Firewalker LED Sneakers

Created by Becky Stern

https://learn.adafruit.com/firewalker-led-sneakers

Last updated on 2025-12-09 03:51:24 PM EST

Table of Contents

Overview

Light up your stride! Mod a pair of high-tops with NeoPixel strip and FLORA, Adafruit's wearable electronics platform. These sneakers use a velostat step sensor in the heel to trigger firey animations as you walk!

This project involves sewing and soldering! We recommend you read the Getting Started with FLORA (https://adafru.it/aSZ) guide before beginning this project.

This project is a collaboration with Phillip Burgess aka Paint Your Dragon! Glamour shots by John de Cristofaro. Project help by Risa Rose, and special thanks to Ary from Anamanaguchi (https://adafru.it/cDb) for modeling!


NEVER
LOG
OFF

Tools & Supplies

For **a pair of** shoes, you will need:

- 2 FLORA main boards (http://adafru.it/659)
- velostat (http://adafru.it/1361)
- 2 meters NeoPixel 60-LED strip in black (http://adafru.it/1461) or white (http://adafru.it/1138)
- 2 3xAAA battery holder with switch (http://adafru.it/727)
- conductive thread (http://adafru.it/641)
- shoes, we used high-top Vans (https://adafru.it/cDa)
- premium jumpers (https://adafru.it/fV8) or three pieces of silicone coated stranded wire (https://adafru.it/fV5)

other supplies:

- fine grit sandpaper
- tape
- Permatex 66B adhesive (or any adhesive that sticks to both the silicone LED sheathing and your shoe material)
- marking pencil
- plain thread
- alligator clips (https://adafru.it/Ccs)
- usb cable for plugging in FLORA

Use needles (http://adafru.it/615) to stitch up the conductive thread to the sensor!

Use flush snips (http://adafru.it/152) and wire strippers (http://adafru.it/527) to deal with the NeoPixel strip.

Sharp scissors make sewing more fun!

Circuit Diagram

Click to enlarge.

Velostat pressure sensor (described in the next step) connected to GND and pad marked D9 (this pin is also A9, and we will be using it as an analog input).

NeoPixel strip DIN -> FLORA D6
NeoPixel strip +5V -> FLORA VBATT
NeoPixel strip GND -> FLORA GND (any)

3xAAA battery pack plugged into FLORA's JST connector and tucked into the shoes laces

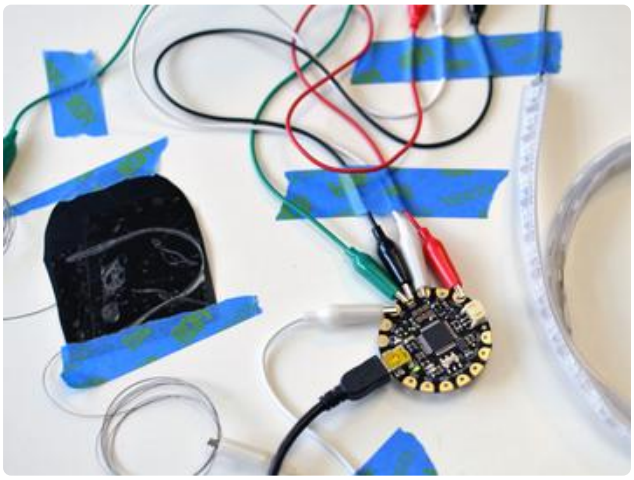
Make Velostat Step Sensors

Cut a heel-shaped piece of velostat and tape a loop of conductive thread to it, with a long (at least 18 inches) tail sticking out to one side.

Cut another piece for the other shoe and repeat.

Flip the pieces of velostat over and tape another loop of conductive thread . The tails should be on the same side, but at least two inches apart.

Test Circuit



Solder a piece of 3-ply stranded ribbon cable to the three pads on your NeoPixel strip. (Wires from our premium jumpers (https://adafru.it/fV8) work well for this, as do three pieces of silicone coated stranded wire (https://adafru.it/fV5).)

Hook up your sensor and LED strip with alligator clips, according to the circuit diagram (https://adafru.it/cD9). Program your FLORA (https://adafru.it/aWE) with the sample code below, which will help you calibrate your handmade sensor.

Open up Arduino's serial monitor and use a blunt object like a roll of tape or a plastic cup to press on the velostat sensor. You should see the sensor values streaming onto the screen.

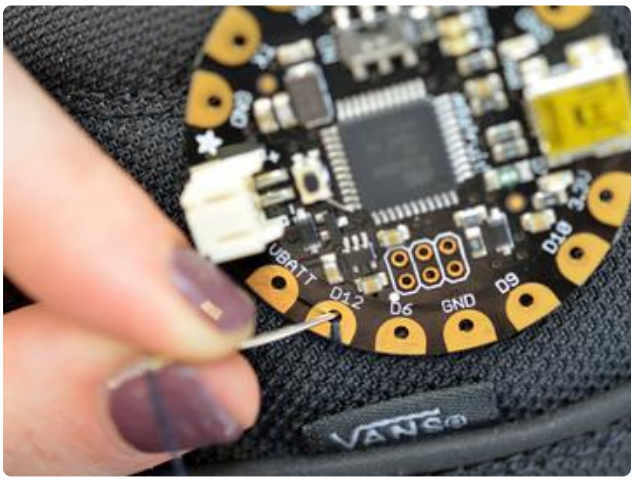
Since there are 60 LEDs connected to this circuit, it can be too much current for some USB ports! We found that our computer's onboard port could power the circuit, but the keyboard USB port could not.

The sketch below is programmed to only light up the first 10 pixels in the strand so it should work with any USB port:

```
#include &lt;Adafruit_NeoPixel.h&gt;

const int analogInPin = A9;  // Analog input pin that the potentiometer is attached
to
Adafruit_NeoPixel strip = Adafruit_NeoPixel(10, 6, NEO_GRB + NEO_KHZ800);
int sensorValue = 0;        // value read from the pot

void setup() {
  // initialize serial communications at 9600 bps:
  Serial.begin(9600);
 pinMode(9, INPUT_PULLUP);
   strip.begin();
  strip.show(); // Initialize all pixels to 'off'
```

```
}

void loop() {
  // read the analog in value:
  sensorValue = analogRead(analogInPin);
  // print the results to the serial monitor:
  Serial.print("sensor = " );
  Serial.println(sensorValue);

if (sensorValue &lt; 100){
  Serial.println("leds triggered");
colorWipe(strip.Color(255, 0, 0), 25);
colorWipe(strip.Color(0, 0, 0), 25);
}

}

void colorWipe(uint32_t c, uint8_t wait) {
  for(uint16_t i=0; i&lt;strip.numPixels(); i++) {
      strip.setPixelColor(i, c);
      strip.show();
      delay(wait);
  }
}
```

Sew Sensors to FLORA



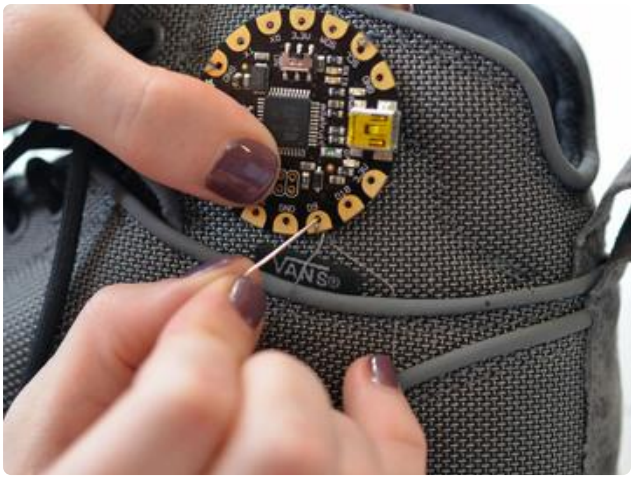
Pick a spot for the FLORA main board to live on your shoes-- we chose the upper part of the high-tops-- and stitch the board in place with plain thread through two unused pads (D12 and SCL work nicely in this case).

Pick up one sensor thread tail and thread it through a needle, then pierce the needle through the side of the shoe, right near the sole. Repeat with the other piece of thread attached to the sensor, keeping the two threads a short distance from each other.

Pull the two threads taut, which will draw the sensor into the heel of the shoe. We positioned ours underneath the shoe's insole.

If your shoes are made of a hard-to-sew-through material, it really helps to use pliers to manipulate the sewing needle.

u can get your hands on one, a stitching awl can be helpful for sewing ugh tough materials like leather and vinyl.



Stitch the threads separately up to the FLORA main board, hiding the thread in the shoe material if desired.

Referring to the circuit diagram, stitch one of these threads to GND on FLORA, and the other to D9 (which we will be using as an analog input).

Knot the thread on the inside, seal your knots with fray check, and snip off excess thread.

Now you can test your sensors! With a long USB cable plugged into the computer, put on one of the shoes and open the serial monitor. Notice how the sensor values change as you step! Repeat for the other shoe.

Attach NeoPixel LED Strip

Sand the surface of the smooth shoe rubber for better adhesion.

Wipe the entire rubber surface with alcohol to make sure it's super clean, and do the same for the back of the LED strip.

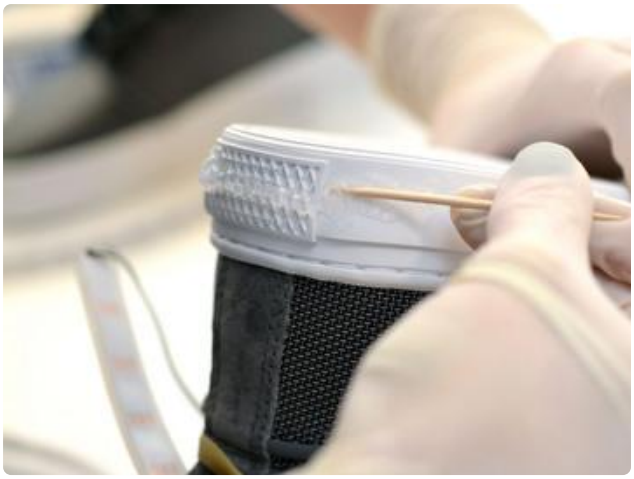
Put the shoes on and bend the toes as much as you can. Use a marking pencil to mark off the section that bends the most--you'll skip this section when applying glue.

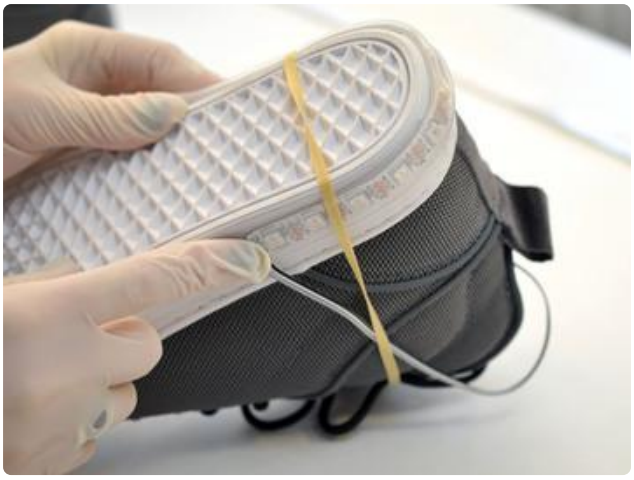
Do this on both the “outer” and “inner” faces of each shoe.

Set up for a messy work surface by laying down scrap paper and donning rubber gloves. Keep paper towels nearby in case of an unexpected spill.

You'll need rubber bands and toothpicks to aid in glueing.

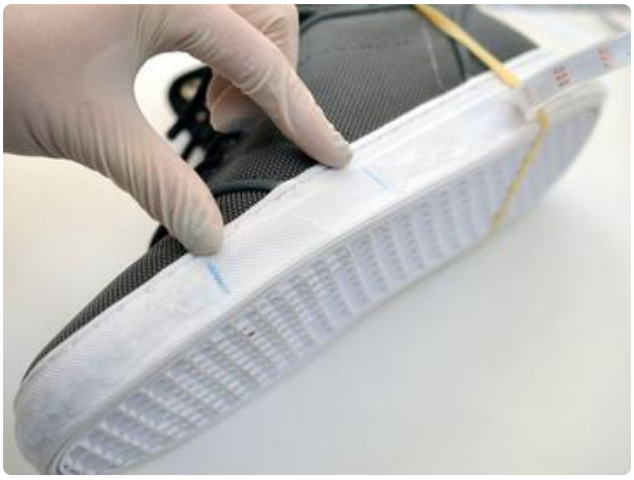
Apply a bead of adhesive to the shoe sole, starting at the inside of the heel, and stick the start of your NeoPixel strand to it.





Continue applying adhesive around the back of the heel, smudging it out with a toothpick if necessary.

Wrap the strip around the heel and secure with rubber bands as you go.

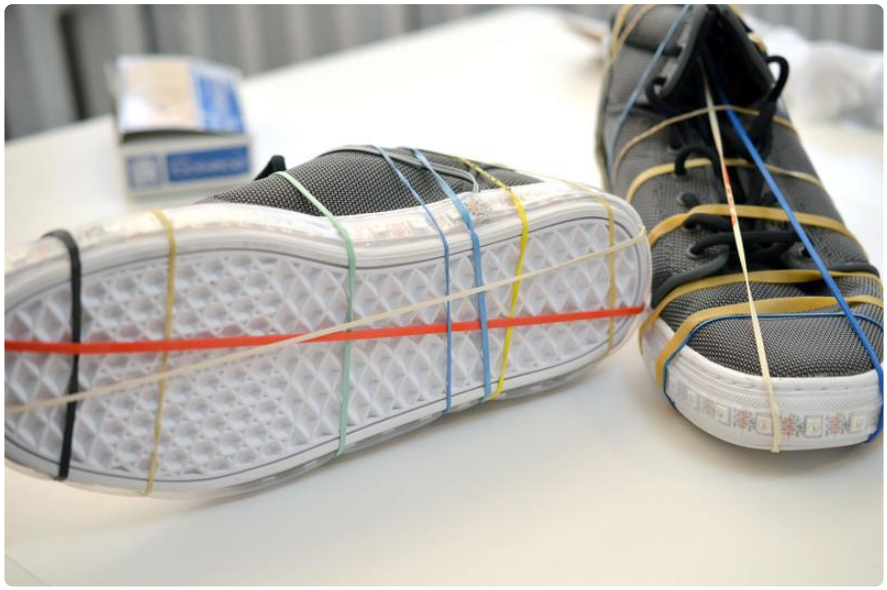


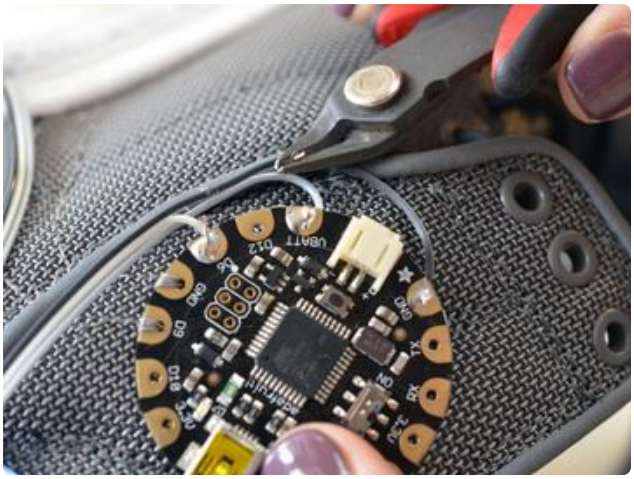
Remember to skip the rubber between your markings. This will allow the LED strip to float away from the shoe when it bends. Otherwise the sharp bending could crack the flexible circuit board inside the strip, since it's not meant to bend laterally.

Continue glueing all the way around, remembering to skip the second bendy sections after you come around the toe.

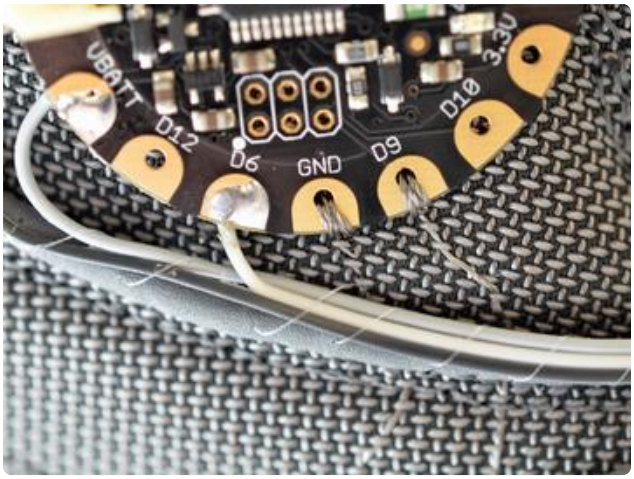
When you get back to where you started, use diagonal flush snips to trim the LED strip to length.

Squish a little bit of glue into the cut ends of the LED casing to create a water resistant seal, and use rubber bands to "clamp" the strip to the shoes while the glue dries, ~24 hours.

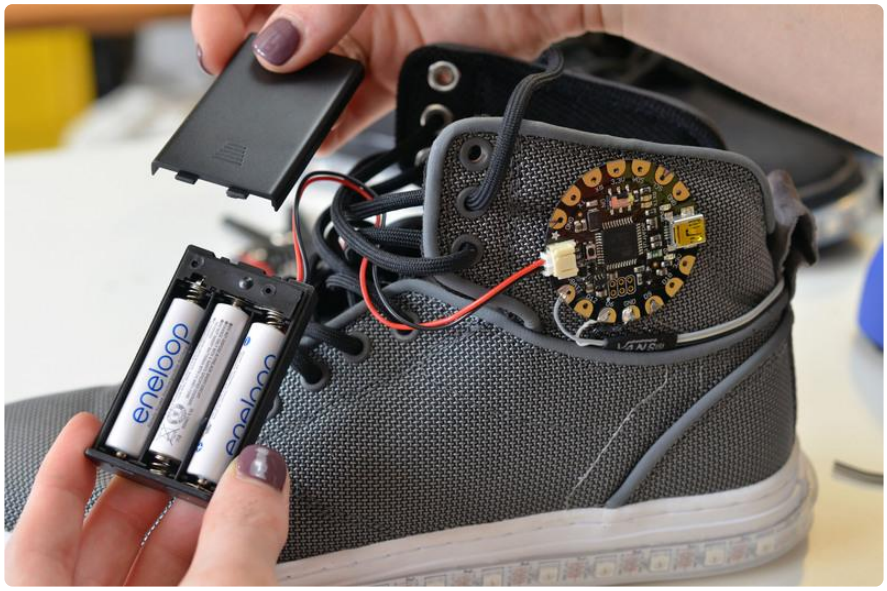




Route the NeoPixel strip's ribbon cable to the FLORA main board, and strip and solder the three wires according to the [circuit diagram](https://adafru.it/cD9).



Use clear thread or a thread that matches your shoes, and whip-stitch the ribbon cable to the shoes, securing it in place.



Plug in the 3xAAA battery pack and tuck it in the laces of the shoe, or wherever is comfortable!

Firewalker Code

After you've tested your shoes with the simple sketch, you can upgrade to this sparkly, firey animation. Copy the sketch and load it onto your FLORA shoes with the Adafruit Arduino IDE:

```
// SPDX-FileCopyrightText: 2020 Phillip Burgess for Adafruit Industries
//
// SPDX-License-Identifier: MIT

// 'Firewalker' LED sneakers sketch for Adafruit NeoPixels by Phillip Burgess

#include <Adafruit_NeoPixel.h>

const uint8_t gamma1[] PROGMEM = { // Gamma correction table for LED brightness
    0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
    0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  1,  1,  1,  1,  1,
    1,  1,  1,  1,  1,  1,  1,  1,  1,  2,  2,  2,  2,  2,  2,  2,
    2,  3,  3,  3,  3,  3,  3,  3,  4,  4,  4,  4,  4,  5,  5,  5,
    5,  6,  6,  6,  6,  7,  7,  7,  7,  8,  8,  8,  9,  9,  9, 10,
   10, 10, 11, 11, 11, 12, 12, 13, 13, 13, 14, 14, 15, 15, 16, 16,
   17, 17, 18, 18, 19, 19, 20, 20, 21, 21, 22, 22, 23, 24, 24, 25,
   25, 26, 27, 27, 28, 29, 29, 30, 31, 32, 32, 33, 34, 35, 35, 36,
   37, 38, 39, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 50,
   51, 52, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 66, 67, 68,
   69, 70, 72, 73, 74, 75, 77, 78, 79, 81, 82, 83, 85, 86, 87, 89,
   90, 92, 93, 95, 96, 98, 99,101,102,104,105,107,109,110,112,114,
  115,117,119,120,122,124,126,127,129,131,133,135,137,138,140,142,
  144,146,148,150,152,154,156,158,160,162,164,167,169,171,173,175,
  177,180,182,184,186,189,191,193,196,198,200,203,205,208,210,213,
  215,218,220,223,225,228,231,233,236,239,241,244,247,249,252,255 };

// LEDs go around the full perimeter of the shoe sole, but the step animation
// is mirrored on both the inside and outside faces, while the strip doesn't
// necessarily start and end at the heel or toe.  These constants help configure
// the strip and shoe sizes, and the positions of the front- and rear-most LEDs.
// Becky's shoes: 39 LEDs total, 20 LEDs long, LED #5 at back.
// Phil's shoes: 43 LEDs total, 22 LEDs long, LED #6 at back.
#define N_LEDS        39 // TOTAL number of LEDs in strip
#define SHOE_LEN_LEDS 20 // Number of LEDs down ONE SIDE of shoe
```

```
#define SHOE_LED_BACK  5 // Index of REAR-MOST LED on shoe
#define STEP_PIN      A2 // Analog input for footstep
#define LED_PIN       A0 // NeoPixel strip is connected here
#define MAXSTEPS       3 // Process (up to) this many concurrent steps

Adafruit_NeoPixel strip = Adafruit_NeoPixel(N_LEDS, LED_PIN, NEO_GRB + NEO_KHZ800);

// The readings from the sensors are usually around 250-350 when not being pressed,
// then dip below 100 when the heel is standing on it (for Phil's shoes; Becky's
// don't dip quite as low because she's smaller).
#define STEP_TRIGGER    150  // Reading must be below this to trigger step
#define STEP_HYSTERESIS 200  // After trigger, must return to this level

int
  stepMag[MAXSTEPS],  // Magnitude of steps
  stepX[MAXSTEPS],    // Position of 'step wave' along strip
  mag[SHOE_LEN_LEDS], // Brightness buffer (one side of shoe)
  stepFiltered,       // Current filtered pressure reading
  stepCount,          // Number of 'frames' current step has lasted
  stepMin;            // Minimum reading during current step
uint8_t
  stepNum = 0,        // Current step number in stepMag/stepX tables
  dup[SHOE_LEN_LEDS]; // Inside/outside copy indexes
boolean
  stepping  = false;  // If set, step was triggered, waiting to release


void setup() {
  pinMode(9, INPUT_PULLUP); // Set internal pullup resistor for sensor pin
  // As previously mentioned, the step animation is mirrored on the inside and
  // outside faces of the shoe.  To avoid a bunch of math and offsets later, the
  // 'dup' array indicates where each pixel on the outside face of the shoe should
  // be copied on the inside.  (255 = don't copy, as on front- or rear-most LEDs).
  // Later, the colors for the outside face of the shoe are calculated and then get
  // copied to the appropriate positions on the inside face.
  memset(dup, 255, sizeof(dup));
  int8_t a, b;
  for(a=1               , b=SHOE_LED_BACK-1            ; b>=0    ;) dup[a++] = b--;
  for(a=SHOE_LEN_LEDS-2, b=SHOE_LED_BACK+SHOE_LEN_LEDS; b<N_LEDS;) dup[a--] = b++;

  // Clear step magnitude and position buffers
  memset(stepMag, 0, sizeof(stepMag));
  memset(stepX  , 0, sizeof(stepX));
  strip.begin();
  stepFiltered = analogRead(STEP_PIN); // Initial input
}

void loop() {
  uint8_t i, j;

  // Read analog input, with a little noise filtering
  //stepFiltered = ((stepFiltered * 3) + analogRead(STEP_PIN)) >> 2;
    stepFiltered = (((stepFiltered * 3) - 100) + analogRead(STEP_PIN)) >> 2;

  // The strip doesn't simply display the current pressure reading.  Instead,
  // there's a bit of an animated flourish from heel to toe.  This takes time,
  // and during quick foot-tapping there could be multiple step animations
  // 'in flight,' so a short list is kept.
  if(stepping) { // If a step was previously triggered...
    if(stepFiltered >= STEP_HYSTERESIS) { // Has step let up?
      stepping = false;                   // Yep! Stop monitoring.
      // Add new step to the step list (may be multiple in flight)
      stepMag[stepNum] = (STEP_HYSTERESIS - stepMin) * 6; // Step intensity
      stepX[stepNum]   = -80; // Position starts behind heel, moves forward
      if(++stepNum >= MAXSTEPS) stepNum = 0; // If many, overwrite oldest
    } else if(stepFiltered < stepMin) stepMin = stepFiltered; // Track min val
  } else if(stepFiltered < STEP_TRIGGER) { // No step yet; watch for trigger
    stepping = true;         // Got one!
    stepMin  = stepFiltered; // Note initial value
```

```
  }

  // Render a 'brightness map' for all steps in flight.  It's like
  // a grayscale image; there's no color yet, just intensities.
  int mx1, px1, px2, m;
  memset(mag, 0, sizeof(mag));    // Clear magnitude buffer
  for(i=0; i<MAXSTEPS; i++) {     // For each step...
    if(stepMag[i] <= 0) continue; // Skip if inactive
    for(j=0; j<SHOE_LEN_LEDS; j++) { // For each LED...
      // Each step has sort of a 'wave' that's part of the animation,
      // moving from heel to toe.  The wave position has sub-pixel
      // resolution (4X), and is up to 80 units (20 pixels) long.
      mx1 = (j << 2) - stepX[i]; // Position of LED along wave
      if((mx1 <= 0) || (mx1 >= 80)) continue; // Out of range
      if(mx1 > 64) { // Rising edge of wave; ramp up fast (4 px)
        m = ((long)stepMag[i] * (long)(80 - mx1)) >> 4;
      } else { // Falling edge of wave; fade slow (16 px)
        m = ((long)stepMag[i] * (long)mx1) >> 6;
      }
      mag[j] += m; // Add magnitude to buffered sum
    }
    stepX[i]++; // Update position of step wave
    if(stepX[i] >= (80 + (SHOE_LEN_LEDS << 2)))
      stepMag[i] = 0; // Off end; disable step wave
    else
      stepMag[i] = ((long)stepMag[i] * 127L) >> 7; // Fade
  }

  // For a little visual interest, some 'sparkle' is added.
  // The cumulative step magnitude is added to one pixel at random.
  long sum = 0;
  for(i=0; i<MAXSTEPS; i++) sum += stepMag[i];
  if(sum > 0) {
    i = random(SHOE_LEN_LEDS);
    mag[i] += sum / 4;
  }

  // Now the grayscale magnitude buffer is remapped to color for the LEDs.
  // The code below uses a blackbody palette, which fades from white to yellow
  // to red to black.  The goal here was specifically a "walking on fire"
  // aesthetic, so the usual ostentatious rainbow of hues seen in most LED
  // projects is purposefully skipped in favor of a more plain effect.
  uint8_t r, g, b;
  int     level;
  for(i=0; i<SHOE_LEN_LEDS; i++) { // For each LED on one side...
    level = mag[i];                // Pixel magnitude (brightness)
    if(level < 255) {              // 0-254 = black to red-1
      r = pgm_read_byte(&gamma1[level]);
      g = b = 0;
    } else if(level < 510) {       // 255-509 = red to yellow-1
      r = 255;
      g = pgm_read_byte(&gamma1[level - 255]);
      b = 0;
    } else if(level < 765) {       // 510-764 = yellow to white-1
      r = g = 255;
      b = pgm_read_byte(&gamma1[level - 510]);
    } else {                       // 765+ = white
      r = g = b = 255;
    }
    // Set R/G/B color along outside of shoe
    strip.setPixelColor(i+SHOE_LED_BACK, r, g, b);
    // Pixels along inside are funny...
    j = dup[i];
    if(j < 255) strip.setPixelColor(j, r, g, b);
  }

  strip.show();
  delayMicroseconds(1500);
}
```

```
// 'Firewalker' LED sneakers sketch for Adafruit NeoPixels by Phillip Burgess

#include &lt;Adafruit_NeoPixel.h&gt;

const uint8_t gamma[] PROGMEM = { // Gamma correction table for LED brightness
    0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
    0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  1,  1,  1,  1,
    1,  1,  1,  1,  1,  1,  1,  1,  1,  2,  2,  2,  2,  2,  2,  2,
    2,  3,  3,  3,  3,  3,  3,  3,  4,  4,  4,  4,  4,  5,  5,  5,
    5,  6,  6,  6,  6,  7,  7,  7,  7,  8,  8,  8,  9,  9,  9, 10,
   10, 10, 11, 11, 11, 12, 12, 13, 13, 13, 14, 14, 15, 15, 16, 16,
   17, 17, 18, 18, 19, 19, 20, 20, 21, 21, 22, 22, 23, 24, 24, 25,
   25, 26, 27, 27, 28, 29, 29, 30, 31, 32, 32, 33, 34, 35, 35, 36,
   37, 38, 39, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 50,
   51, 52, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 66, 67, 68,
   69, 70, 72, 73, 74, 75, 77, 78, 79, 81, 82, 83, 85, 86, 87, 89,
   90, 92, 93, 95, 96, 98, 99,101,102,104,105,107,109,110,112,114,
  115,117,119,120,122,124,126,127,129,131,133,135,137,138,140,142,
  144,146,148,150,152,154,156,158,160,162,164,167,169,171,173,175,
  177,180,182,184,186,189,191,193,196,198,200,203,205,208,210,213,
  215,218,220,223,225,228,231,233,236,239,241,244,247,249,252,255 };

// LEDs go around the full perimeter of the shoe sole, but the step animation
// is mirrored on both the inside and outside faces, while the strip doesn't
// necessarily start and end at the heel or toe.  These constants help configure
// the strip and shoe sizes, and the positions of the front- and rear-most LEDs.
// Becky's shoes: 39 LEDs total, 20 LEDs long, LED #5 at back.
// Phil's shoes: 43 LEDs total, 22 LEDs long, LED #6 at back.
#define N_LEDS        39 // TOTAL number of LEDs in strip
#define SHOE_LEN_LEDS 20 // Number of LEDs down ONE SIDE of shoe
#define SHOE_LED_BACK  5 // Index of REAR-MOST LED on shoe
#define STEP_PIN      A9 // Analog input for footstep
#define LED_PIN        6 // NeoPixel strip is connected here
#define MAXSTEPS       3 // Process (up to) this many concurrent steps

Adafruit_NeoPixel strip = Adafruit_NeoPixel(N_LEDS, LED_PIN, NEO_GRB + NEO_KHZ800);

// The readings from the sensors are usually around 250-350 when not being pressed,
// then dip below 100 when the heel is standing on it (for Phil's shoes; Becky's
// don't dip quite as low because she's smaller).
#define STEP_TRIGGER    150  // Reading must be below this to trigger step
#define STEP_HYSTERESIS 200  // After trigger, must return to this level

int
  stepMag[MAXSTEPS],  // Magnitude of steps
  stepX[MAXSTEPS],    // Position of 'step wave' along strip
  mag[SHOE_LEN_LEDS], // Brightness buffer (one side of shoe)
  stepFiltered,       // Current filtered pressure reading
  stepCount,          // Number of 'frames' current step has lasted
  stepMin;            // Minimum reading during current step
uint8_t
  stepNum = 0,        // Current step number in stepMag/stepX tables
  dup[SHOE_LEN_LEDS]; // Inside/outside copy indexes
boolean
  stepping  = false;  // If set, step was triggered, waiting to release


void setup() {
  pinMode(9, INPUT_PULLUP); // Set internal pullup resistor for sensor pin
  // As previously mentioned, the step animation is mirrored on the inside and
  // outside faces of the shoe.  To avoid a bunch of math and offsets later, the
  // 'dup' array indicates where each pixel on the outside face of the shoe should
  // be copied on the inside.  (255 = don't copy, as on front- or rear-most LEDs).
  // Later, the colors for the outside face of the shoe are calculated and then get
  // copied to the appropriate positions on the inside face.
  memset(dup, 255, sizeof(dup));
  int8_t a, b;
  for(a=1              , b=SHOE_LED_BACK-1            ; b&gt;=0    ;) dup[a++] =
```

```
b--;
  for(a=SHOE_LEN_LEDS-2, b=SHOE_LED_BACK+SHOE_LEN_LEDS; b&lt;N_LEDS;) dup[a--] = b+
+;

  // Clear step magnitude and position buffers
  memset(stepMag, 0, sizeof(stepMag));
  memset(stepX  , 0, sizeof(stepX));
  strip.begin();
  stepFiltered = analogRead(STEP_PIN); // Initial input
}

void loop() {
  uint8_t i, j;

  // Read analog input, with a little noise filtering
  stepFiltered = ((stepFiltered * 3) + analogRead(STEP_PIN)) &gt;&gt; 2;

  // The strip doesn't simply display the current pressure reading.  Instead,
  // there's a bit of an animated flourish from heel to toe.  This takes time,
  // and during quick foot-tapping there could be multiple step animations
  // 'in flight,' so a short list is kept.
  if(stepping) { // If a step was previously triggered...
    if(stepFiltered &gt;= STEP_HYSTERESIS) { // Has step let up?
      stepping = false;                   // Yep! Stop monitoring.
      // Add new step to the step list (may be multiple in flight)
      stepMag[stepNum] = (STEP_HYSTERESIS - stepMin) * 6; // Step intensity
      stepX[stepNum]   = -80; // Position starts behind heel, moves forward
      if(++stepNum &gt;= MAXSTEPS) stepNum = 0; // If many, overwrite oldest
    } else if(stepFiltered &lt; stepMin) stepMin = stepFiltered; // Track min val
  } else if(stepFiltered &lt; STEP_TRIGGER) { // No step yet; watch for trigger
    stepping = true;         // Got one!
    stepMin  = stepFiltered; // Note initial value
  }

  // Render a 'brightness map' for all steps in flight.  It's like
  // a grayscale image; there's no color yet, just intensities.
  int mx1, px1, px2, m;
  memset(mag, 0, sizeof(mag));    // Clear magnitude buffer
  for(i=0; i&lt;MAXSTEPS; i++) {     // For each step...
    if(stepMag[i] &lt;= 0) continue; // Skip if inactive
    for(j=0; j&lt;SHOE_LEN_LEDS; j++) { // For each LED...
      // Each step has sort of a 'wave' that's part of the animation,
      // moving from heel to toe.  The wave position has sub-pixel
      // resolution (4X), and is up to 80 units (20 pixels) long.
      mx1 = (j &lt;&lt; 2) - stepX[i]; // Position of LED along wave
      if((mx1 &lt;= 0) || (mx1 &gt;= 80)) continue; // Out of range
      if(mx1 &gt; 64) { // Rising edge of wave; ramp up fast (4 px)
        m = ((long)stepMag[i] * (long)(80 - mx1)) &gt;&gt; 4;
      } else { // Falling edge of wave; fade slow (16 px)
        m = ((long)stepMag[i] * (long)mx1) &gt;&gt; 6;
      }
      mag[j] += m; // Add magnitude to buffered sum
    }
    stepX[i]++; // Update position of step wave
    if(stepX[i] &gt;= (80 + (SHOE_LEN_LEDS &lt;&lt; 2)))
      stepMag[i] = 0; // Off end; disable step wave
    else
      stepMag[i] = ((long)stepMag[i] * 127L) &gt;&gt; 7; // Fade
  }

  // For a little visual interest, some 'sparkle' is added.
  // The cumulative step magnitude is added to one pixel at random.
  long sum = 0;
  for(i=0; i&lt;MAXSTEPS; i++) sum += stepMag[i];
  if(sum &gt; 0) {
    i = random(SHOE_LEN_LEDS);
    mag[i] += sum / 4;
  }
```

```
  // Now the grayscale magnitude buffer is remapped to color for the LEDs.
  // The code below uses a blackbody palette, which fades from white to yellow
  // to red to black.  The goal here was specifically a "walking on fire"
  // aesthetic, so the usual ostentatious rainbow of hues seen in most LED
  // projects is purposefully skipped in favor of a more plain effect.
  uint8_t r, g, b;
  int     level;
  for(i=0; i&lt;SHOE_LEN_LEDS; i++) { // For each LED on one side...
    level = mag[i];               // Pixel magnitude (brightness)
    if(level &lt; 255) {              // 0-254 = black to red-1
      r = pgm_read_byte(&amp;gamma[level]);
      g = b = 0;
    } else if(level &lt; 510) {       // 255-509 = red to yellow-1
      r = 255;
      g = pgm_read_byte(&amp;gamma[level - 255]);
      b = 0;
    } else if(level &lt; 765) {       // 510-764 = yellow to white-1
      r = g = 255;
      b = pgm_read_byte(&amp;gamma[level - 510]);
    } else {                      // 765+ = white
      r = g = b = 255;
    }
    // Set R/G/B color along outside of shoe
    strip.setPixelColor(i+SHOE_LED_BACK, r, g, b);
    // Pixels along inside are funny...
    j = dup[i];
    if(j &lt; 255) strip.setPixelColor(j, r, g, b);
  }

  strip.show();
  delayMicroseconds(1500);
}
```

Wear 'em!

Although the NeoPixel strip is water resistant, the FLORA circuit is unprotected, so flip your shoes off if you get stuck in the rain.

Even with the precautions we took near the bendy part of the shoes, the NeoPixel strip might eventually crack if it endures too much stress, so wear your shoes cautiously for maximum life!

Frequently Asked Questions

Can I use LiPoly batteries instead of alkaline packs?

Sure! Alkaline packs are recommended for beginners, but advanced users with experience using/caring for rechargeable lipoly batteries might upgrade to a 500mAh cell per shoe. Read more in the guide on battery powering your wearable electronics (https://adafru.it/VyF).

What happens if it rains? Can I waterproof my Firewalkers?

The LED strip is in a weather resistant sheathing and should suffice for short grass and wet pavement, but if you want to go out in the rain, you should switch out the conductive thread for stranded wire and take steps to waterproof your circuit board. Here's a video on "rugged-izing" your wearables:

Can I use GEMMA instead? Is the code compatible?

So this project was published before GEMMA existed. Yes, you can use GEMMA (in the sketch, change the sensor pin to A2 and the pixel output to 1), it's just harder to calibrate your sensor since there's no simple serial debugging on GEMMA. You can hack a serial connection with our guide (https://adafru.it/Vza), or find your sensor levels through time-consuming trial and error.

How can I change the color scheme of the firewalker animation?

Adafruit support engineer Bill writes (https://adafru.it/cKn):

> The firewalker code is a little tricky. It calculates all the color values before calling setPixelColor.
>
> What I did was simply exchange the "r" and "b" values so the color spectrum starts at blue instead of red.
>
> Just replace the similar looking 'for' loop in your code with:

```
for(i=0; i&lt;SHOE_LEN_LEDS; i++) { // For each LED on one side...
  level = mag[i];                // Pixel magnitude (brightness)
  if(level &lt; 255) {              // 0-254 = black to blue-1
    b = pgm_read_byte(&amp;gamma[level]);
    g = r = 0;
```

```
    } else if(level &lt; 510) {       // 255-509 = blue to cyan-1
      b = 255;
      g = pgm_read_byte(&amp;gamma[level - 255]);
      r = 0;
    } else if(level &lt; 765) {       // 510-764 = cyan to white-1
      b = g = 255;
      r = pgm_read_byte(&amp;gamma[level - 510]);
    } else {                        // 765+ = white
      r = g = b = 255;
    }
    // Set R/G/B color along outside of shoe
    strip.setPixelColor(i+SHOE_LED_BACK, r, g, b);
    // Pixels along inside are funny...
    j = dup[i];
    if(j &lt; 255) strip.setPixelColor(j, r, g, b);
  }
```

Can I use a different density of NeoPixel strip?

Yep, the code has a variable for the index of the pixels at the front and rear of the shoe, which will adjust the animation accordingly.

When I try to program my board I get the error "A9 was not declared in this scope"

You probably don't have the right board selected in the Arduino IDE. A9 doesn't exist on an Arduino Uno, so make sure you have Adafruit Flora selected as your board under Hardware-> Board.

Over time the strip cracks where the shoe bends. How can I avoid this?

answer

I don't want to make these. Where can I buy shoes like this?!

If you want to commission a maker to build these for your, post up in the Adafruit Jobs board (https://adafru.it/fVn)! We've seen similar looking LED sneakers (without the rad pressure-sensitive fire animation) on Ebay and Etsy, so you may have some luck finding some online.