Lightpainting

```
#from neopixel import *
import time
import json
#import RPi.GPIO as GPIO
import os, fnmatch
"""
#Button-config:
GPIO.setmode(GPIO.BCM)
GPIO.setup(25, GPIO.IN, pull_up_down=GPIO.PUD_UP)

# LED strip configuration:
LED_COUNT        = 144    # Number of LED pixels.
LED_PIN          = 18     # GPIO pin connected to the pixels (18 uses PWM!).
LED_FREQ_HZ      = 800000 # LED signal frequency in hertz (usually 800khz)
LED_DMA          = 5      # DMA channel to use for generating signal (try 5)
LED_BRIGHTNESS = 50       # Set to 0 for darkest and 255 for brightest
LED_INVERT       = False  # True to invert the signal (when using NPN transistor level
LED_CHANNEL      = 0      # set to '1' for GPIOs 13, 19, 41, 45 or 53
LED_STRIP        = ws.WS2811_STRIP_GRB   # Strip type and colour ordering
strip = Adafruit_NeoPixel(LED_COUNT, LED_PIN, LED_FREQ_HZ, LED_DMA, LED_INVERT, LED_BRIG
strip.begin()
"""

def convert_image(image_path):
        with open(image_path,"r") as f:
                new_image = []
                new_image.append(f.readline())
                new_image.append(f.readline())
                dims=f.readline()
                new_image.append(dims)
                file_len=int(dims[:dims.index(" ")])*int(dims[dims.index(" "):])
                new_image.append(f.readline())
                for i in range(int(file_len)):
                        p1 = int(f.readline()[:-1])
                        p2= int(f.readline()[:-1])
                        p3= int(f.readline()[:-1])
                        pixel = [p1,p2,p3]
                        new_image.append(pixel)
                f.close()
        return new_image


def read_image(image_path,number_of_image):
        converted_image = convert_image(image_path)
        converted_image = converted_image[2:]
        image_dims = converted_image[0]
        converted_image = converted_image[1:]
        image_width=int(image_dims[0:image_dims.index(" ")])
        converted_image = converted_image[1:]
        inverted_list = image_width*[144*[""]]
        print(inverted_list[0][0])
        list_iteration_counter = 0
        print(inverted_list[0][0])
        for i in range(image_width):
                print(inverted_list[0][0])
                for j in range(144):
                        print(inverted_list[143][143])
                        inverted_list[i][j] = converted_image[list_iteration_counter]
                        list_iteration_counter+=1
        with open("images/new_format/transition"+str(number_of_image)+".json","w") as f:
                json.dump(inverted_list,f)
                f.close()
        del inverted_list
        width_of_each_image.append(image_width)
        image_as_json_list.append("images/new_format/transition"+str(number_of_image)+".

def show_picture(json_path,image_width):
        print(json_path,image_width)
        with open(json_path,"r") as f:
                pixels = json.load(f)
                f.close()
        for _ in range(len(pixels)):
                for __ in range(len(pixels[_])):
                        pixel_temp = pixels[_][__]
                        #print(pixel_temp[0],pixel_temp[1],pixel_temp[2])
                        #strip.setPixelColorRGB(__,pixel_temp[0],pixel_temp[1],pixel_tem
                #strip.show()
                #time.sleep(0.08)

"""
def clear_strip():
        for i in range(144):
                strip.setPixelColorRGB(i,0,0,0)
"""

################################################################################
#Init. vars:
image_old_list = []
image_as_json_list = []
width_of_each_image = []
button_counter=0
intermed_counter = 0


#show program start (not ready):
"""
for _ in range(10):
         strip.setPixelColorRGB(_,200,0,0)
"""
list_all = os.listdir('images')
pattern = "*.ppm"
for entry in list_all:
        if fnmatch.fnmatch(entry, pattern):
                 image_old_list.append(entry)
for _ in image_old_list:
        read_image("images/"+_,intermed_counter)
        intermed_counter+=1

#show that ready:
"""
for _ in range(10):
        strip.setPixelColorRGB(_,0,200,0)
"""
"""
while True:
        input_state = GPIO.input(25)
        if input_state == False:
                print("Button pressed")
                show_picture(image_as_json_list[button_counter],width_of_each_image[butt
                button_counter+=1
        time.sleep(0.2)
        if counter == len(image_new_list):
                counter = 0
"""
show_picture(image_as_json_list[0],width_of_each_image[0])
```