# PHYS210 Electronics Lesson

Coco Zhang
ProtoScan 2D: A Naïve Camera

October 2025

## 1 Project Overview

Are you into photography? Do you have a camera? Have you ever delved into how they work? This lesson helps you walk through how to build a simple light-detection blackbox camera that scans a 2-Dimensional pixel image. By building an I to V converter circuit with photodiodes op-amp, we generate voltage signals about the brightness. This is a sensor unit, which produces signals readable by Arduino. Our blackbox camera will contain an array of sensor units that's moved along a perpendicular axis with a stepper motor. Together, it creates a 2-D scan of a binary pixel image, i.e. a image with two degree of light or dark.

### 1.1 Required Components

- Zaber X-LMH200A Stepper Motor
- Arduino Uno
- Photodiodes
- 10M$\omega$ resistors
- LF411 Op-amps
- Breadboard power supply 5V/12V
- Breadboard and wires
- Cardboard boxes, 3D printed plates, and zip ties (mechanical support: any materials that can create a box and attach components together)

We need the same number of resistor, op-amp, and photodiodes to form units of light detection. I used 6 for each to build 6 units. You can adjust based on your scale and needs!

## 1.2 Major Components Description

Photodiodes can respond to visible light incident on it. When there's no incoming light, there's no current flowing through the photodiode. However, it generates a nonzero reverse current when the light level is nonzero. The current size is proportional to the brightness it senses. We can use the photodiode in an I to V converter circuit to detect the brightness from a specific pixel in the binary image. When using a photodiode, we have to be careful to plug the photodiode in the opposite way to our expected direction of current flow.

### 1.2.1 Photodiodes

Figure 1: This is a Si photodiode

### 1.2.2 LF411 Op-amps

An Operational Amplifier (Op-Amp) is a central component of the I to V converter as it amplifies the small amount of current generated by the photodiode into a measurable output current. Each unit in our light detector array will use an op-amp to output one voltage signal, which will be recorded.


Figure 2: This is an LF411 Operational Amplifier (Op-Amp)

An Op-amp has 8 pins of different functions and has to be connected into the breadboard circuit very carefully. You should always identify the orientation of the op-amp and compare it to the configuration chart to correctly connect it with other components, and always refer to the following chart to double-check before powering the op-amp.

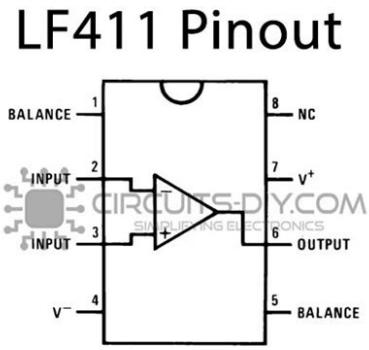


Figure 3: The chart shows the pin configuration and functions for all pins on the Op-amp

### 1.2.3 Zaber X-LMH200A Stepper Motor

A stepper motor converts electrical signals to precise rotational motion that can move an attached item with precision in position and speed. Zaber X-LMH200A Motor is controlled by a package via python programming, which we will learn more about in the code explanation section. To stably attach the detector unit to the motor, I used a 3D printed board that is screw-driven into the platform. You have to be careful with mechanical mounting of the components with a stepper motor. A rule of thumb when programming motion with Zaber motor is to ensure your desired range of movement is within its performance range. Otherwise, the python package shoots an error message.

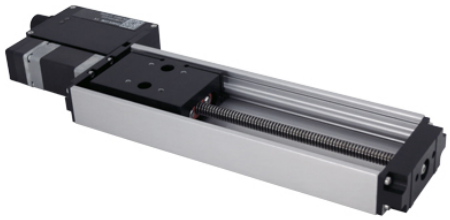

Figure 4: This is a Zaber X-LMH200A Stepper Motor with a controller on the left and a long stage for controlled travelling.

# 2 Circuit

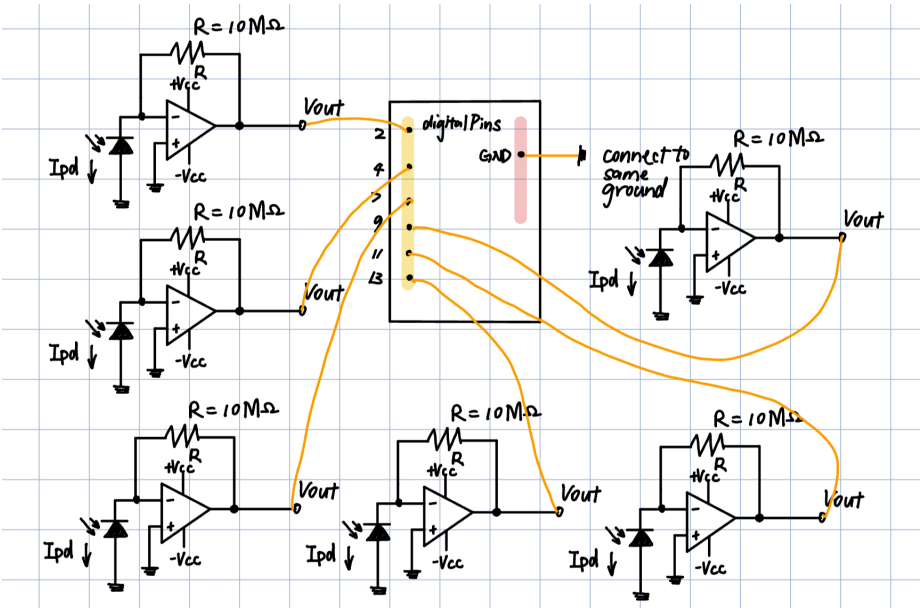


Figure 5: This is the breadboard circuit I used, along with what pins I connected it to on the Arduino. It includes 6 I to V Converter circuits, and their $V_{out}$s are all connected to DigitalPins on the Arduino.

A special note here on the circuit construction: I tested with resistors of different resistances in order to find the best-fitting one that accurately captures both type of brightness. Dark (Digital pin read = LOW, 0) is when the photodiode is blocked by a sheet of cardboard. Bright (Digital pin read = HIGH, 1) is when the photodiode is exposed to the room light without shielding. Your environment may require a different resistor value, which is worth experimenting with before fully assembling your detector.

# 3 Code

## 3.1 Arduino Code

```
int pins[6] = {13, 11, 9, 7, 4, 2}; // Create a list of digital pins in
    use
int values[6];
int inByte = 0;      // Initialize serial byte

void setup() {
  for (int i = 0; i<6; i++) {
```

```
    pinMode(pins[i], INPUT);
  }
  Serial.begin(115200); // Initialize for the Python code to read
}

void loop() { // if we get a valid byte, read analog ins:
  if (Serial.available() > 0) {
    for (int i = 0; i < 6; i++) {
      values[i] = digitalRead(pins[i]);
      }
      Serial.print("["); // Print results as a string
      for (int i = 0; i < 6; i++) {
        Serial.print(values[i]);
        if (i < 5) {
          Serial.print(",");
          }
        }
      Serial.println("]");
      delay(1000);
  }
}
```

The Arduino code listed above sets up all digital pins in use to be in INPUT mode that reads the voltage signal in binary upon request. In the loop, Arduino reads every voltage signal in the array and outputs them to the serial, which writes into python. The voltage signals have to be output as a single object, and we will print them into a string expression of a list, which will be decoded in python.

## 3.2 Python Code

Our python code refers to and updates the code given by from Double Slit Measurement experiment in PHYS210. It links controls the Zaber motor, links to Arduino serial, and visualises the data we collected. I'll show the code I have edited that are different from the given DoubleSlitMeas.ino file.

### 3.2.1 Performance Measurement function

This function performs a measurement from the specified stop position to the stop position in steps of the chosen step size. It first intializes the list of positions to stop at, and a 2D array data structure to save data in. Then looping over the positions, it moves stage to position, sends message for Arduino to take measurement, and reads data. It returns both the list of position, and 2D array of data collected.

```
    length_unit = chooseUnits(units) #choose the unit length for code
    # Generate list of positions
```

```
    positions = np.arange(start,stop,stepSize,int) #list of stopping
        positions, set start, stop, stepSize
    data = np.zeros((len(positions)+1,len(positions)+1)) # Initiliases a
        2D array of 0s for the full scan
    counter = 0

    for i in tqdm(range(len(positions))): #progress bar
        axis.move_absolute(int(positions[i]),length_unit) # Move to
            position
        axis.wait_until_idle() # Waits until Zaber finishes moving to
            avoid time-out
        time.sleep(wait) # Wait time for vibration to settle, can be
            adjusted, defaults to 0.5 s
        arduino.reset_input_buffer()
        arduino.write('s'.encode()) # Tell ardiuno to perform a
            measurement
        time.sleep(wait)
        line = arduino.readline().decode().strip() # high = 1, low = 0
        if line:
            try:
                values = ast.literal_eval(line) # Decodes string output
                    from Arduino into python list
                print(f"line{i}: {values}")
                data[i, :] = values
            except Exception as e:
                print(f"Error {e}. Raw line: {line}")
        time.sleep(pause) # Wait time between measurements,
            theoretically unnecessary, defaults to 0.5 s
    arduino.close()
    print("Serial port closed.") # Clean serial port and prepare for the
        next measurement
    return data,positions
```

## 3.3 Set-up the Desired Measurement Range

When setting up the measurement range to input into the function above, you can refer to this list of inputs I used for each variable. Be careful to check if your measurments are possible for the motion range of the Zaber stepper motor.

```
unit = "cm" # Specify units as centimeters "cm", millimeters "mm", or
    micrometers "um"
start = 6 # Start position in specified units
step = 2.8 # step size in specified units
datapoints = 6 # Desired number of datapoints

stop = (datapoints-1)*step+start # End position in specified units

pause = 1 # Optional pause between measurements, in seconds
```

```
wait = 1 # Optional additional wait time after moving stage but before
    performing measurement, in seconds
```

### 3.4 Data visualization

To visualise the brightness data represented in a 2D array, we can use matplotlib package to generate heat maps with the code below.

```
fig, ax = plt.subplots()
im = ax.imshow(data, cmap="binary_r") # binary_r coloring shows 0 =
    black and 1 = white
```

## 4 Project set-up and limitations

My sample blackbox camera is being held together by a cardboard box, which provides shielding to prevent excess light from getting through and supports the structure of the zaber motor. It also provides a slot facing the light detector, which allows us to insert different carved out images to be scanned.

If you're curious to explore, there are a few things you can further experiment with:

1. Arduino Uno has 6 digital pins, which allow accurate reading of voltage signals. As we know the current generated by a photodiode is proportional to the incoming light, we can create a more precise formula to represent a gradient of brightness instead of the black-and-white representation we have right now.

2. You can consider representing data more creatively in real-time, for example, through daisy-chaining a group of LEDs.

## 5 Appendix

### 5.1 Exemplar Photographs

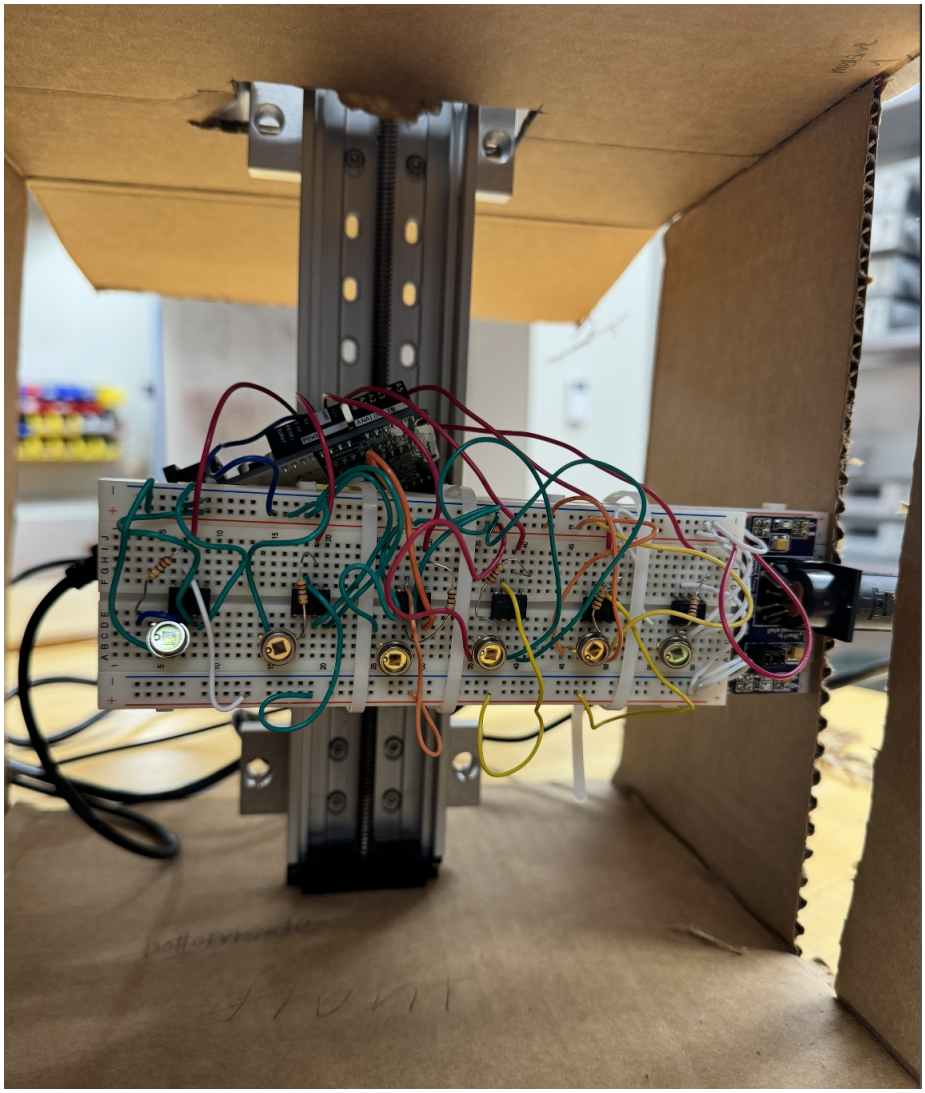

Figure 6: My breadboard circuit (brightness detection) tied to Arduino and attached to the Zaber Motor

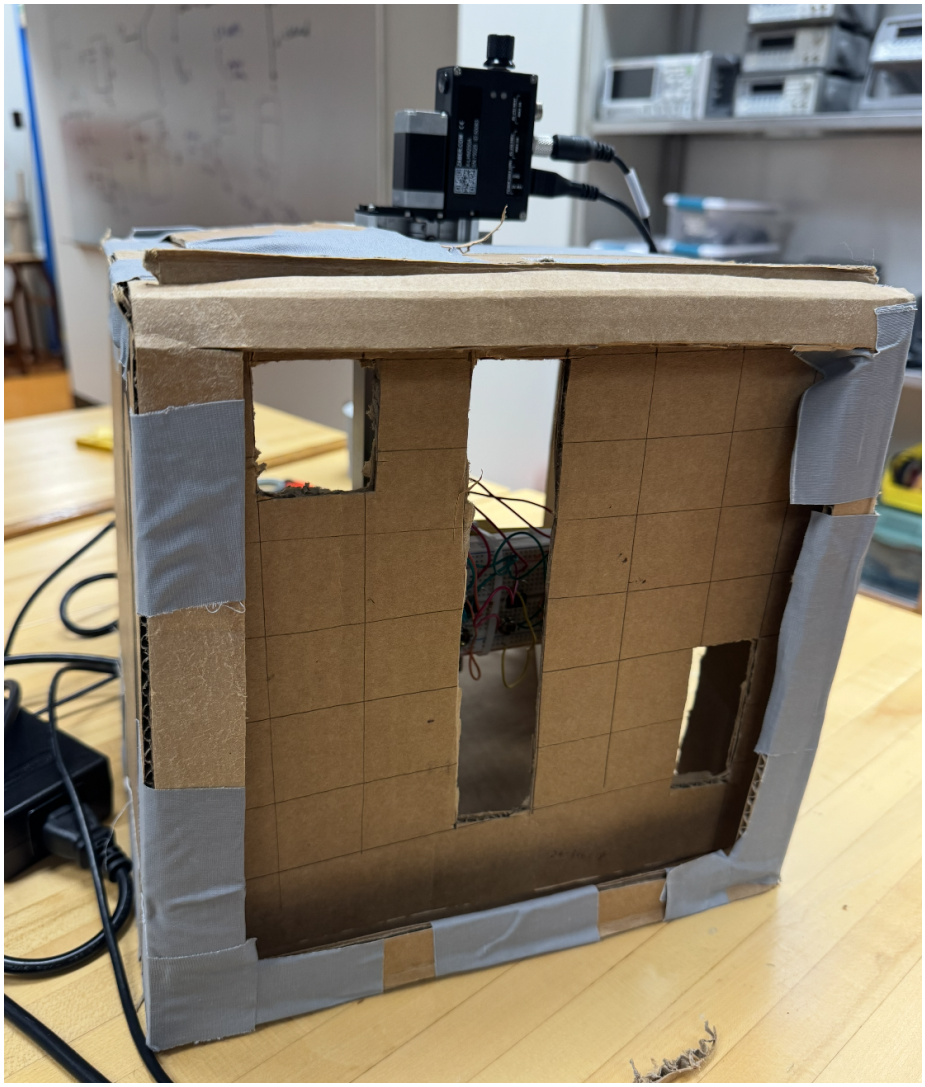

Figure 7: The blackbox camera with the image sheet slotted into the box (front)

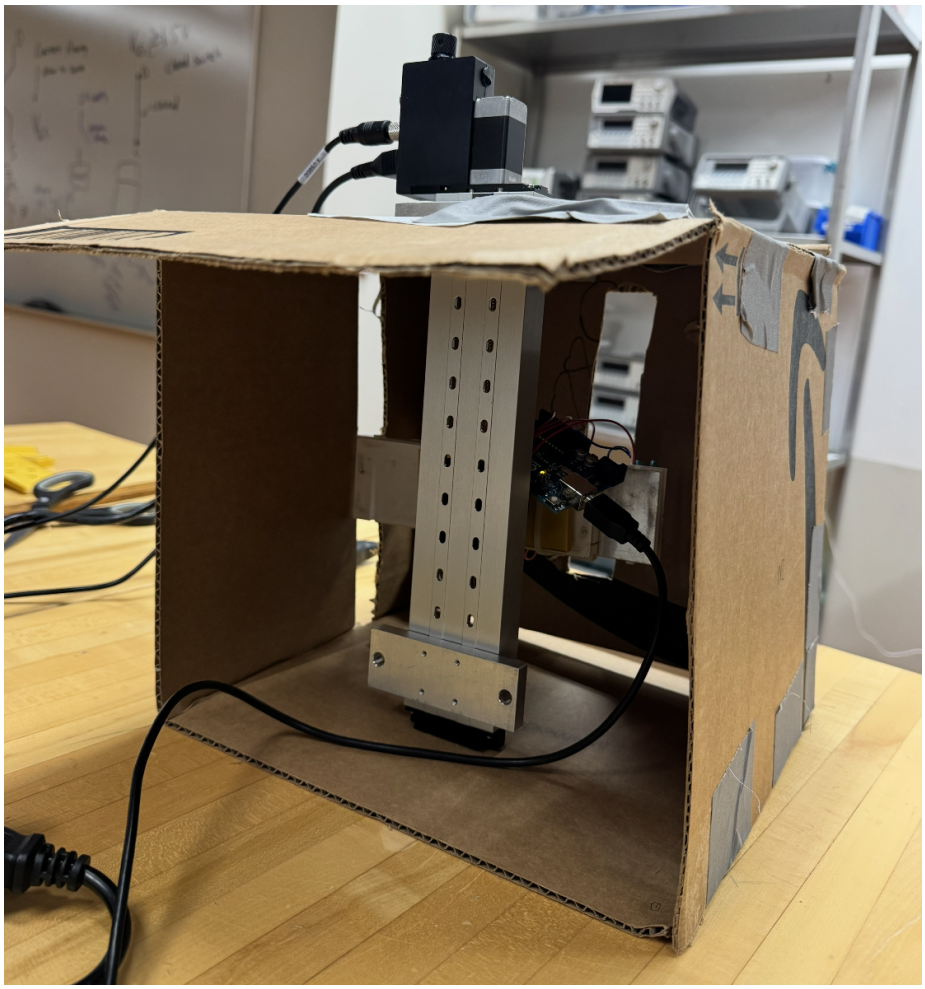

Figure 8: The blackbox camera with the image sheet slotted into the box (back)