# Automatic Light - Version 2

15th October 2016

This project is continued from my previous project (Automatic Light - Version 1), please read that write up before this one.

## Overview:

In my previous write-up, I discussed the limitations of my automatic light system. The main issues were that:

1. The system only works with about 70% accuracy, due to lack of datapoints
2. I cannot add on new sensors or outputs due to the limited power of my raspberry pi.
3. The system automatically turns the light on even when it’s daytime and the light does not need turning on

I have now rectified these problems.

## Issue 1: Accuracy

The system is only about 70% accurate, which is okay. But can be annoying and can waste electricity if the light comes on when I leave the room. The way that I originally thought to fix this was to put another pressure plate outside the door which would add another data point, however this was not possible as I could not put the plate under the carpet or a rug without it’s position being constantly moved, furthermore it would randomly be trodden on which would not be good for automation.

My solution to this is a laser shining on a light dependent resistor (LDR) in the door frame. When a person walks through the door, the laser will be blocked which in turn will raise the resistance of the LDR and thus blocking the current from flowing. This will give me a datapoint outside of my door which could be used with the pressure plate on the inside of my door to accurately tell me if someone walks in or out.

## Issue 2: Power Limitations

My issue is now power; I need more of it to set up a laser with an LDR. I decided that the easiest way to extend my current system without increasing the pi’s power output is to use a micro-controller, such as an Arduino which would supply its own power.

## Issue 3: Automation deactivated during day

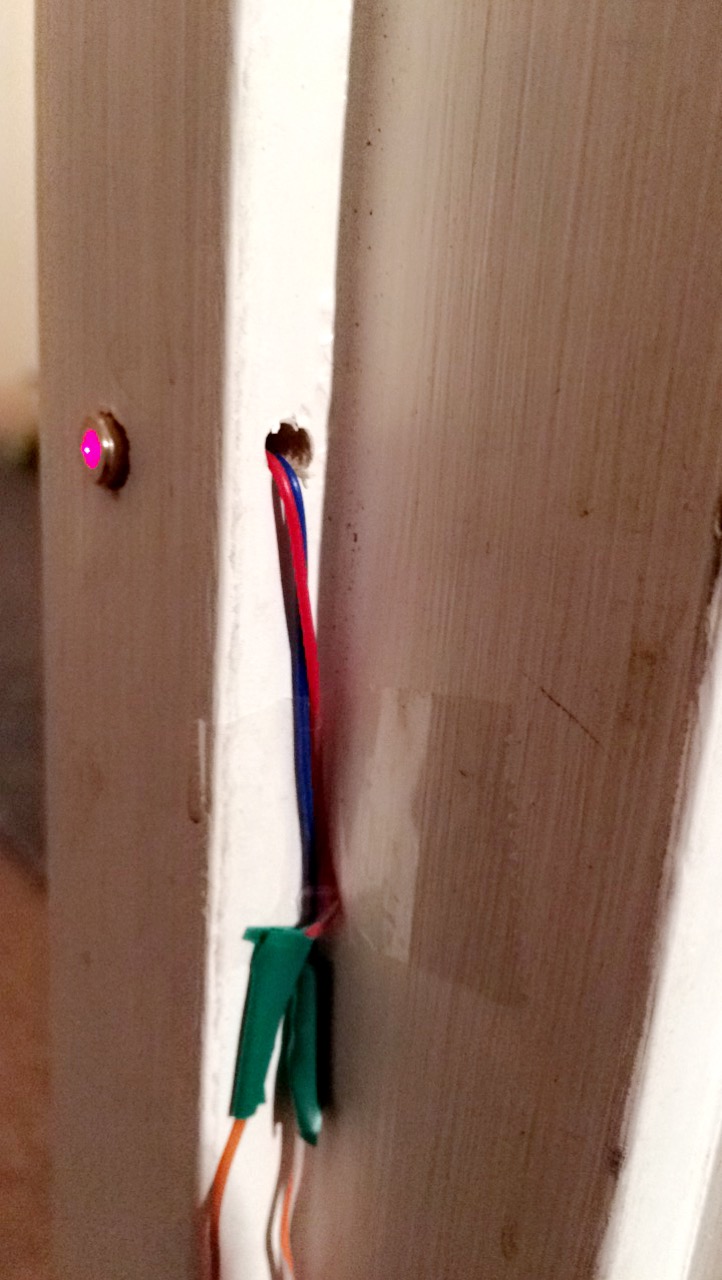
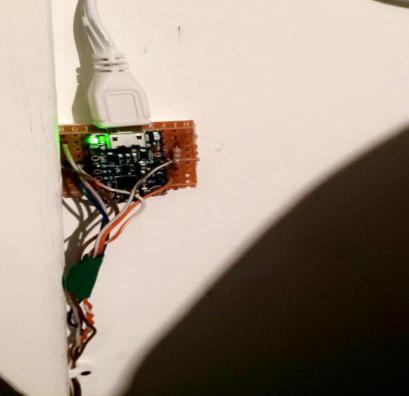
Issue 3 was fixed without me even realizing it due to the way the LDR works. I only added a very small resistor in series with the LDR which in retrospect should have been a larger resistor. However, the unforeseen benefit of this was that when it’s very bright in the house, such as in the morning or afternoon, the laser sensor cannot be tripped because there is enough light hitting the LDR to allow current to flow without the laser shining on the it. This then means that the program that I wrote cannot detect whether someone is walking in or out during times when it’s not appropriate to turn the light on.

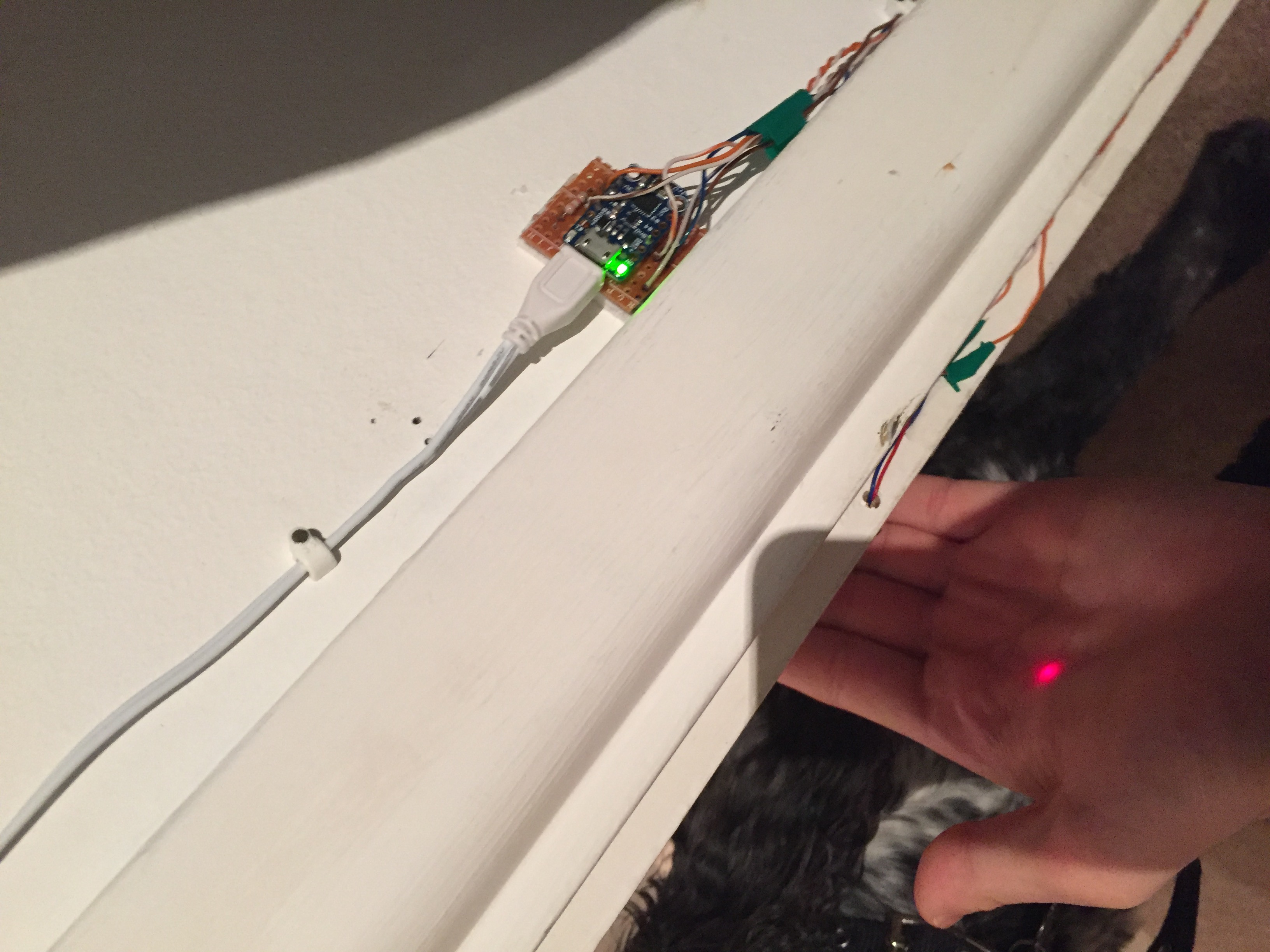
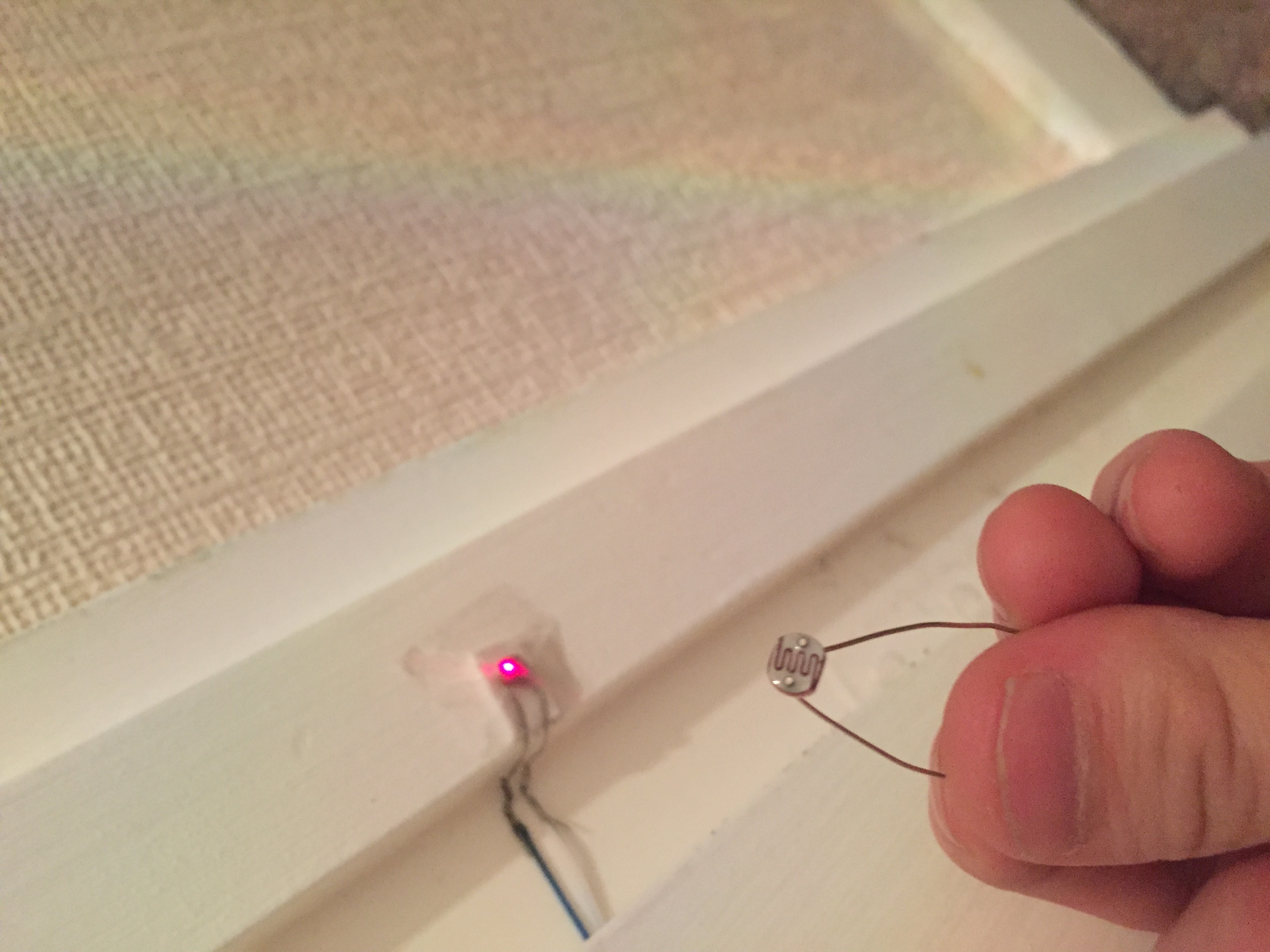
I could have perhaps added another resistor/a larger resistor in series with the LDR to increase the resistance to a level where the laser trip wire works at all times of the day, and then added an LDR facing out of the window to determine whether it’s dark enough for it to be appropriate to turn the light on. However, what I have now works fine and adding another LDR would mean another voltage output from my already dwindling power supply in the Pi

## Solution and Implementation:

I decided to buy an Arduino Trinket, a laser diode and an LDR to create a laser trip “wire” that is fitted into the door-frame outside my bedroom door. This would be used in part with the already present pressure pad on the inside of the door to detect whether someone walks in or out.

I created a simple circuit by soldering the trinket to a prototyping board along with the LDR, Laser Diode and Output Wires (For communicating with the Pi). The Arduino has its own power supply which meant that it was able to reliably provide power to the laser. The laser has been drilled into the door frame so that it is flush and it’s orientation does not change. Here are some pictures of the setup:





Here is the system in use:

<https://www.youtube.com/watch?v=Ha27VcBW-lM>

## Source Code:

Arduino Source Code:

int inputPin = 2;

int outputPin = 0 ;

void setup() {

pinMode(outputPin, OUTPUT);

}

void loop() {

digitalWrite(outputPin, digitalRead(inputPin));

delay(1);

}

Raspberry Pi source code:

[github.com/BenjaminThomas1999/Automatic-Light-Raspberry-Pi/blob/laser-implement/HardwareIO.py](https://github.com/BenjaminThomas1999/Automatic-Light-Raspberry-Pi/blob/laser-implement/HardwareIO.py)

import time

from threading import Timer

import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BCM)

GPIO.setup(24, GPIO.OUT)

def cycle(arr, item):

arr.append(item)

return arr[1:]

all\_inputs = []

class HwInput(object):

def \_\_init\_\_(self, PIN, name):

self.PIN = PIN

self.name = name

GPIO.setup(PIN, GPIO.IN, pull\_up\_down=GPIO.PUD\_DOWN)

all\_inputs.append(self)

self.history = [0, 0]

def state(self):#state() returns the change of input

#True = [0, 1] = stepped on

#False = [1, 0] = stepped off

#None = No change

if not self.history[0] == self.history[1]:

if self.history[1] == 1:

return True

elif self.history[1] == 0:

return False

else:

return None

def update(self):

self.history = cycle(self.history, GPIO.input(self.PIN))

sequence = []

for i in range(2):

sequence.append("")

door\_sensor = HwInput(18, "door\_sensor")

pressure\_plate = HwInput(23, "pressure\_plate")

wall\_pad = HwInput(25, "wall\_pad")

main\_switch = HwInput(4, "main\_switch")

door\_laser = HwInput(17, "door\_laser")

light\_state = True #light is on by default

laser\_state = False #laser is not tripped by default

def coolDownFunction():

global sequence

print("sequence cleared due to cool down")

sequence = ["", ""]

def restartCoolDownTimer():

global cool\_down\_timer

cool\_down\_timer.cancel()

cool\_down\_timer = Timer(8, coolDownFunction)

cool\_down\_timer.start()

cool\_down\_timer = Timer(8, coolDownFunction)

def toggleLight():

global light\_state

light\_state = not light\_state

def ioUpdate():

global light\_state, sequence, cool\_down\_timer

for i in all\_inputs:

i.update()

print(sequence)

if wall\_pad.state() == True:#When wall pad is pressed toggle light

toggleLight()

if main\_switch.state() == True or main\_switch.state() == False:#when light switch changes, toggle light

toggleLight()

if pressure\_plate.state() == True:

sequence = cycle(sequence, "PA")#PA = Plate Activated

restartCoolDownTimer()

elif door\_laser.state() == True:

sequence = cycle(sequence, "LT")#LT = Laser Triggered

restartCoolDownTimer()

if sequence == ["LT", "PA"]:

if light\_state == False:

light\_state = True

#sequence = ["", ""]

elif sequence == ["PA", "LT"]:

if light\_state == True:

light\_state = False

#sequence = ["", ""]

web\_file = open("/var/www/html/output.txt")#output from web client

web\_input = web\_file.read()

if web\_input == "1":

light\_state = True

elif web\_input == "0":

light\_state = False

web\_file.close()

web\_file = open("/var/www/html/output.txt", "w")#clear file

web\_file.close()

web\_output = open("/var/www/html/state.txt", "w")#output to web client

if light\_state == True:

GPIO.output(24, False)

web\_output.write("on")

else:

GPIO.output(24, True)

web\_output.write("off")

web\_output.close()

def ioUpdateLoop():

while True:

ioUpdate()

time.sleep(0.025)

if \_\_name\_\_ == '\_\_main\_\_':

ioUpdateLoop()

## Conclusion:

Overall, I would say that the laser addition has been highly successful. This addition has allowed the system to increase from about 70% accuracy in the last edition, to about 95% accuracy. If I was to improve this, I would replace the red laser that’s fitted in the door to an infra-red laser so that it is not visible to the naked eye this would allow for a more a discrete and aesthetically pleasing system overall.