensitive than the plain old red LED. The readings seem more consistent than with the plain old LED. Ambient is a little over 1 msec, varying a few 10's of usecs. And it is quite sensitive to red light from a similar Rgb LED out to at least 2 inches (I made no precise measurements).

GRN - Measurements in ambient are single digit msecs, varying a few hundred usec. Pretty sensitive to shadows, but not that sensitive to a similar rGb LED except maybe in very low ambient light at close range (like < 1 inch).

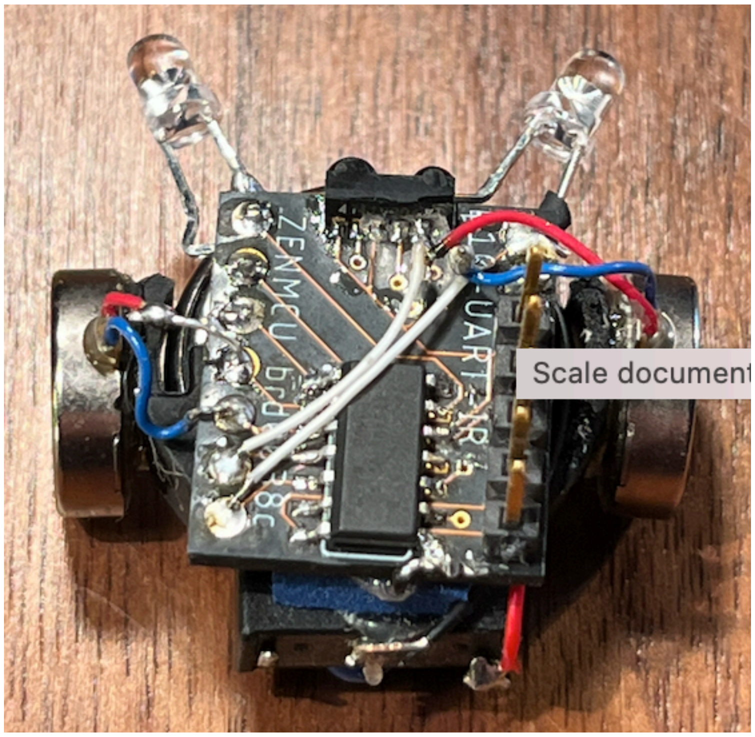
BLU - Moderately stable readings varying about a msec or so in ambient. Seems reasonably sensitive to a similar rgB LED out to at least 2 inches.

Now at this point I was emboldened and I started working on communications via LED. I didn't get it to work well and I decided to park that and work on locating an LED by sensing it with other LEDs. So I built a small robot with two blue LEDs for eyes, and used it to try to sense another blue LED.

Using two LEDs, each pointing slightly to left or right, should allow me to sense each LED's light level and then turn the robot towards the brighter one. Simple. This is exactly how Robigotchi and his contemporaries worked but they used CdS photoresistors and just sensed ambient light. They worked very well. So this should work the same, right?

*As an aside, this was an excuse to build a steerable vibrobot powered by a rechargeable lithium coin cell. In short, I demonstrated that:*

- *Two pancake style vibration motors can be driven directly off the MCU pins. Very good to know.*
- *The robot can drive forward, backward, left, and right under the control of firmware. Motion isn't extremely precise, but it is better than I expected, honestly. Also very good to know as I wondered about this for some time.*
- *The coin cell only powers the robot for a couple of minutes. I didn't time it precisely because I was doing light sensing experiments, but it didn't last long enough in my opinion. I kind of expected this, but it is disappointing. A larger battery will be ... larger, and heavier, so the robot may need to be larger.*
- *I didn't use the I/R UART for comms in the experiments so far. It's coming, just need to get there. For light sensing I just held the robot and dumped values over USB.*



Each of the two blue LEDs sensed another blue LED just fine. But the two LEDS were incredibly mismatched with respect to how fast they discharge in the same light conditions.

My attempts to calibrate them to each other have not worked well at all. I suspected that maybe I did something stupid when I wired it up, but I checked it and it seems right.

So I set up another MCU on a breadboard and ran the same sensing code on it, swapping out various similar blue LEDs. And no two of them had the even roughly the same sensitivity. And in any case I don't want to have to test a ton of LEDs to find matched pairs.

So, I decided to try out a color sensor IC. I ordered some TCS34725 I2C colors sensor boards off Tindie. I have not tested them yet, but I have written some initial driver code and wired them up. However, I discovered that they have a fixed single I2C node address. So I cannot put two in the same I2C bus. And I only have one I2C bus.

Fortunately I discovered that there is another identical part with a different address. So I hopped online to order some of those, only to learn that these parts are no longer manufactured and no one has any in stock.

So I searched for a replacement sensor that can be given two addresses and came up empty. I ended up ordering some VEML6040 color sensors, but they only have one address also.

So the plan of record is something like:

See how well the TCS34725s I already have work, as a baseline.

See how well the VEML6040s work.

Then, assuming these work well enough, decide how to perform light seeking efficiently using only one color sensor. There are two obvious options.

1) Mount the sensor stationary on the front of the robot and scan side to side.

2) Mount the sensor on a stepper motor turret and scan side to side. (This may be just an excuse to play with the Switec steppers I have laying around.)

3) Use an MCU variant with two I2C busses. The 20 pin SAMD10/11 parts have an additional I2C on PA22, PA23. So, experiments #1 and #2 may be interesting, but #3 is, I suspect, likely to be the most usable solution. And, honestly, I may need to do this anyway to get enough pins to do all of the robot-y things.

4) Use multiple MCUs. One MCU as the left half, another as the right, and a main brain that they plug into. Of course, that is getting away from the simple/minimal goal...

And there is another experiment that is worth trying; see if the color sensor can detect reflected light from a surface in front of it. I'm sure it can, but the idea would be to use it as a sort of crude short range proximity detector.

But ... these sensors are tiny. (@_@) Can I even solder them?

So the plan of record is to use the SAMD11C (14 pin) parts to test the sensors and the drivers. I will use a small Sparkfun I2C RGB LED strip to create a moving LED to track from a sensor mounted on a servo. Then I will switch to the SAMD11D so I can have two I2C busses and thus two sensors, and see how much the tracking is improved.

Watch this spot for updates.

12/10/22 Rethinking some of this... the reason to switch to the color sensors is because I was not impressed with the results of my simple LED tests. While I intend to continue testing the color sensors and RG LED drivers, I still want to pursue the plain old RGB LEDs driver by MCU pins. If I can make that work it will be the least complex solution I have contemplated.

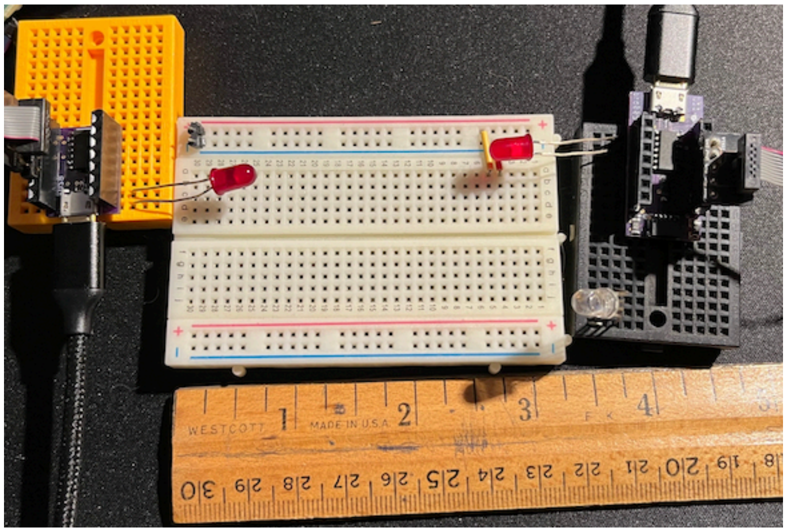
Currently the plan of record is (may be) to try to get simple, short range, low bandwidth, LED to LED communications working. The bot to bot messages are small, and high throughput isn't really an issue, and if this works then it is less expensive than using the i/r transceivers.

12/10/22 I have reread all of the papers I have collected on communicating via LEDs. I am convinced that is can be done, but so far my experiments have been unsatisfactory. The mismatching of the blue LEDs of the vibrobot is still quite tedious. I need to replicate the experiment using a less elaborate test bed. I admit that I overcomplicated things by seizing the (long awaited) opportunity to try the vibrobot experiment by combining it with the LED experiment as the means for aiming the LEDs.

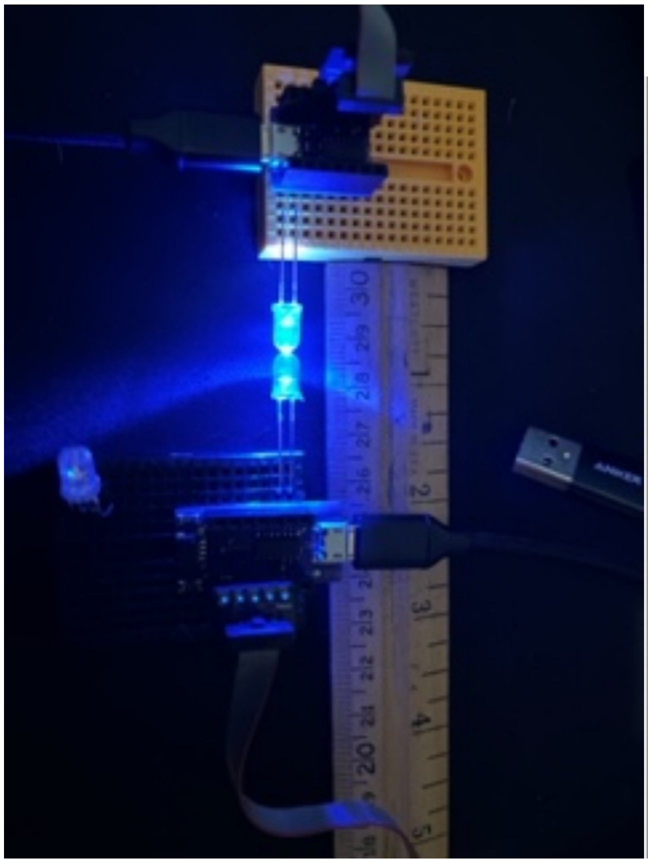
**Current State of Experimentation**

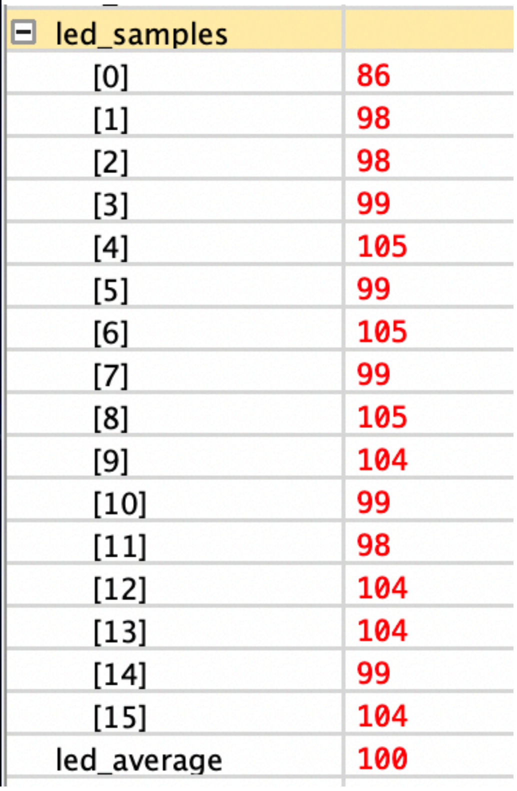
There are two MCUs with LEDs. Initially red LEDs but they were quite dim so I switched to blue LEDs.

One MCU simply lights its LED and leaves it on. The other MCU senses incoming light via its LED and records how long it takes for the sensing LED to discharge. The sensing MCU charges the LED by reverse biasing it, and then watches for the i/o pin to fall from high to low as the capacitance of the LED discharges. In darkness the LED should not discharge, and in bright light it should discharge very quickly (less than 100 microseconds, according to the Mitsubishi paper).



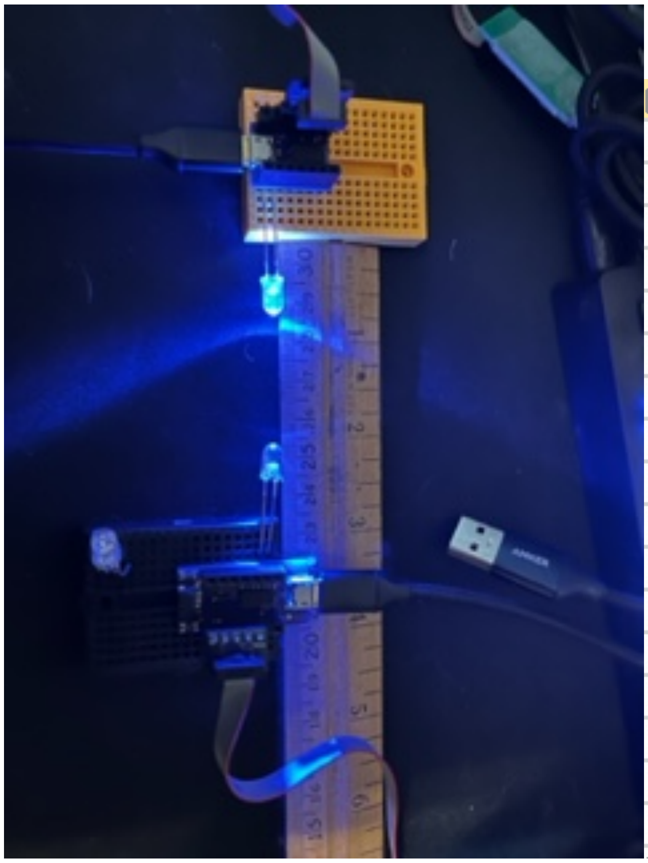
Here is a reading with the blue LEDs touching. That is about 100 usecs to discharge in dim ambient light with the LEDs touching. Smaller numbers mean 'detecting more light' and larger numbers mean 'detecting less light'.

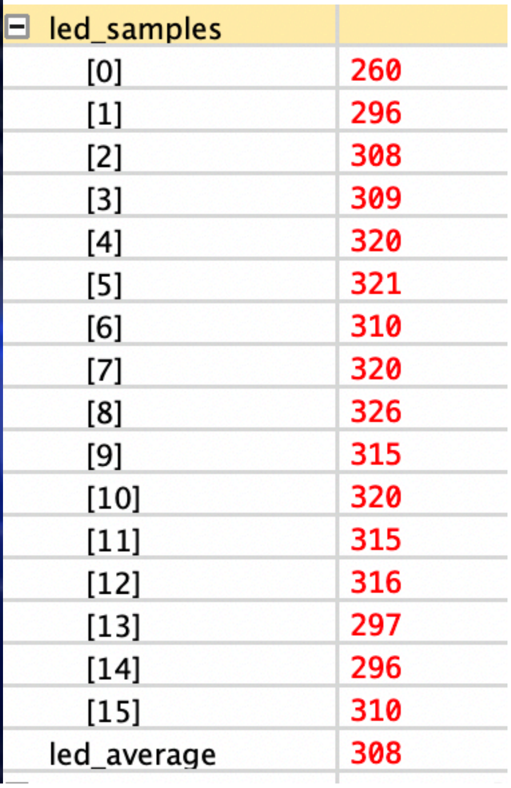




| ⊟ led_samples | |
|---|---|
| [0] | 86 |
| [1] | 98 |
| [2] | 98 |
| [3] | 99 |
| [4] | 105 |
| [5] | 99 |
| [6] | 105 |
| [7] | 99 |
| [8] | 105 |
| [9] | 104 |
| [10] | 99 |
| [11] | 98 |
| [12] | 104 |
| [13] | 104 |
| [14] | 99 |
| [15] | 104 |
| led_average | 100 |

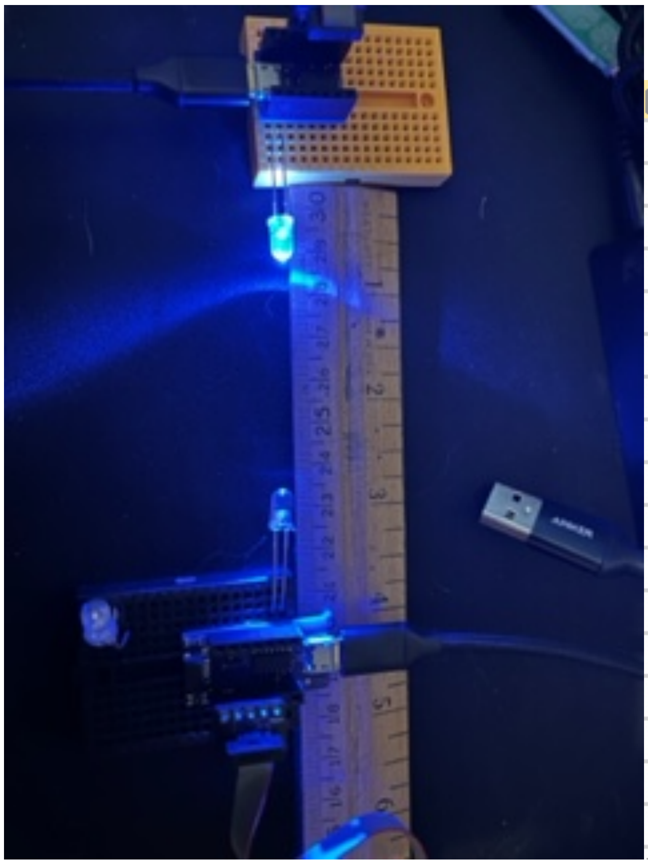
This is a distance of about ~1 inch, with the timeout set to 500 microseconds.

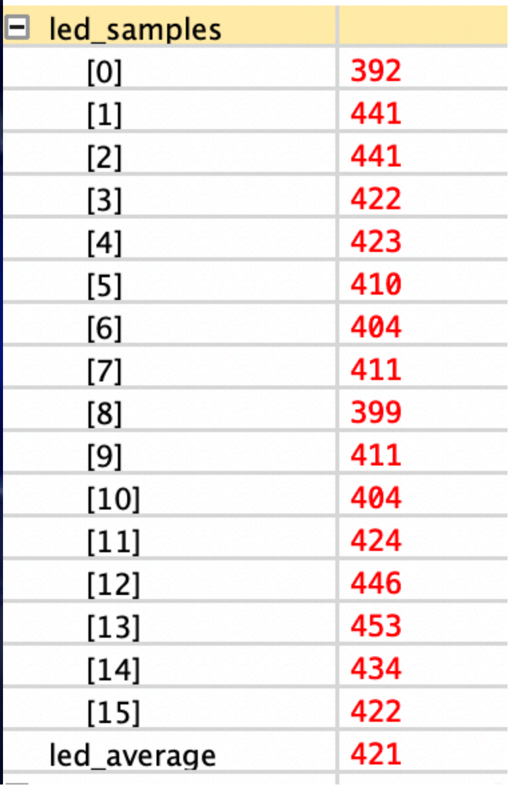




| led_samples | |
|---|---|
| [0] | 260 |
| [1] | 296 |
| [2] | 308 |
| [3] | 309 |
| [4] | 320 |
| [5] | 321 |
| [6] | 310 |
| [7] | 320 |
| [8] | 326 |
| [9] | 315 |
| [10] | 320 |
| [11] | 315 |
| [12] | 316 |
| [13] | 297 |
| [14] | 296 |
| [15] | 310 |
| led_average | 308 |

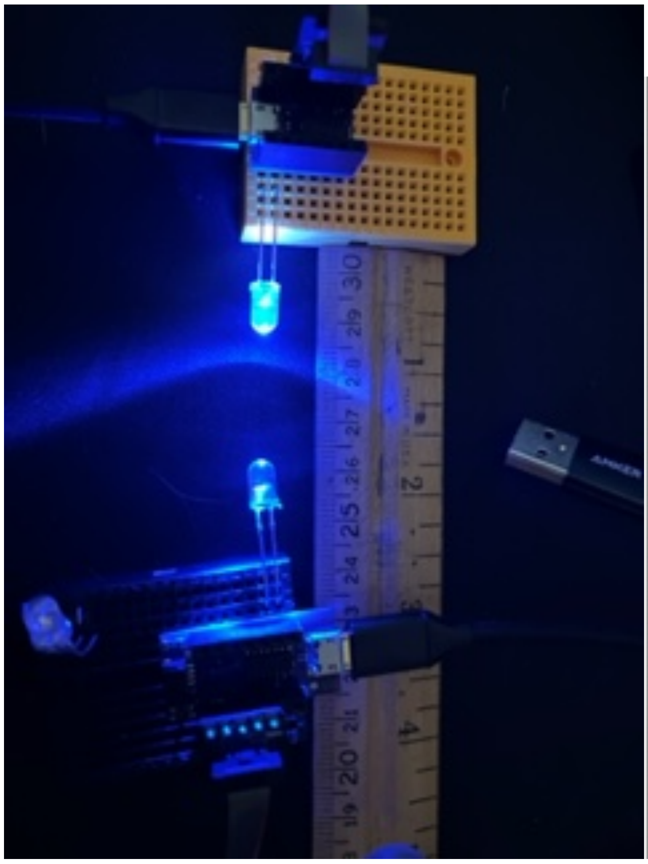
This is at ~2 inches.

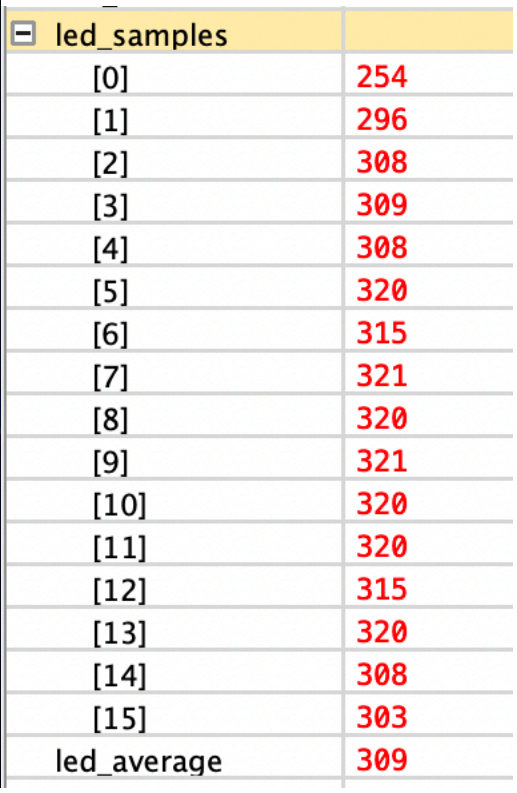




| led_samples | |
|---|---|
| [0] | 392 |
| [1] | 441 |
| [2] | 441 |
| [3] | 422 |
| [4] | 423 |
| [5] | 410 |
| [6] | 404 |
| [7] | 411 |
| [8] | 399 |
| [9] | 411 |
| [10] | 404 |
| [11] | 424 |
| [12] | 446 |
| [13] | 453 |
| [14] | 434 |
| [15] | 422 |
| led_average | 421 |

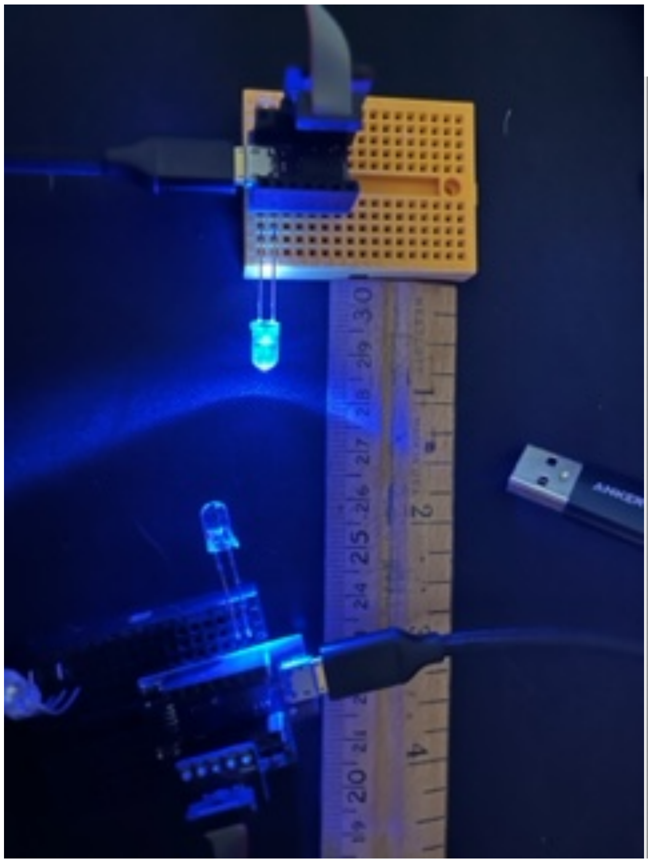
Notice that in the previous cases the LEDs are aimed as directly at each other as I can manage. Here is an example at 1 inch with a slight angle. So, that isn't much different from when I aimed them straight on.

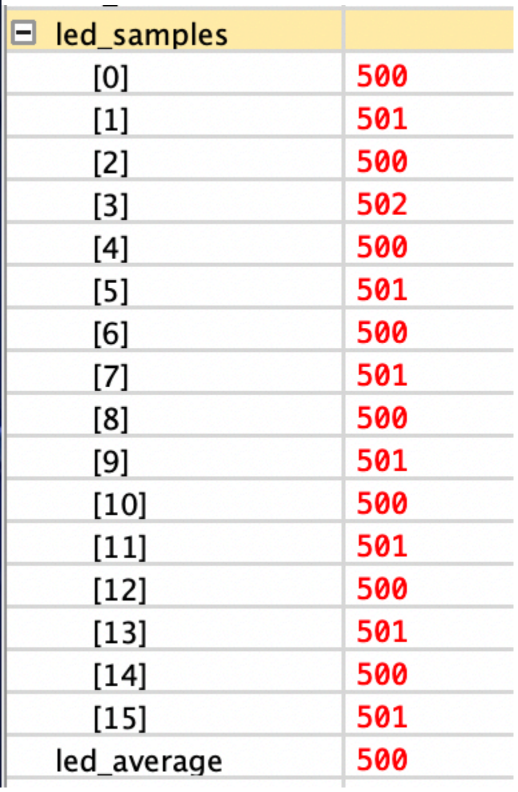




| | |
|---|---|
| led_samples | |
| [0] | 254 |
| [1] | 296 |
| [2] | 308 |
| [3] | 309 |
| [4] | 308 |
| [5] | 320 |
| [6] | 315 |
| [7] | 321 |
| [8] | 320 |
| [9] | 321 |
| [10] | 320 |
| [11] | 320 |
| [12] | 315 |
| [13] | 320 |
| [14] | 308 |
| [15] | 303 |
| led_average | 309 |

And another with a larger angle. Clearly at this point the sensing LED doesn't really see the emitting LED.





| led_samples | |
|---|---|
| [0] | 500 |
| [1] | 501 |
| [2] | 500 |
| [3] | 502 |
| [4] | 500 |
| [5] | 501 |
| [6] | 500 |
| [7] | 501 |
| [8] | 500 |
| [9] | 501 |
| [10] | 500 |
| [11] | 501 |
| [12] | 500 |
| [13] | 501 |
| [14] | 500 |
| [15] | 501 |
| led_average | 500 |

Some thoughts:

The Mitsubishi paper shows scope traces wherein the LED clearly discharges fully in under 100 usecs when the sensing LED is "illuminated by another LED", which is kind of unclear. But I'd imagine that if the LEDs are touching each other that's as illuminated as they are going to get. And under those conditions I was seeing about 100 usecs to discharge. So that's encouraging, I suppose, that I am seeing a similar result.

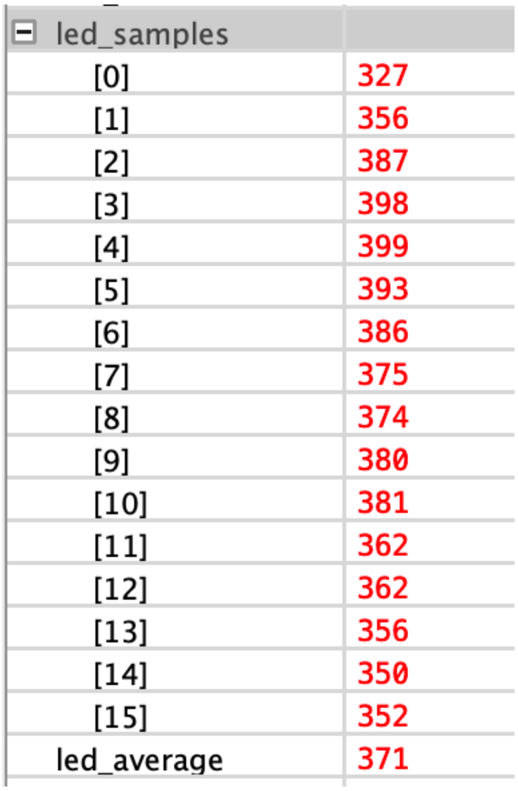
The way my older robots sense light is simply via two instances of a photoresistor in series with a capacitor, one as a left eye, one as a right eye. I charge the capacitor and time how long it takes to discharge. The robot then turns towards whichever side is brighter. If the readings are "about even" then the robot goes forward.

The photocells don't really have a significant 'angle of incidence' issue, but the LEDs seem to. And the sensitivity of LEDs to angle of incidence seems problematic for my purposes. I am curious whether this might work better with surface mount LEDs.

So, I soldered some SMT LEDs onto pins and installed them. I tried the blue, but they were basically unable to detect the other unless they were essentially touching.

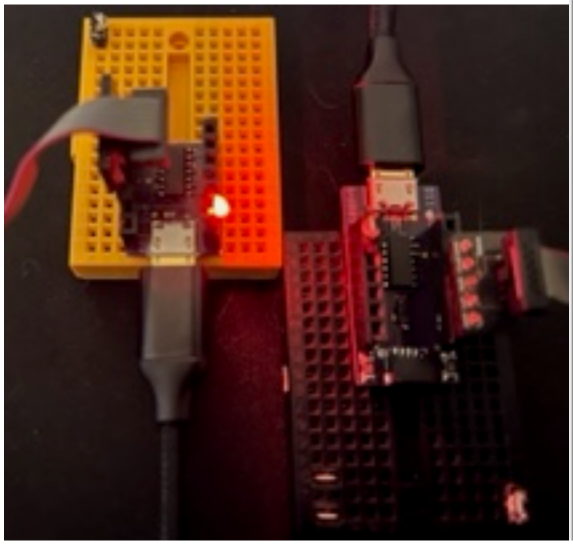
So I tried red. This was, interestingly, more promising. In moderate ambient light at one inch the sensor could see the emitter.

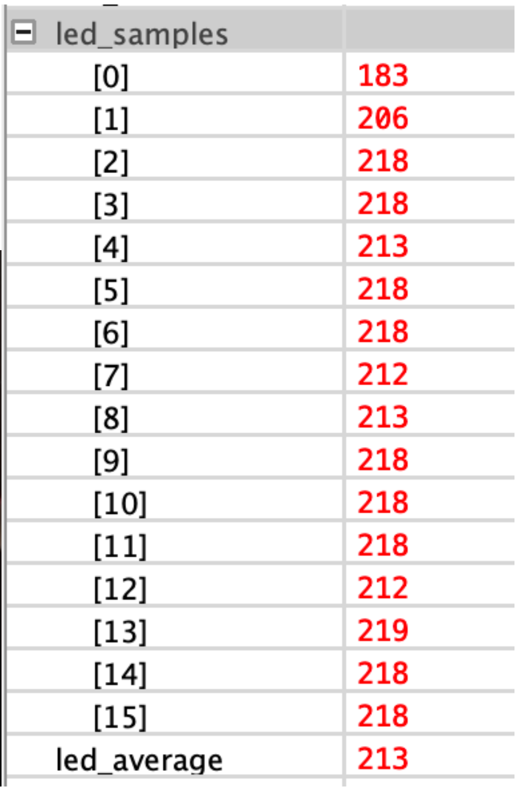
Here is the ambient reading:



| led_samples | |
|---|---|
| [0] | 327 |
| [1] | 356 |
| [2] | 387 |
| [3] | 398 |
| [4] | 399 |
| [5] | 393 |
| [6] | 386 |
| [7] | 375 |
| [8] | 374 |
| [9] | 380 |
| [10] | 381 |
| [11] | 362 |
| [12] | 362 |
| [13] | 356 |
| [14] | 350 |
| [15] | 352 |
| led_average | 371 |

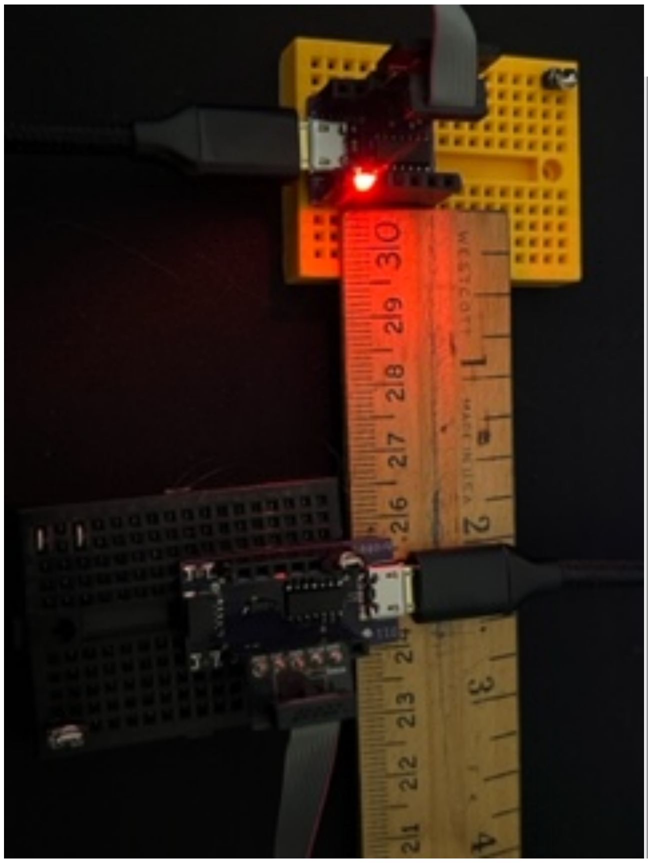
This is at about 1 inch, ambient, aimed straight on.

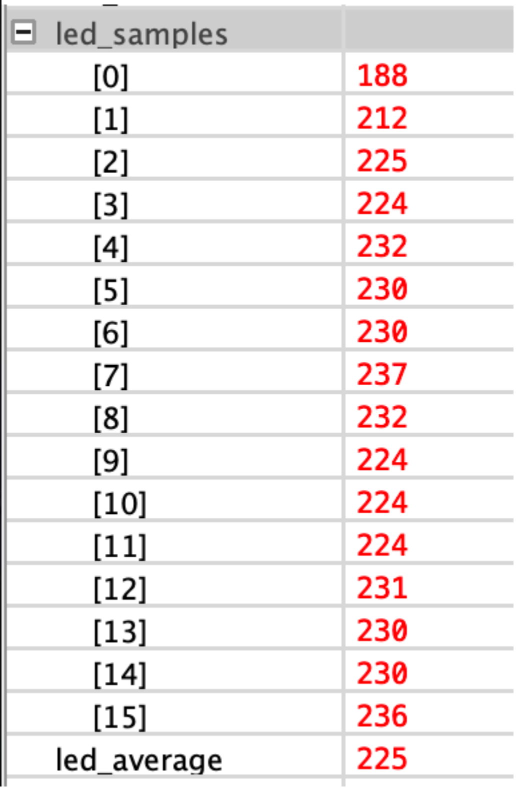




| led_samples | |
|---|---|
| [0] | 183 |
| [1] | 206 |
| [2] | 218 |
| [3] | 218 |
| [4] | 213 |
| [5] | 218 |
| [6] | 218 |
| [7] | 212 |
| [8] | 213 |
| [9] | 218 |
| [10] | 218 |
| [11] | 218 |
| [12] | 212 |
| [13] | 219 |
| [14] | 218 |
| [15] | 218 |
| led_average | 213 |

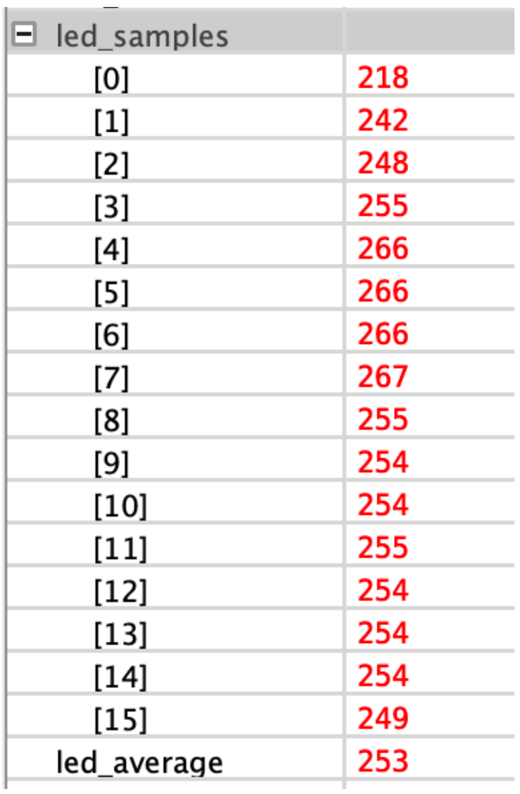
Red SMT LEDs at 2 inches same ambient conditions (i.e., my office).





|  |  |
|---|---|
| ⊟ led_samples | |
| [0] | 188 |
| [1] | 212 |
| [2] | 225 |
| [3] | 224 |
| [4] | 232 |
| [5] | 230 |
| [6] | 230 |
| [7] | 237 |
| [8] | 232 |
| [9] | 224 |
| [10] | 224 |
| [11] | 224 |
| [12] | 231 |
| [13] | 230 |
| [14] | 230 |
| [15] | 236 |
| led_average | 225 |

Same setup, red, 2 inches, but darker ambient (overhead light off, desk lamps on, about 6 feet away). It seems to take longer to discharge because there is less ambient red light ... is my theory.



| ⊟ led_samples | |
|---|---|
| [0] | 218 |
| [1] | 242 |
| [2] | 248 |
| [3] | 255 |
| [4] | 266 |
| [5] | 266 |
| [6] | 266 |
| [7] | 267 |
| [8] | 255 |
| [9] | 254 |
| [10] | 254 |
| [11] | 255 |
| [12] | 254 |
| [13] | 254 |
| [14] | 254 |
| [15] | 249 |
| led_average | 253 |

So at this point I'm thinking "hey, this isn't too bad, really." So of course I got stupid and decided to swap the emitting and sensing LEDs to see if they both gave roughly the same readings. And then things kind of stopped working right. I wasn't able to get back to the original results, even if I put the LEDs back to their original positions.

So, I tried UV LEDs, and white LEDs, and the results were awful. Basically not sensing light. So I decided something must be wrong with the setup. So I tried the red SMT LEDs again, and they were 'working' in the sense that they could see each other. So I tried 5mm red LEDs again, this time from a different manufacturer. And they seemed to work ... really well. And get decent distance (like at least 4 or 5 inches) but they are very directional still.

So have I learned anything useful at all? Well, it's not going to be very easy for robots to detect each other, move into proximity, align their beams, and communicate using this scheme.

**Maybe Do**

Place an MCU on a breadboard and install two 'identical' standard (not RGB) LEDs, each slightly angled to the side (like the vibrobot's eyes). Place the I2C RGB LED strip horizontally so that the LEDs face the MCU's LEDs. Light each position on the LED bar one at a time and take readings from the two MCU LEDs. Determine if the MCU LEDs register approximately equivalent light when illuminated "head on' between then, and more/less light when illuminated to the sides. I.e, can they be used to determine which way to turn?

Will this fail because the LEDs are mainly sensitive to other 'identical' LEDs? Or will it be good enough?. Possibly I need to use plain old LEDs instead of RGBs.

Perform the tests for the LEDS I have and try to collect the data in a meaningful way so I don't have to repeat these experiments again.

**End**