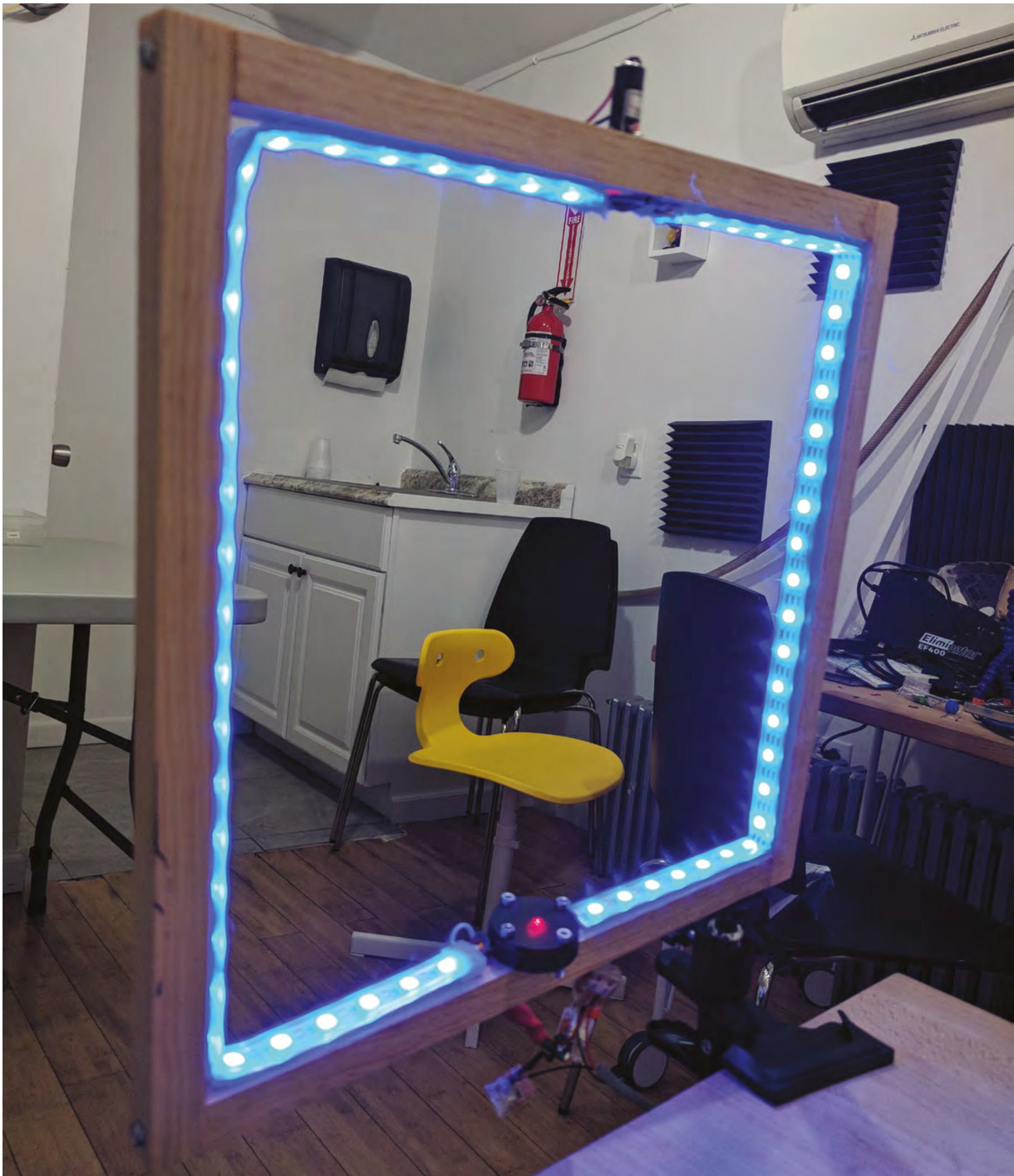
Laser tripped LED drone racing rings

Build laser trip-wires and use them to create a futuristic indoor drone racing game!

Mike Fischthal

WhatWillYouMake.com

Mike Fischthal is the CEO and Founder of the Pixel Academy, a technology and STEM-focused provider of after-school and camp programs in New York City



Right A completed build of a gate, ready for racing

In this tutorial we'll make an Arduino powered laser 'trip-wire' that we can use to build an indoor drone racing game. The end product will be a series of glowing LED gates that can each detect when a drone flies through it (and breaks the laser beam). A gate will light up and when the drone passes through, we will record the success and light up the next gate.

HERE'S HOW OUR PROJECT IS GOING TO WORK:

1. We'll use an Arduino to tell a gate to light up its LEDs.

2. After a short (500 ms) delay that gives the LEDs time to reach full brightness, we'll measure the voltage produced by a tiny solar cell. This will be our baseline voltage that accounts for the ambient room light, the LED lights, and a red laser beam targeted at it.

3. The laser beam is the biggest single factor contributing to the solar cell's voltage, so we can confidently record a beam break when the cell's voltage drops a certain amount below the baseline.

4. An optional tilt sensor will be triggered if a drone hits the gate walls instead of passing cleanly through. Acting as a simple switch, the tilt switch, when triggered, will short the power line with the solar cell output and send a measurable voltage spike over the baseline.

5. When one of these conditions is met, we'll send a signal to that gate to perform an animation or colour change.

6. Finally, a new gate is selected and the process repeats.

LET'S TAKE A LOOK AT THE INDIVIDUAL COMPONENTS OF OUR CIRCUIT:

1. The laser beam: needs 5 V power.

2. The solar cell: will need a reference ground line and a Vout line that the Arduino will measure.

3. A mechanical tilt sensor and resistor that we'll splice between the 5 V power and the Vout.

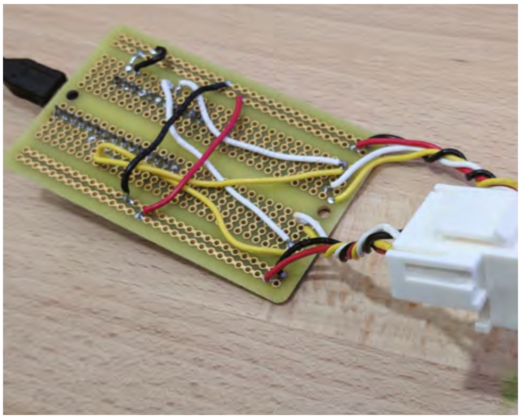
4. The RGB LEDS: need 5 V power, GND, and an analogue data input to control the colours.

We can see from these components that we'll need four seperate lines to each gate. The control box, which will house the Arduino, will therefore use four pins: 5 V, GND, Analog In (Ain), and PWM out (~Dout). The power and ground can be shared, but each gate will require a unique Ain and ~Dout.

Drone racing isn't a tabletop game, so these gates are going to need to be far apart from each other and far from the control box. To avoid having to deal with four separate wires running along the floor, I decided to use a basic telephone cable, which conveniently has four wires bundled together. Phone cables are cheap, flexible, and come in many lengths.

Below is the code used to control the two gates. These addressable LEDs work with Adafruit's NeoPixel library, which you can add to your Arduino IDE by going to Manage Libraries and searching for NeoPixel.

```
#include <Adafruit_NeoPixel.h>
#include <math.h>

#define LED_COUNT 60
//The number of total LEDs in each gate

int currentRing = 0;
// Tracks the active gate

int numRings = 2;
// The total number of gates

int rings_output[] = {5,6};
//Arduino pins that send data to the LEDs

int rings_input[] = {A0,A1};
//Arduino pins that read the voltage from a gate's
solar cell
```



Left Using phone wires has the added benefit of allowing for easy swapping of gates at the control box by using phone jacks as the connection point

YOU'LL NEED

- **Addressable RGB LED light strip** (e.g. WS2812) sparkfun.com/products/12027
- **An Arduino and computer** with Arduino IDE
- **4 × M3 30 mm screws** hsmag.cc/dZEPCI
- **4 × M3 nuts** hsmag.cc/qdbiyx
- **Square pine moulding** (at least 6 ft / 1.8 m) hsmag.cc/NIWKwb
- **Red laser** (1 per gate) ($6 ea) adafruit.com/product/1054
- **Miniature solar cell** (1 per gate) ($1.50 ea) sparkfun.com/products/9541
- **Solid core colour wires**
- **Drill, soldering iron** and various common equipment
- **Wood glue and wood screws**
- **Hot glue gun**

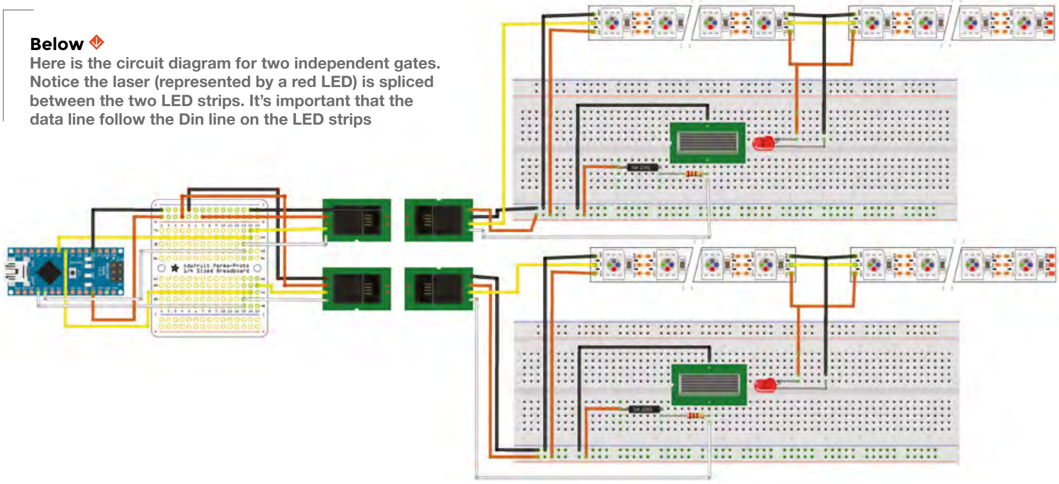
OPTIONAL

- **Mechanical tilt sensor** (1 per gate) ($2 ea) hsmag.cc/XFgUHZ
- **Any resistor** (1 per gate)
- **RJ11 Keystone Jack** (1 per gate) ($0.67 each at Monoprice)
- **RJ11 25 foot phone cable** (1 per gate) ($1.17 each at Monoprice)
- **Square clamp**
- **¼ inch wood inserts** (3 per gate) ($14.03 for 100) hsmag.cc/cemjoV

```
int sensorState = 0, lastState=0;
int onlyShowActive = 1;
float Vstart[] = {0.0,0.0};
//Saves our baseline voltage recordings

float margin_of_error = .05;
// The threshold of voltage drop that triggers a
'hit'

int firstTime = 1;

Adafruit_NeoPixel led1 = Adafruit_NeoPixel(LED_
COUNT, rings_output[0], NEO_GRB + NEO_KHZ800);
Adafruit_NeoPixel led2 = Adafruit_NeoPixel(LED_
COUNT, rings_output[1], NEO_GRB + NEO_KHZ800);

Adafruit_NeoPixel gate_leds[2] = {
  led1,
  led2
};

//Sets a specific gate to all of one color
void setAllToColor(int gate, int r, int g, int b)
{
  for (int j=0; j<LED_COUNT; j++)
        {
          gate_leds[gate].setPixelColor(j, gate_
leds[gate].Color(r,g,b));
        }
  gate_leds[gate].show();
}

//Turns off all of the LEDs in a specific gate
void clearLEDs(int j)
{
   setAllToColor(j, 0,0,0);
}

void setup() {

  pinMode(5, OUTPUT);
  pinMode(A0, INPUT);
  pinMode(6, OUTPUT);
  pinMode(A1, INPUT);
  for (int i=0; i < 2; i++) {
    gate_leds[i].begin();
    //Call this to start up the LED strip.
    setAllToColor(i, 0,0,255);
    //Turns all gates on to make sure they are
working
    gate_leds[i].show();
    //The LEDs don't acrtually light up until you
show() them
  }
  delay(500);
  //Pause for LEDs to reach full brightness before
taking the baseline reading
}

//Sets the new baseline for a specific gate
void resetVStart(int i) {
    float voltage= analogRead(rings_input[i]) *
(5.0 / 1023.0);
    Vstart[i] = voltage;
}

void loop() {

 if (firstTime) {
    for (int i=0; i < numRings; i++) {
      resetVStart(i);
    }
```

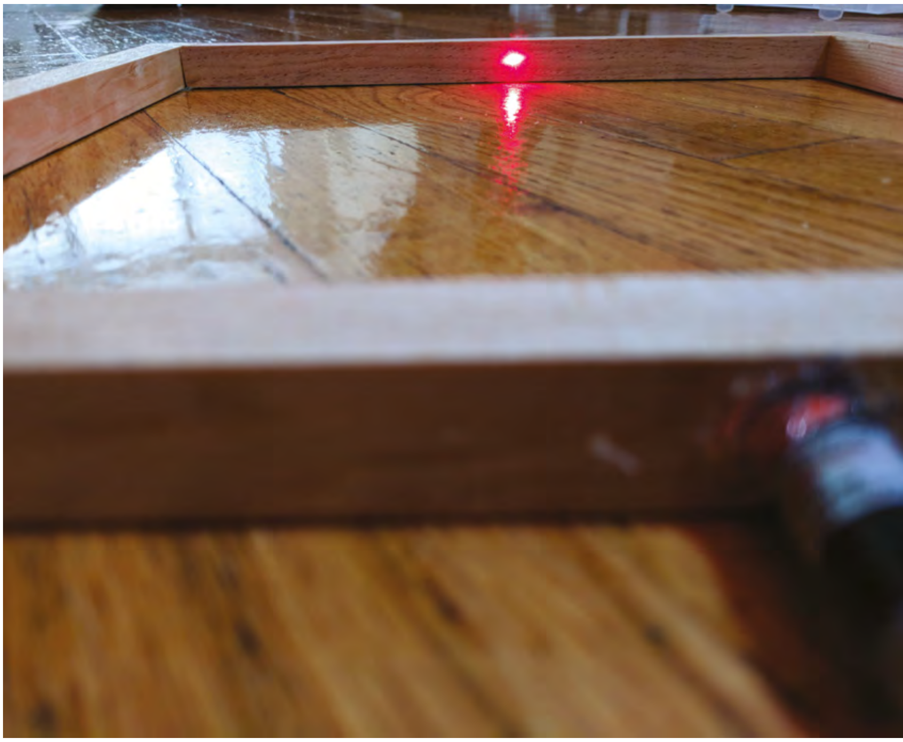
DIGITAL AND PWM

In order to output an analogue signal, the Arduino uses a technique called pulse width modulation (abbreviated as PWM and indicated on the Arduino as an asterisk or a tilde) that rapidly modulates the digital signal to fake an analogue output.



Below
Here is the circuit diagram for two independent gates. Notice the laser (represented by a red LED) is spliced between the two LED strips. It's important that the data line follow the Din line on the LED strips

```
    firstTime = 0;
  }
  //if this is the first loop, record the baseline
voltage

    for (int i=0; i < numRings; i++) {
      float voltage= analogRead(rings_input[i]) *
(5.0 / 1023.0);
      //Read the current voltage input from the
sensor
      if (currentRing==i) {

        //If this is the active ring check if the
current reading indicates a beam break
        if (voltage < (Vstart[i] - margin_of_
error)) {

                      //The beam is broken, flash
green to indicate success
            setAllToColor(i, 0,255,0);
            gate_leds[i].show();
            delay(250);
            setAllToColor(i, 0,0,0);
            gate_leds[i].show();
            delay(250);
            setAllToColor(i, 0,255,0);
            gate_leds[i].show();
            delay(250);
            setAllToColor(i, 0,0,0);
            gate_leds[i].show();
            delay(250);
            setAllToColor(i, 0,255,0);
            gate_leds[i].show();
            delay(250);
            setAllToColor(i, 0,0,0);
            gate_leds[i].show();
            delay(250);
            gate_leds[i].show();

                       currentRing++;
            //Cycle to next gate in the series

            if (currentRing>=numRings)
             {
              currentRing = 0;
             }
             //If that was the last gate, start
over

            setAllToColor(currentRing, 0,0,255);
            //Turn the new gate blue

            delay(500);
```



```
            //Pause again before taking the new
baseline reading
        } else {
          //If the beam is unbroken, keep the same
color
          setAllToColor(i, 0,0,255);
        }
      } else {
        //If the gate is not an active one, keep
it OFF
        setAllToColor(i, 0,0,0);
      }
      gate_leds[i].show();
      //Show the LEDs
  }

}
```

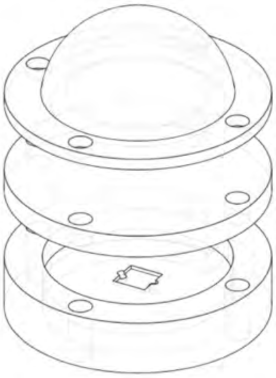
We want to ensure that the drone breaks the laser beam when passing through the gate, even if it doesn't go through directly in the centre. So we're going to make the inside width slightly smaller than double the width of the drone. Even a near miss of either wall will have part of the drone hitting the laser beam.

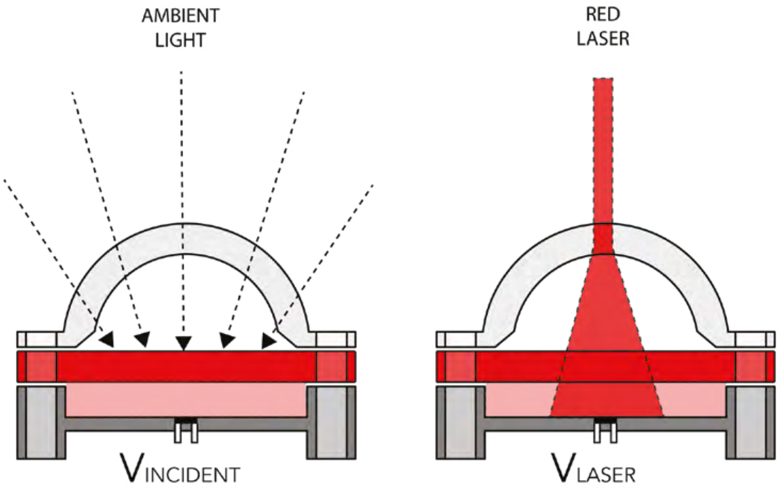
The drone I'm using is a regular indoor drone that cost under $100 and has a wingspan of about 7″ (18 cm), which is about average. An interior width of 11″ (28 cm) will ensure the drone hits the beam and will not make the gate too easy to fly through.

Using the wood moulding, cut two each of 11″ (28 cm) and 12.25″ (31 cm). →

Above
Once your beam is lined up, hot-glue it in place so it stays lined up

Below
Why just glue something when you can 3D-print a custom holder?





Above Adding a diffuser allows more margin for misalignment of the laser, and adding red acrylic cuts down on the ambient light

QUICK TIP

When screwing together thin wood like this, you should always drill a pilot hole first to prevent splitting.

Use a right-angle clamp and wood glue to build the square. If you don't have a right-angle clamp (you should get one), just be careful about alignment. If the top and bottom aren't parallel, it will be a little trickier to get the laser and sensor to line up.

Once the glue dried, I used some wood screws to secure the joints. Knowing that drones may be flying full speed into the gates, I wanted to make them as strong as possible.

Now that we have our (hopefully) perfect square, we're going to add the laser. Find the centre of one of the sides of the square and mark it. We're going to make a small through-hole and a larger half-hole so the laser emitter doesn't go completely through the wood, only the laser beam does. If we make the hole for the beam aperture ¼" (0.6 cm), that will allow us to still get to the focus adjustment (that slot on the front of the emitter) with a screwdriver. Drill the ¼" hole all the way through the wood, keeping the drill as vertical as possible. The laser emitter has a diameter of about 7/16" (1.1 cm), so we're going to drill a hole slightly smaller than that and then sand the inside until the emitter fits perfectly. Use a locking collar on your bit to drill about half-way through the wood, using the smaller hole as a pilot hole. If you don't have a collar for your bit, mark the desired depth on it with tape and just go slow. Use a Dremel or any other sanding tool to slowly widen the hole until the laser emitter fits inside.

> **"You can test the trip wire by connecting the laser to a 5 V power source and reading the voltage off the solar cell"**

ALIGN THE LASER

Power up your laser with any 5 V source and check the alignment. You want to hit as close to centre on the opposite side of the square as possible. If you're way off the mark, you can widen the hole further to give your emitter more of an angle. Once the beam is where you want it, use hot glue to lock it in place.

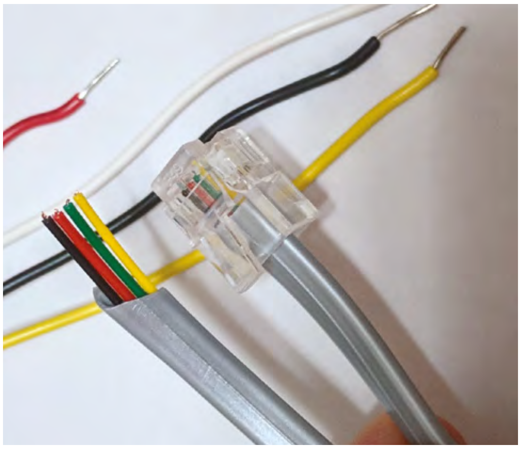
Now we need to install the solar cell. Before you attach it to the gate, you should solder a couple of wires onto the leads. There is a small metal strip on one side of the solar cell that indicates the positive lead. The easiest thing to do now is line the solar cell up so it's hit directly by the laser beam, create a little bed of hot glue, and hold the sensor perfectly in place until it cools and sets. You could do that, but because the sensor is so small I decided to 3D-print a shell for it, and make some functional improvements. The shell has two holes in the centre for the solar cell's leads to make soldering to them, and attaching to the wooden gate, much simpler. Instead of stopping there, I also added an optional frosted diffusion dome that can fit on top of the sensor and a laser-cut red acrylic circle (also frosted) to filter the incoming light and make the solar cell less sensitive to other colours. While none of this is necessary, it does improve functionality and you can achieve similar results with hacked together solutions like wrapping it in red cellophane. All of the files for these parts are available at **hsmag.cc/yAeCPQ**.

You can test the trip-wire by connecting the laser to a 5 V power source and reading the voltage off the solar cell. I've been able to get a maximum of around 500 mV with the laser hitting even a portion of the solar cell. Using my hand to break the beam, I was able to see a drop to 180 mV from just the ambient light in my apartment. That's the best-case scenario; once the LEDs are lit and shining around the gate, I've consistently seen a drop of at least 50 mV. If you avoid lighting the LEDs red, you'll get a better result as well.

The last part of the sensor-system is the tilt switch. These mechanical switches have a little metal ball

RED LIGHT… DOESN'T SPELL DANGER

Why use a red laser? Although the solar cell we're using has peak sensitivity at lower wavelengths (like red), blue and green lasers would have a more noticeable impact on the voltage change since they are much brighter. But that added brightness also makes them more dangerous for our eyes – even the scattered light off the solar cell can quickly become uncomfortable. You should never use a laser outside of the visible spectrum (like UV) because you won't be able to see it and will be clueless while it burns out your retina.



Above
Everything else you can connect directly to a stripped end of one of the phone cables

inside and when it's tilted, the ball rolls down to the contacts and closes the switch. Drill a hole and glue in your tilt switch and make sure that it is in the OFF state while the gate is vertical. You can even angle it down a little bit to prevent accidental triggers.

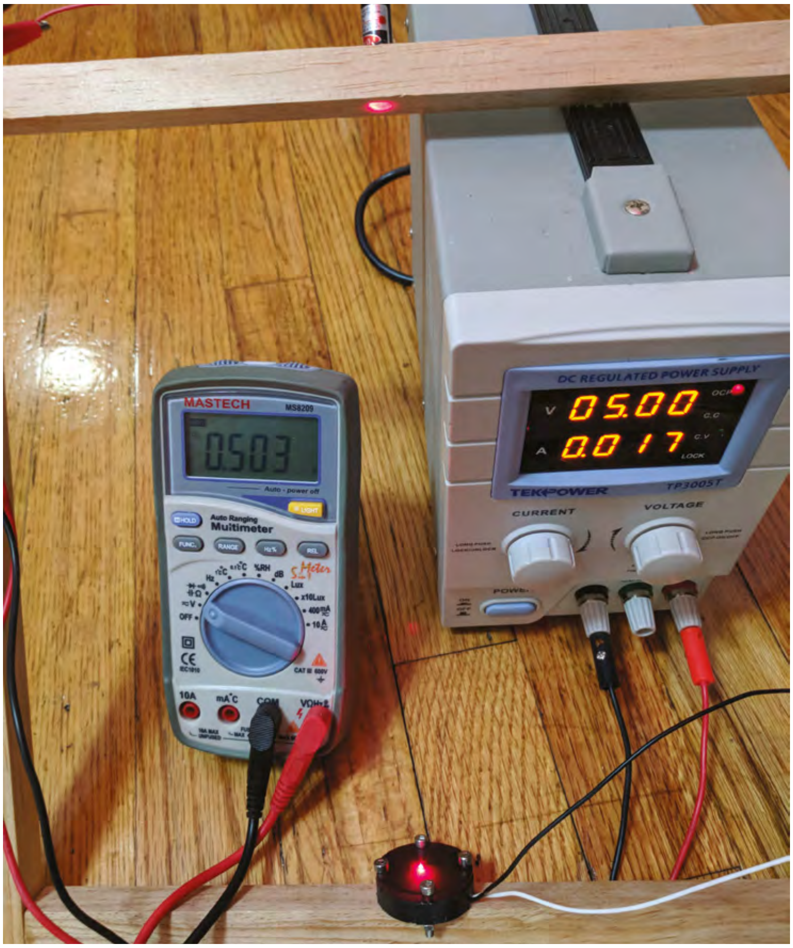
If you want, you can drill a few holes in the bottom of the gate and screw in some 1/4" (0.6 cm) inserts. That will let you easily attach each gate to any tripod or camera mount. We put one on each end and one in the centre to keep our options open.

ADD THE LEDS

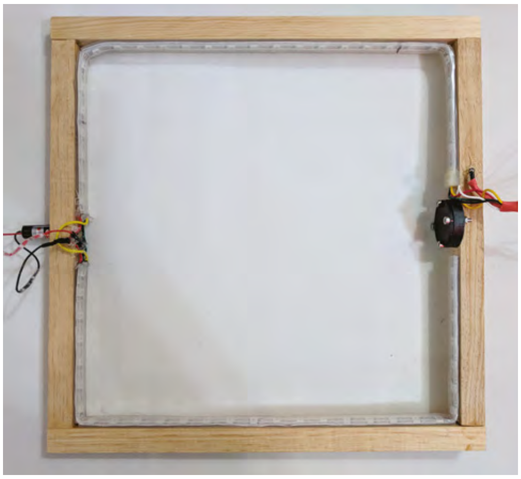
We're almost done with the gate; we just need to add the LEDs. These strips can be cut between any LED along the dotted lines. Run the perimeter of the gate starting at the sensor and cut the strip when you reach the laser aperture. Line up the strip on the other side of the laser and run the perimeter back down the other side. We made some very small cuts on the sides of the protective plastic at each corner to get a sharp bend. Be careful not to cut any of the electronics inside! Pay attention to the direction of the data in the LED strips. You want to make sure the data is always flowing in the same direction and you don't accidentally line up two Din connections around the laser. Tape the LED strips until you're done soldering all the wires. Once they're hooked up, you should use hot glue to secure the strips in place.

All that's left now is to follow the circuit diagram and solder everything together!

Make as many gates as you want (and can fit on the Arduino) to create a fun indoor drone racing course. There's plenty of room to expand the game: try to make the gates count down to put time pressure on the player and change faster with each successful pass; add a score counter, and track fastest times; add speakers to play cool sound effects; put the gates on servos to make them spin; or make a two-player race with gates that light different colours. We've also tinkered with making a laser maze that you have to snake through. In that scenario you want to see the beam, so you may want to upgrade to blue or green lasers. Throw in a fog machine or some spray haze and you've got a great party game. □



Above
You can test the trip wire by connecting the laser to a 5V power source and reading the voltage off of the solar cell with a multimeter



Left
Have all of your wires exit the gate on the same side. You'll want to bundle and hide them to prevent the drones from getting snagged