# *A Portable 3D Zoetrope*

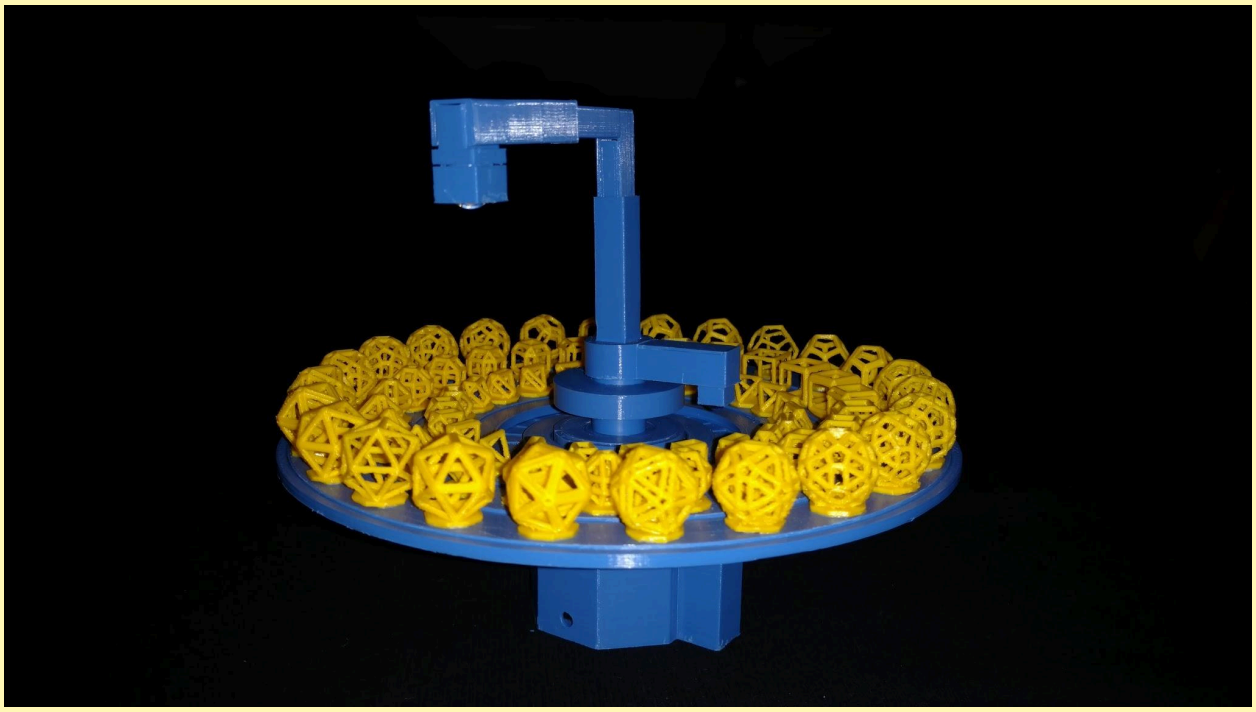
By Paul S. Strauss

Demonstrated at Maker Faire Bay Area 2017.

This document describes a portable 3D zoetrope that I designed and built. It is inspired by the large, motorized, 3D zoetropes built by Pixar and others. What makes this one different is that there is no motor—you hold this and spin it by hand. It is 3D-printed, except for the electronics and some magnets. I designed the entire zoetrope, including all of the original animated models, in OpenSCAD.

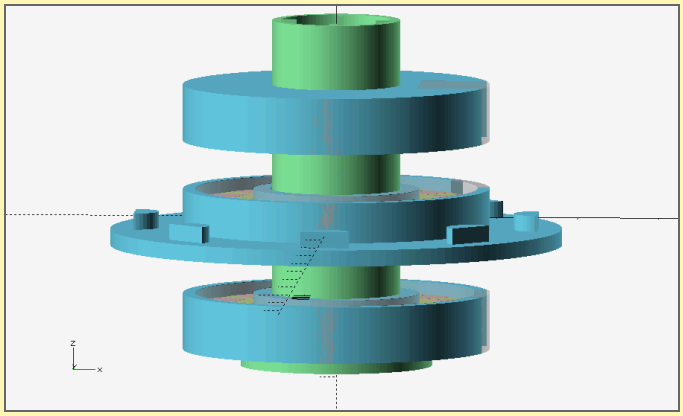
Like other 3D zoetropes, this one has a spinning platform and a strobe light that flashes at intervals to illuminate the models, giving the illusion of animation.

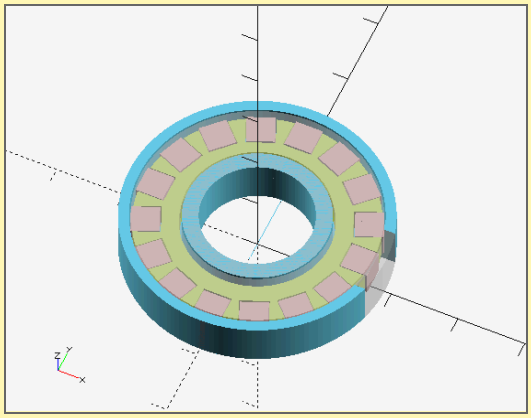
This video gives an overview of the zoetrope's appearance and features and shows it in operation. [Updated video added March 6, 2024.]



## Magnetic Bearing

Because the zoetrope is hand-powered, the platform needs to spin freely for a reasonably long time. Since all of the moving parts are plastic, friction can be a problem. I created a magnetic bearing that reduces the friction.

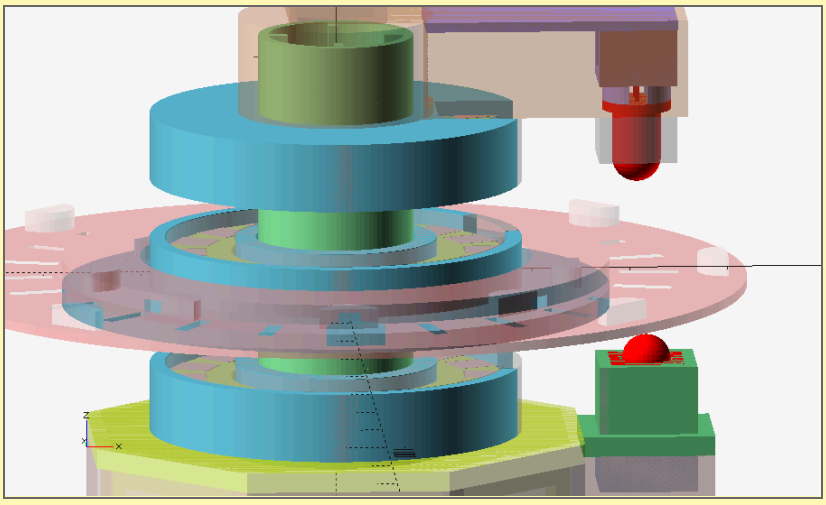




The image on the left shows the axle (green) and three rings forming the magnetic bearing. The top and bottom rings fit tightly on the axle, while the middle one is free to spin; it also has pegs for attaching the rest of the platform. Each of the three rings contains 16 Buckycube magnets, as shown in the image on the right. The magnets are arranged so that the top and bottom rings both repel the center ring, causing it to hover between them. The main source of friction is where the center ring touches the axle, which isn't that much.

## Strobe Timing

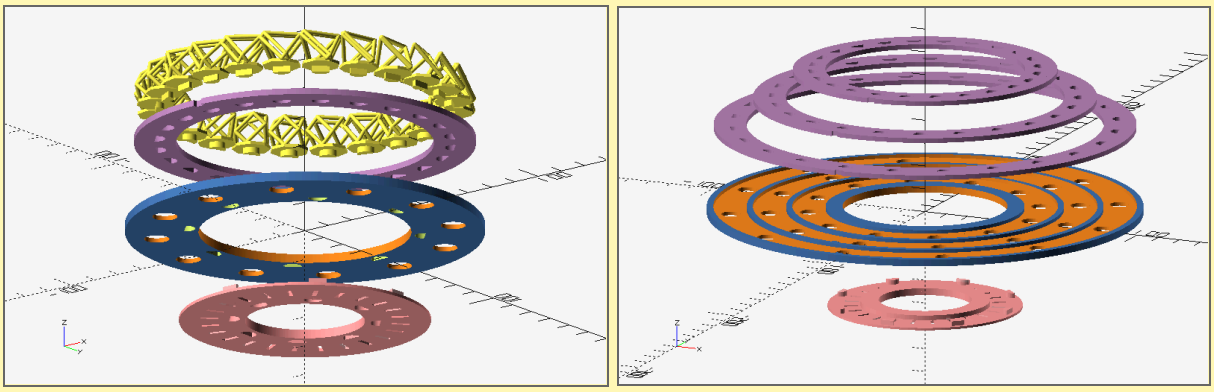
Motorized zoetropes rotate at a constant speed, so their strobe timing is simple. Since this zoetrope is spun by hand, its rotation rate is anything but constant, making it challenging to time the strobe. I used an infrared emitter/detector pair along with a rotating slotted screen to solve this.



The screen (pink) is attached to the center bearing ring so it rotates with the platform. There is one slot for each animated model on the platform's stage, so the detector diode (below the screen) receives the infrared signal from the always-on emitter diode (above the screen) at the precisely correct frequency based on the rotation rate.

# Modular Platform

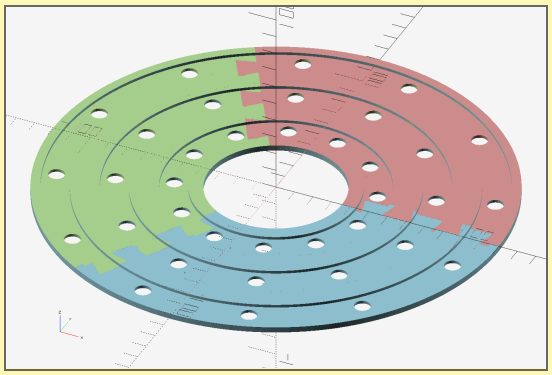
The platform consists of the timing screen, the stage, and the animated models. Rather than 3D print the entire platform as one piece, I decided to make it modular so I could experiment with different screens, stages, and models without having to use large amounts of time and plastic.



The platform on the left, shown from below, consists of a 24-slot screen (pink), a stage (blue), and a single track (purple) holding 24 models (yellow). The same hemicylindrical peg and hole rings are used to join all parts. The stage contains extra holes to make it lighter and less wasteful of plastic, and also to make it easier to remove the model tracks. The platform on the right has a stage that holds three tracks and is about as big as I can print on my 3D printer. The models are not shown in this image for simplicity.

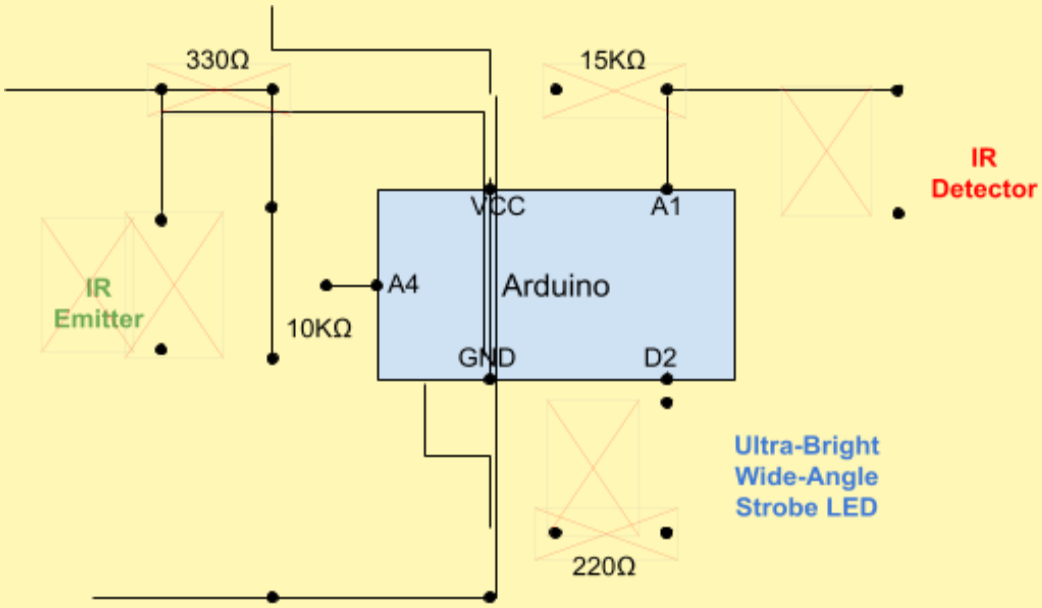
Each model is printed with a base that has a peg to attach it to a track. Since 3D printing is not exact, I find that using a little white glue keeps the models attached pretty well. It is very easy to remove an entire track with its models from a stage to try different animations. I'm hoping to create lots of different animations in the future. If I get access to a larger 3D printer, I may also create larger stages and tracks.

Update: I was able to print a larger stage (120mm) and track in pieces that fit together with dovetail joints. It worked out pretty well.



## Electronics

The octagonal base of the zoetrope is hollow and contains a TinyDuino processor board with lithium battery, USB board for connectivity and charging, and a proto board to connect it to the rest of the electronics. Here is the very simple circuit diagram for the whole thing.




The 10KΩ trim potentiometer connected to analog input pin A4 allows for some experimentation in the control program. I have used it to vary the IR threshold and the strobe duration during zoetrope operation to find optimal values. There is a small hole in the side of the base for adjusting the trimpot with a small screwdriver.

A hole in the bottom of the base provides access to the on/off switch for the TinyDuino and the USB port for reprogramming or charging.

## Programming

The Arduino program loop for the zoetrope is very simple. If the IR detector is receiving light from the emitter through the screen, the voltage on the detector analog input pin A1 will be below a certain threshold, and the program sets the strobe LED output pin D2 to high. If the strobe has been on too long (determined by counting loop iterations), it is turned off. Otherwise, the models can get blurry when the platform is spinning more slowly.

```
//-----------------------------------------------------------------------------
//
// The program running on the Arduino to make the zoetrope work. The basics:
//   - Reads infrared detector voltage on an analog pin.
//   - If the voltage is low enough (meaning it has detected IR), it activates
//     the strobe LED attached to an output pin.
//   - Turns off the strobe after a maximum duration to keep blurring from
//     occurring.
//   - Reads the voltage from a potentiometer to allow the maximum strobe
//     duration to be adjusted.
//
//-----------------------------------------------------------------------------

// Constants for various Arduino pins.
static const int kInfraRedInputPin      = A1;
static const int kPotentiometerInputPin = A4;
static const int kStrobeLedOutputPin    = 2;

// The voltage threshold for the infrared detector input. A value higher than
// this indicates the detector is not receiving IR input.
static const int  kVoltageThreshold = 980;

void setup() {
    pinMode(kInfraRedInputPin,      INPUT);
    pinMode(kPotentiometerInputPin, INPUT);
    pinMode(kStrobeLedOutputPin,    OUTPUT);

    // The IDE and TinyDuino baud rates don't seem to match; set the baud rate
    // in the IDE to half of what is set here (57600).
    Serial.begin(115200);
}

void loop() {
    static unsigned long start_micros = 0;

    // The time for a loop iteration is known by using empirical values
    // computed with a different program:
    //
    //   Time for analog  read  = 109 usecs
    //   Time for digital write =   2 usecs
    //   Time for call to loop  =   4 usecs
    // Sum:                       115 usecs per loop iteration.
    //
    // Therefore, 10 iterations is approximately 1 millisecond.

    // This counter tracks loop iterations.
```

```
    static long iterations = 1000;

    // This counter is used to keep track of how many iterations have occurred
    // since the strobe LED was turned on.
    static long iterations_on = 0;

    // This is the maximum number of iterations to leave the strobe on, based
    // on the current potentiometer reading. All the way counterclockwise reads
    // as value 1023. Update this every N iterations so that it doesn't always
    // slow down the loop.
    static long max_iterations = 1;
    if (iterations++ == 1000) {
        max_iterations = 5 + (1023 - analogRead(kPotentiometerInputPin)) / 20;
        iterations = 1;
    }

    const int ir_voltage = analogRead(kInfraRedInputPin);  // Returns 0 to 1023.
    int       strobe_state = LOW;
    if (ir_voltage > kVoltageThreshold) {
        // High voltage reading means the detector is not detecting IR.
        iterations_on = 0;
    } else {
        // IR is being detected. Turn the strobe on if within the maximum
        // number of iterations.
        if (++iterations_on <= max_iterations)
            strobe_state = HIGH;
    }
    digitalWrite(kStrobeLedOutputPin, strobe_state);
}
```

## Animated Models

So far I have experimented with only geometric models. A wrote an OpenSCAD module for morphing a ball-and-stick geometric shape into its dual. So far I have used it to create a tetrahedron morphing into its dual tetrahedron, a cube morphing into an octahedron, and a dodecahedron morphing into an icosahedron. I also created a model that is a 3D projection of a 4D hypercube (tesseract), rotating through 4D in interesting ways.

I’m hoping to get contributions from others to create interesting animations. The hard part is creating something that is interesting but that can be 3D-printed.

Update: I created some new non-geometric models for the larger stage and showed them at Maker Faire. I’ll try to add a video at some point.