RGB LED Matrix Cube with 25,000 LEDs

Created by Ruiz Brothers

https://learn.adafruit.com/rgb-led-matrix-cube-for-pi

Last updated on 2023-08-29 04:48:29 PM EDT
### Table of Contents

#### Overview
- 3D PixelDust
- An Unusual Guide
- Care and Feeding of Your Cube
- Parts from Adafruit
- Parts Sourced Elsewhere
- You Will Also Need

#### 3D Printing
- Small Cube CAD Assembly
- Big Cube CAD Assembly
- Slicing Small Cube Frames
- Slicing Bigger Cube Brackets and Mounts
- Frames for Smaller Cube
- Pi Mount for Smaller Cube
- Pi Mount for Bigger Cube
- Brackets for Bigger Cube
- Side Mounts for Bigger Cube

#### Frames for Smaller Panels
- Marking Labels on Frames
- Highlight Labels
- Magnets for frames
- Install Magnets
- Magnet Polarities
- Force Fitting Magnets
- Continue Magnets on Remaining Frames
- Removing Stock Frames
- Removed Stock Frame
- Replacing frames on panel
- Frame Label Reference Sheet
- Secure new frames to panel
- Assembled Panels

#### Pi Mount for Smaller Cube
- Pi Mount Prep
- Test Fit Mount
- Hardware for LIS3DH
- Secure LIS3DH to Pi Mount
- Install Standoffs to Raspberry Pi
- Install RGB Matrix Bonnet
- Secured Pi
- Cables for Pi
- Install Power Wires to RGB Matrix Bonnet
- Connect Cables to Pi

#### Frames for Bigger Cube
- Labeling Panels
- Magnets for Brackets
- Install Magnets to Brackets
- Brackets and Side Mounts
- Bracket Sets

©Adafruit Industries
• Brackets Placement
• Install Brackets to Panels
• Secured Brackets
• Magnets for Side Mounts
• Install Magnets to Side Mounts
• Side Mount Set
• Install Side Brackets to Panel #1
• Secure Corner Brackets
• Secure Panel 2 to Panel 1
• Installing Panel 2
• Secure Panel 2
• Installing Panel 1 to Panel 4
• Secured Panel 4
• Install Side Mounts to Panel 3
• Installing Panel 3 to Panel 4
• Secure Panel 3
• Securing Panel 3 (Continued)
• Assembled Panels
• Install Cables to Panel 5
• Install Pi to Panel 5

Pi Mount for Bigger Cube

• Hardware for Pi
• Install Standoffs to Pi
• Magnets for Pi Mount
• Test Pi Mount on Panel
• Install Magnets to Pi Mount
• Hardware for LIS3DH
• Secure LIS3DH to Pi Mount
• Install RGB Matrix Bonnet
• Secured Pi
• Cables for RGB Matrix Bonnet
• Install Power Wires to RGB Matrix Bonnet
• Connect Cables to Pi

RGB Matrix Bonnet and LIS3DH Wiring

• Jumper E8
• Jumper Wire for RGB Matrix Bonnet
• Solder Jumper Wire
• STEMMA Cable for LIS3DH
• Prep STEMMA Cable
• Solder Cable to RGB Matrix Bonnet
• Connect LIS3DH to RGB Matrix Bonnet

Power Cables

• Power Cable to Matrices
• Cut Cables
• Pairs of Cables
• Removing Extra Wires
• Removed Wire
• Set of Power Cables
• Tinning Wires
• Complete Power Cable Set
• Power Wires for RGB Matrix Bonnet
Initial Software Setup
- System Setup
- Prerequisite Software and Setup

First Light

Software Part 1
- Image Viewer
- Rotating Globe
- Conway’s Game of Life

Software Part 2

Small Cube Assembly
- Cables for Panels
- Install Cables to Panel #5
- Install Cables Through Pi Mount
- Fit Pi Mount onto Panel #5
- Connect Panel #5 to RGB Matrix Bonnet
- Panel #4 Install
- Panel #3 Install
- Panel #2 Install
- Panel #1 Install
- Panel #1 Cube Fit
- Panel #0 Install
- Power Cables
- VCC Bus
- GND Bus
- Install Buses to Cube
- Connect Battery
- Install Battery
- Final Cube Assembly
- Turning it Off

Bigger Cube Assembly
- Power and Data Cables
- Install Power Cables
- Connect Panel 4 to Panel 3
- Connect Panel 3 to Panel 2
- Connect Panel 2 to Panel 1
- Connect Panel 1 data out
- Connect Panel 4 to Panel 5
- Install Panel 5
- Power Bus
- Ground Bus
- Panel 0 Connect
- Connect Batteries
- Install Battery
- Final Cube Assembly
- Turning it Off

Auto-Start on Boot

Designing Images and Code
Field Notes

- Miscellany
- Power
- Raspberry Pi Setup
- Acrylic Box for Smaller Cube
- Presentation
Overview

One can never have too many LEDs...but we won’t let that stop us from trying. This project brings together nearly 25,000 full-color LEDs in a mesmerizing, self-contained interactive cube, with the small-and-mighty Raspberry Pi 4 — the Altoids of computers — running the show. Possibly the most ostentatious project we’ve attempted!

3D PixelDust

The flowing pixel sand demo uses an accelerometer for a mesmerizing motion effect.

The grains of sand trickle down and move across all six panels making this physics toy unlike anything else.

An Unusual Guide

IMPORTANT: We’ve assigned this guide a rare “Advanced” skill level recommendation. Not that it’s especially deep or technical, but we must admit it’s expensive. Most any day we would celebrate learning through rookie mistakes...but here, one mis-step like connecting the wrong power supply could ruin hundreds in parts. It’s best for folks who’ve cultivated patience and thoroughness through experience.

This build is also unusual in that it’s non-linear, with several steps cycling between 3D printing, soldering and wiring, and installing and testing code...it’s not simply one after the other. This is to allow testing and troubleshooting at the earliest opportunity, while everything’s out in the open, to help avoid lengthy setbacks later (see “patience and thoroughness through experience,” above). And starting with the 3D printing allows multitasking...you can get started while electronics are in transit, and do some initial board and code testing as printing gradually proceeds.

Before committing, skim through the guide to see what it all entails.
Care and Feeding of Your Cube

This is one of our most ambitious DIY projects, with a lot of big-ticket parts and fine tolerances, and you’ll want to take good care of it. The LED matrices are designed for digital signs and not really for handheld use. With rough handling, individual LEDs may get sheared off along the edges and corners. The rest of the matrix will still work, there will just be blank spots and you’ll be a little sad.

Therefore:

- Lift the cube with both hands flat against the faces, not along the edges. Gently pass it from flat palm to flat palm.
- Use something like a plastic spudger tool to carefully open the cube, do not claw at it with your fingers. A guitar pick or old credit or transit card can work just fine!
- Resist the overwhelming urge to spin the cube with fingers on opposite corners.

Parts from Adafruit

64x64 RGB LED Matrix Panel - 2mm Pitch
Wintertime can be rough in the city. The sky is gray. The weather is unpredictable. So slough off those seasonal blues with some Times Square razzle dazzle from this...
https://www.adafruit.com/product/5362

64x64 RGB LED Matrix Panel with 45 Degree Curb-Cut - 2.5mm Pitch
Wintertime can be rough in the city. The sky is gray. The weather is unpredictable. So slough off those seasonal blues with some Times Square razzle dazzle from this...
https://www.adafruit.com/product/5407
2 x Power Distribution Bus
7 x 6mm diameter solid brass
https://www.adafruit.com/product/737

1 x Adafruit RGB Matrix HAT + RTC for Raspberry Pi
Adafruit RGB Matrix HAT + RTC for Raspberry Pi
https://www.adafruit.com/product/2345

1 x USB-C to USB-A Cable
Black Woven Right Angle USB C to USB A Cable - 0.2m long
https://www.adafruit.com/product/5045

Adafruit RGB Matrix Bonnet for Raspberry Pi
You can now create a dazzling display with your Raspberry Pi with the Adafruit RGB Matrix Bonnet. These boards plug into your Pi and makes it super...
https://www.adafruit.com/product/3211

Raspberry Pi 4 Model B - 2 GB RAM
The Raspberry Pi 4 Model B is the newest Raspberry Pi computer made, and the Pi Foundation knows you can always make a good thing
https://www.adafruit.com/product/4292

Adafruit LIS3DH Triple-Axis Accelerometer (+-2g/4g/8g/16g)
The LIS3DH is a very popular low power triple-axis accelerometer. It's low-cost, but has just about every 'extra' you'd want in...
https://www.adafruit.com/product/2809
Male Headers Cable
150mm Long
1 x STEMMA QT / Qwiic JST SH 4-pin to Premium Male Headers Cable
150mm Long
1 x M2.5 Black Nylon Hardware Kit
M2.5 Black Nylon Hardware Kit
1 x M3 Black Nylon Hardware Kit
M3 Black Nylon Hardware Kit
1 x Neodymium Magnets
1/4in diameter x 1/8in thick 1D42 – Quantity x48
2 x 5V 3A USB Battery
10000mAh USB Battery
1 x USB-A to DC Cable
USB to Barrel Jack 5V DC Plug 2.1 mm Power Cable

Parts Sourced Elsewhere

You Will Also Need

- Soldering iron and related paraphernalia. There’s very little soldering required, but non-zero.
- Basic maker tools like pliers and small screwdrivers (for DC power screw terminals).
- A plastic “spudger” tool for prying open the assembled cube, or a makeshift equivalent (guitar pick, old credit card), perhaps sanding one edge to better feed into narrow slots.

3D Printing

There are two different designs for this project: a “smaller cube” for our 2mm pitch 64x64 matrices, or “bigger cube” for 2.5mm matrices with curb-cut frames. The 2mm design is particularly challenging, not just for its size, but because the stock frames must be carefully removed from the matrices and replaced with our 3D-printed design. Skim through the steps and take some time to reflect on your personal degree of patience and kit-building skill.
Once you choose a design and have parts ordered, you can start on the 3D printing, which might take several days to complete. The next few pages discuss the 3D parts... but if the printer’s still working, you can skip ahead to get started on the electronics, then return to this section later.

**Small Cube CAD Assembly**
The stock frames are removed from the 2mm pitch RGB panels and replaced with 3D printed counter parts.

Each frame requires 8x magnets.

The Raspberry Pi 4 and LIS3DH breakout are secured to the Pi Mount. The Pi mount uses magnets to attach to the panels.

**Big Cube CAD Assembly**
The 2.5mm RGB panels feature curb cut frames with 45 degree bezels. Four of the six panels are secured together using side mounts. The remaining two panels use brackets with magnets and are detachable from the four panels.

The Raspberry Pi and LIS3DH breakout are secured to the Pi mount. The Pi mount uses magnets to attach to the panels.

Downloads STLs for Cubes

Cube_Source_Files.zip
Slicing Small Cube Frames
Use PLA filament that is tested and proven to produce strong parts. Reference the following slice settings for best results.

220C Nozzle Temperature
60C Heated Bed
0.4mm Line Width
2x Wall Line Count
10% Infill Gyroid
6x Top/Bottom Layers

Slicing Bigger Cube Brackets and Mounts
Use PLA filament that is tested and proven to produce strong parts. Reference the following slice settings for best results.

220C Nozzle Temperature
60C Heated Bed
0.4mm Line Width
2x Wall Line Count
10% Infill Gyroid
6x Top/Bottom Layers
Frames for Smaller Cube
Each frame features labels with letters and numbers.
The numbers represent the panel number.

Pi Mount for Smaller Cube
The Pi Mount for the smaller cuber features mounting holes sized for M2.5 hardware and 1/4in diameter 1/8in thick magnets.

Pi Mount for Bigger Cube
The Pi Mount for the bigger cuber features mounting holes sized for M2.5 hardware and 1/4in diameter 1/8in thick magnets.
Brackets for Bigger Cube
These brackets are secured to Panel #5 and Panel #0 only using M3 hardware screws.

Each bracket uses 3x magnets.

Print 8x of these brackets.

Side Mounts for Bigger Cube
These side mounts are secured to Panels 1, 2, 3 and 4 using M3 hardware screws.

Each side mount uses 2x magnets.

Print 8x of these side mounts.

Frames for Smaller Panels

Marking Labels on Frames
The labels are embossed (raised) on the surface of each panel.

To make them more visible, we suggest coloring the labels with a sharpie will make them standout.
Highlight Labels
Carefully mark each label by lightly marking the raised portion of the labels.

Magnets for frames
Grab the 8x neodymium magnets to get them ready for installing onto the frames.

Install Magnets
Begin installing the magnets to the tabs on the frames.

Each magnet is firmly press fitted into the cavities on each tab.
Magnet Polarities
Tabs with magnets laid flat should all be the same polarity.

Double check each tab to ensure the polarities match.

Force Fitting Magnets
Tolerances are very tight so we recommend using a pair of vise grips to assist.

Carefully use vise grips to force magnets into the tabs.

Continue Magnets on Remaining Frames
Take a moment to test polarity on each tab.

Ensure magnets are tightly installed and do not pop out by testing against other magnets.

Proceed to install magnets on each of the six frames.

With luck, by this point you'll already have the electronic components in hand. Before proceeding, it’s extremely prudent to test all the matrices in their just-arrived state, before they’re built into anything. This makes troubleshooting and replacement much easier. If you haven’t already, skip ahead to the “RGB Matrix Bonnet” through “Initial Software Setup” pages, then return here once all the electronics have been tested.
Removing Stock Frames
Use a screw driver to remove stock frames some the six panels.

Take a moment to locate the 12 screws across the centers and corners of the panels.

Removed Stock Frame
Remove each screw before removing the frame from the panel.

Carefully detach the frame from the panels PCB ensuring all of the screws have been removed.

Proceed to remove the stock frames from the rest of the panels.

Save the screws!! They will be reused in the assembly. We suggest storing screws in a small plastic bag.

Replacing frames on panel
Get the 3D printed frames ready to install onto the panels.

Take a moment to orient the panel correctly with the 3D printed frame.

Use the data and orientation arrows to reference correct placement.
Secure new frames to panel
Place the frame over the panel and line up with corner mounting holes.
Carefully fasten the screws into the four corners.

Assembled Panels
Double check the 3D printed frames have been secured to the panels PCB.
Proceed to install the 3D printed frames on the rest of the panels.
Pi Mount for Smaller Cube

Pi Mount Prep
The 3D printed Pi mount uses 4x magnets to attach to panel #5.

Get the magnets and panel #5 ready for installation.

Test Fit Mount
The Pi mount attaches to the flat tabs on panel #5.

Place the Pi mount over the panel and use the magnets to determine the correct polarity needed for installing.

Install 4x magnets onto the tabs of the Pi mount.

The LIS3DH accelerometer is optional and only needed for the “sand” program. If you only plan on running the globe and Game of Life programs, this part can be omitted and you can skip a couple of steps until you get to the Raspberry Pi mounting instructions further down.

Hardware for LIS3DH
M2.5 hardware is used to secure the LIS3DH accelerometer to the Pi mount.

Place the LIS3DH over the Pi mount and line up the mounting holes.

Double check the orientation and placement is the same as the photo.
TEMPORARY NOTE: PLEASE SCREAM AT THEM TO GET THE ACCELEROMETER ORIENTATION RIGHT. If it’s flipped 180°, sand won’t work right and they’ll have to tear it down and rearrange this part.

Secure LIS3DH to Pi Mount
Use the following hardware to secure the LIS3DH accelerometer to the Pi mount.

- 4x M2.5 x 10mm screws
- 4x M2.5 hex nuts

Install Standoffs to Raspberry Pi
Secure 4x M2.5 x 10mm long standoffs with 4x M2.5 x 6mm long screws.

Install RGB Matrix Bonnet
Install the RGB Matrix bonnet onto the GPIO headers of the Raspberry Pi.

Get the Pi ready to secure to the 3D printed mount.
Secured Pi
Place the Pi 4 over the mount and line up the mounting holes.

Reference the photo for correct placement.

Install and fasten 4x M2.5 x 6mm long screws to secure the Pi to the mount.

Cables for Pi
Get the following cables ready to connect to the Raspberry Pi.

1x USB type A to USB-C short cable
1x USB type A to 2.1mm DC jack cable
2x Wires for RGB Matrix Bonnet

Install Power Wires to RGB Matrix Bonnet
Insert the two wires into the screw block terminals on the RGB Matrix Bonnet. Red wire to voltage, black wire to ground.

Use a screw driver with flat head bit to secure the wires to the screw block terminals.
Connect Cables to Pi
Plug in the USB-C cable to the USB-C connector on the Raspberry Pi.

Plug in the 2.1mm DC jack cable to the DC jack on the RGB Matrix Bonnet.

Frames for Bigger Cube

Labeling Panels
We strongly recommend adding labels to the edges of each panel.

Use a label printer, silver marker or whatever method you prefer. Ideally the labeling is visually identifiable and won’t wash or rub off easily.

We suggest adding both alpha and numeric labels to each edge.

The six matrices should be labeled as follows, as seen from the back:

Double-check your work when finished. There should be exactly two of each letter A–L. Two A’s, two B’s and so forth. Getting the numbers and letter positions right is vital to correct assembly later.
Magnets for Brackets
Get magnets ready to install into the brackets.

A total of 24x magnets are needed for 8x brackets.

Install Magnets to Brackets
Begin press fitting magnets into the flat cavities on each bracket.

Polarity of the magnets on the flat tabs should be all the same.

Begin press fitting magnets into the vertical tabs on each bracket.

Polarity of vertical tabs should be all the same.
Brackets and Side Mounts

The magnets on the brackets should attach to the corner mounts.

Take a moment to test the polarity between the two parts.
Bracket Sets
Proceed to install magnets onto all 8x brackets.

Double check each tab and magnet to ensure polarity is consistent.

Brackets Placement
The brackets are secured to the two heat set inserts on the corners of each panel.

Place a bracket over the heat set inserts and line up the mounting holes.
Install Brackets to Panels
Use M3 x 6mm long screws to secure the brackets to the panels.

Secured Brackets
Proceed to install 4x brackets to Panel #5 and Panel #0.
Magnets for Side Mounts
A total of 16x magnets are needed for 8x side mounts.

8x side mounts are secured to panels #1, #2, #3, and #4 only.

Install Magnets to Side Mounts
Press fit 2x magnets into each side mount.

Double check the side mount properly attaches to the brackets on panel #5 and #0.

Side Mount Set
Proceed to install magnets to all 8x side mounts.
Install Side Brackets to Panel #1
Get panel #1 and begin placing the side mounts onto the frames.
Reference the photo for correct placement.

Secure Corner Brackets
Use M3 x 6mm long screws to secure the side mounts to the frame.

Secure Panel 2 to Panel 1
Get Panel #2 ready to secure to Panel #1.
Match the labels on the edges. In the photo, the edge with the labels "K" will be joined together.
Installing Panel 2
Lay panel #2 flat on your work surface.

Hold Panel #1 over Panel #2 and line up the mounting holes.

Ensure the edges "K" are perpendicularly fitting together.

Secure Panel 2
Insert and fasten M3 x 6mm long screws to secure the side mounts to the panel.

Installing Panel 1 to Panel 4
Get Panel #4 ready to secure to Panel #1.

Orient the two panels so the edges with the label "E" are joined together.
Secured Panel 4
Use M3 x 6mm long screws to secure Panel #4 to Panel #1.

Install Side Mounts to Panel 3
Get Panel #3 ready to secure the remaining side mounts.

Use M3 x 6mm long screws to secure the side mounts to the panel.

Installing Panel 3 to Panel 4
Lay Panel #3 flat on your work surface.

Bring the 3x panel assembly over to Panel #3 and match the edge with the label "G".
Secure Panel 3
Place the panel assembly over Panel #3 and line up the mounting holes.

Securing Panel 3 (Continued)
Use M3x6mm long screws to secure the side mounts to the panel.
Assembled Panels
Take a moment to inspect the four panel assembly are secured correctly.

Install Cables to Panel 5
Get the IDC and power cables ready.

Connect two IDC cables to the data in and out connectors.

Connect the power cable to the power port on the panel.

Install Pi to Panel 5
Place the Pi mount over the panel with the magnets attaching to the brackets.

Double check the various USB cables are connected to the Pi.
Pi Mount for Bigger Cube

Hardware for Pi
Use the following hardware for the Pi 4
4x M3 x 10mm FF standoffs
8x M3 x 6mm screws

Install Standoffs to Pi
Insert the M3 screws through the top of the Pi mounting holes and secure the standoffs to the bottom of the PCB.

Proceed to install standoffs on all four mounting holes.

Magnets for Pi Mount
4x magnets are needed to attach the Pi mount to panel #5.
Test Pi Mount on Panel
Take a moment to test fitting the Pi mount by placing it over the brackets on Panel #5.

Check the magnets to ensure the polarity matches in order to correctly attach to the panel.

Install Magnets to Pi Mount
Press fit the magnets into the cavities on the Pi mount.

The LIS3DH accelerometer is optional and only needed for the “sand” program. If you only plan on running the globe and Game of Life programs, this part can be omitted and you can skip a couple of steps until you get to the Raspberry Pi mounting instructions further down.

Hardware for LIS3DH
M2.5 hardware is used to secure the LIS3DH accelerometer to the Pi mount.

Place the LIS3DH over the Pi mount and line up the mounting holes.

Double check the orientation and placement is the same as the photo.

TEMPORARY NOTE: PLEASE SCREAM AT THEM TO GET THE ACCELEROMETER ORIENTATION RIGHT. If it's flipped 180°, sand won’t work right and they’ll have to tear it down and rearrange this part.
Secure LIS3DH to Pi Mount
Use the following hardware to secure the LIS3DH accelerometer to the Pi mount.

4x M2.5 x 10mm screws
4x M2.5 hex nuts

Install RGB Matrix Bonnet
Install the RGB Matrix bonnet onto the GPIO headers of the Raspberry Pi.

Get the Pi ready to secure to the 3D printed mount.

Secured Pi
Place the Pi 4 over the mount and line up the mounting holes.

Reference the photo for correct placement.

Install and fasten 4x M2.5 x 6mm long screws to secure the Pi to the mount.
Cables for RGB Matrix Bonnet
Get the 2x wires ready to install to the RGB matrix bonnet.

Install Power Wires to RGB Matrix Bonnet
Insert the two wires into the screw block terminals on the RGB Matrix Bonnet. Red wire to voltage, black wire to ground.

Use a screw driver with flat head bit to secure the wires to the screw block terminals.

Connect Cables to Pi
Plug in the USB-C cable to the USB-C connector on the Raspberry Pi.

Plug in the 2.1mm DC jack cable to the DC jack on the RGB Matrix Bonnet.
RGB Matrix Bonnet and LIS3DH Wiring

Jumper E8
Add a fair amount of solder to bridge the E and 8 jumper pads on the back of the RGB matrix bonnet.

This is necessary for the RGB matrix bonnet to work with the 64x64 RGB matrices.

Jumper Wire for RGB Matrix Bonnet
Make a short wire to bridge two GPIO pins on the RGB Matrix Bonnet.

The wire in the photo is about 3in(76mm) in length.

Solder Jumper Wire
Connect the jumper wire to GPIO #4 and GPIO #18 on the RGB Matrix Bonnet.

The LIS3DH accelerometer is optional and only needed for the “sand” program. If you only plan on running the globe and Game of Life programs, this part can be omitted and you can skip the next few steps.
STEMMA Cable for LIS3DH
A STEMMA QT cable will make it easier to connect to the LIS3DH breakout board.

Prep STEMMA Cable
Cut off the male header pins and use wire strippers to expose the tips of each wire.

Tin the wires by adding a bit of solder. This will prevent the strands of wire from fraying.

Solder Cable to RGB Matrix Bonnet
Connect the wires from the STEMMA QT cable to the following pins on the RGB Matrix Bonnet.

Red wire to VCC
Black wire to GND
Blue wire to SDA
Yellow wire to SCL
Connect LIS3DH to RGB Matrix Bonnet
Double check the wires from the STEMMA QT cable have been correctly soldered to the pins on the RGB Matrix Bonnet.

Plug in the STEMMA QT cable to the side of the LIS3DH with the on LED.

Power Cables

Power Cable to Matrices
Both panel types come with power cables that feature two connectors.

Get the cable ready to prep for cube assembly.

Cut Cables
Reduce the length of the cables by cutting off the ends.

Tip: Remove the heat shrink from the cables using a hobby knife or flush cutters.
Pairs of Cables
Both cables are used in the final assembly.
These will need to be prepped before installing.

Removing Extra Wires
To reduce the wiring, remove a black and red wire from the two cables.
Tip: Use tweezers to poke visible pins and tug on the wire to release the wire from the connector.

Removed Wire
If done correctly, the wire with jumper will easily come out of the connector.
Proceed to remove a second wire from the cable.
Ensure only a single black and red wire are remaining.
Set of Power Cables
Proceed to repeat this process for each of the panels making a total of six cables.

Tinning Wires
Use wire stripper to remove a bit of insulation for the tips of each wire.

Tin the tips of each wire by adding a bit of solder. This will help to prevent the strands of wires from fraying.

Tip: Use third helping hands () to keep wires steady while soldering.

Complete Power Cable Set
Double check each wire has been properly set up.
Power Wires for RGB Matrix Bonnet

Use the extra wires to create a set of wires for the voltage and ground power input for the RGB Matrix Bonnet.

Use a single red and black wire and proceed to the assembly.

---

**Initial Software Setup**

The next several pages will interleave software setup and final build steps. This seems odd, but makes troubleshooting much easier. It’s essential to test things in stages before the case is assembled!

Begin by getting the Raspberry Pi set up with the minimal software needed for testing. Later, after you know everything’s hooked up properly and matrices are working, additional software components will be installed.

**System Setup**

Basic Raspberry Pi setup is already covered in other guides... so if this is your first time, start there. You can do these initial steps with the Pi alone, no matrices wired up.

Pi power users may have their own setup ritual, but for most folks we recommend using Raspberry Pi Imager to create a bootable micro SD card.

Choose “Raspberry Pi OS Lite (32-bit)” as the operating system. If you require a GUI for setup, use “Desktop (32-bit)” instead... but understand that once inside the cube, the HDMI ports will be inaccessible. Avoid the 64-bit OS versions for now...these are cutting-edge and often problematic.

Click the gear icon, or press Control+Shift+X (Windows) or ⌘+Shift+X (Mac), to show advanced options. Essential system info (hostname, WiFi network, etc.) can all be
configured before the system is even booted the first time…it's so much easier this way! Tip: remember to also set the Wireless LAN country to get WiFi working.

So suppose at this point you have a Raspberry Pi that's booted and accessible via an ssh remote session...

Prerequisite Software and Setup

If you’d like to read about the Matrix Bonnet in depth, there’s a whole guide for that, but here we’ll just speed-run through the essential bits for this project.

We'll start by downloading and running a script to install the underlying LED matrix-driving library:

```bash
cd
sudo bash rgb-matrix.sh
```

The LED-matrix library is (c) Henner Zeller h.zeller@acm.org with GNU General Public License Version 2.0 http://www.gnu.org/licenses/gpl-2.0.txt

When first run, the script will explain its plans and give you the option to cancel.

Next the script asks whether you’re using an Adafruit RGB Matrix Bonnet or RGB Matrix HAT. The cube project can work with either, but the Bonnet is what we show throughout this guide.

Then a question about “quality” vs “convenience.” Choose the quality option…a long chain of six matrices requires it! You should already have the pin-4-to-18 jumper installed as explained on the “RGB Matrix Bonnet and LIS3DH Wiring” page.

The script will confirm your selections and offer one more chance to cancel.

There's a lot of software to update, download and install, so it may take up to 10 minutes or so to complete. Afterward, you'll be asked whether you want to reboot the system. You can answer “n” as there's one more small change to make first...

```bash
sudo nano /boot/cmdline.txt
```

(Substitute your editor of preference in place of nano if you like.)
This file is one continuous long line. Append `isolcpus=3` to the end of this line. It should look something like this (but maybe not exactly, don't copy and paste this whole line, items like the root partition ID may be different):

```
console=serial0,115200 console=tty1 root=PARTUUID=8ba04c84-02 rootfstype=ext4 fsck.repair=yes rootwait isolcpus=3
```

Save changes to the file and exit the editor, then initiate an orderly system shutdown:

```
sudo shutdown -h now
```

Allow about 10 seconds for this to complete, or watch the Pi’s status LED. When it stops flashing, you can disconnect power. We’ll start poking at some hardware next…

---

First Light

Testing all the component before assembly is vital. Most likely all the parts will work on the first try, and you’ll have spent 30 minutes seemingly for nothing. But at worst... you skip testing, build the whole cube, encounter trouble and then you cry as it all has to come apart. Testing is the cheapest insurance in the world! It's also an opportunity to familiarize yourself with the software and parts. Wax on, wax off.

Seat the RGB Matrix Bonnet onto the Raspberry Pi’s GPIO header, making sure all the pins are aligned, that it's not off-by-one. Connect a ribbon cable (included with each LED matrix) from the Bonnet to the INPUT connector on one matrix...

The input connector isn't always clearly labeled, and the manufacturers change up the silkscreen from time to time. Usually there will be two sets of arrows: one set points toward the top of the matrix, the other indicates the flow of data, from the INPUT to OUTPUT connector. Another way is to look for the key or notch on the header shroud. On the input connector, this will face toward the edge of the board, not toward the center.
We’re intentionally being coy about power connections here. That’s another detail that manufacturers change from time to time, and we don’t want to show specific pin positions that could ruin a matrix if taken literally. The connector socket will usually have labels for GND and VCC or +5V, and the power cable included with each matrix has color-coded wires.

Also, there are myriad ways to get power to both the Pi and Matrix. The matrix power cable could go to the screw terminals on the Bonnet with everything powered through the 5V DC jack, or you might want to power the matrix and Pi separately, as shown in the diagram above.

For the love of everything, be certain you are using a 5V power supply! A lot of 12V supplies have the exact same connector...this will destroy the matrix!

Now flip the matrix over, doubling back on the ribbon connector, so you can watch the LEDs during testing:
With the Pi booted, log into the system via ssh again.

The installer created a directory called `rpi-rgb-led-matrix`, and inside this is a subdirectory `examples-api-use` with a few programs that can be used to experiment with the matrix and confirm everything's working.

```
cd ~/rpi-rgb-led-matrix/examples-api-use
```

Here’s the command to test one matrix:

```
sudo ./demo -D 0 --led-rows=64 --led-cols=64 --led-slowdown-gpio=4
```

If everything goes well, you should see a smoothly rotating colorful square. No gaps, no flicker. Press Control+C to stop.

**I’m getting no image at all!**

- Confirm both the voltage and polarity of the power supply. 5 Volts!
- Confirm ribbon cable is connected to the INPUT socket on the LED matrix, not the output.
- Confirm Bonnet is correctly aligned on the Pi’s GPIO header, not off by one.
- Confirm the GPIO4-to-GPIO18 wire is installed on the Bonnet, on the correct pins.

If none of these fix it, swap out the ribbon cable for another (you should have six, one for each matrix). If still nothing, swap out the matrix for another.

**I’m getting an image, but it’s split in half, with black bars**

There’s a solder bridge on the back of the Matrix Bonnet that must be closed. Connect “E” to “8.” You’ll need to remove the Bonnet to access this...shut down the Pi first.

If everything works and you see the rotating square: stop the program (Control+C), disconnect power from just the matrix (keep the Pi running), and repeat this test for each of the five other matrices and their included ribbon cables. It’s important to swap both. While it’s exceedingly rare...one in thousands...very occasionally a ribbon cable is poorly crimped, and testing each one avoids heartbreak later.

If all six matrices and ribbon cables pass the test, then you can be more confident as the build progresses and gets more complicated that any problems are more likely a poor connection somewhere than an inherent defect in any part.
So! Let’s forge ahead and test all six matrices working at once:

![Diagram of six matrices connected in a chain](image)

Again, we’re being intentionally vague about power. You’ll probably tie all the matrix +5V and ground lines together using the power distribution bus bars, but we’ll leave that decision up to you.

Except for the last matrix, each OUTPUT connector now links to the INPUT of the next matrix, forming a chain. Six matrices, six ribbon cables.

Notice we’ve labeled the matrices #5 through #0 here, with #5 closest to the Pi. This starts to make more sense when the chain is flipped around and viewed from the front...

![Diagram of matrix chain flipped around](image)

It’s because these matrices work as shift registers...the first data sent down the wire makes the long trip to the end of the chain. The first bits...the first, leftmost pixels...go to matrix #0. We’ll call them 0 through 5 (rather than 1–6) because that designation is easier if you want to write your own software for this later.

Here’s a command to test a chain of six matrices (this may appear “wrapped” here, but should be entered as one continuous long line):

```bash
sudo ./scrolling-text-example --led-rows=64 --led-cols=64 --led-slowdown-gpio=4 --led-chain=6 -f ../fonts/texgyre-27.bdf Hello World!
```

“Hello World!” should scroll right-to-left down the matrices. Control+C to quit.

Message scrolls partway down the chain, but leftmost matrices remain unlit

Either one or more panels isn’t receiving power, or one or more ribbon cables are faulty. Try swapping the positions of two ribbon cables and see if the problem moves with it. If so, you can try fixing the crimp (a bench vise works, applying uniform pressure across the whole row)...if still not working, request a replacement in the [forums](https://forums.adafruit.com/).
Now that the Pi, essential software, and all six matrices and cables are known good and working...don't dismantle anything yet, but...

- Pat yourself on the back for your patience and maturity to perform this full diagnostic step.
- Celebrate, take a short break, maybe have a treat.
- Next we'll install some software of our own, then move ahead with folding this up into a cube!

## Software Part 1

As long as you've got all the parts out for testing, it's time to install and try some software designed for specifically the cube. It will look all strange and discontiguous when laid out flat, but that's okay, at least you know it runs.

Later, once the cube is assembled, you can return to this page, skipping the install steps and just running the code.

There's one extra program that's a little more advanced... install that one later.

Enter these commands to fetch and build the cube demos:

```
cd
sudo apt-get install -y libjpeg-dev subversion
svn export https://github.com/adafruit/Adafruit_Learning_System_Guides/trunk/Pi_Matrix_Cube
cd Pi_Matrix_Cube
make
```

There are three programs here, the main ones implemented in both C++ and Python. Feature-wise they're identical, though the C++ versions animate more smoothly, such is the nature of these things. Python is generally quick to develop and test...so if writing your own projects, those versions might be best to examine as a starting point. The first of these programs, a simple static image viewer, is Python only...

### Image Viewer

This is a simple program that just displays an image across all the matrices. Sometimes that's all you need. It can also be helpful when verifying an assembled
cube is “spatially coherent” — the code and physical build are in agreement over the placement and orientation of each face. This accepts one argument, an image filename (most common formats are supported):

```
sudo python imageviewer.py images/minecraft.png
```

Press Control+C to stop.

**Help! My assembled cube is NOT coherent!**

The 3D-printed frames we designed make wrong assembly very difficult…the correct positions and orientations should all be fairly obvious. If assembling your own cube though, sometimes mistakes are made. There’s two ways to address this...

1. Try to compensate for the error by modifying the code…there are tables early in each program that describe each faces’ relative position. Must warn though, this is not as simple as it seems and very likely doomed to failure. There are 122,880 different ways to orient the six faces, and only one is correct. Unfortunately we can not walk you through this process, it’s just too difficult to troubleshoot.

2. Partly or fully dismantle your cube and re-assemble in correct layout. As long as you follow our “line up A to A, B to B” plan exactly, it should all come together right.

**Rotating Globe**

At its simplest, just type:

```
sudo ./globe
```

Or, for the Python version:

```
sudo python globe.py
```

Either of these should show a moving image of the Earth…albeit oddly fractured when the cube is unassembled and laid flat. With the cube assembled, it’s a spinning “globe” with the north and south poles centered on the top and bottom faces, respectively.

Both versions of the code can accept the exact same arguments…so the following options apply to both.
To change the map image, use the `-i` flag and specify a JPEG image filename:

```
sudo ./globe -i maps/jupiter.jpg
```

Or, for the Python version:

```
sudo python globe.py -i maps/jupiter.jpg
```

The maps subdirectory includes a few sample images to try out.

If substituting your own, images must be JPEG format. They can be any size or aspect ratio (the code will scale or stretch as needed). Most of the examples are 1024x512 pixels, which is more than adequate for the resolution of this cube. Others, like `boing.jpg`, are intentionally very small, to appear as a pixelated checkerboard rather than a continuous texture.

The space images are from NASA and in the public domain.

The `-v` option (no parameters, just `-v`) changes the projection so the north and south poles are at two opposing corners of the cube, rather than centered on faces. Remember our warning from the opening page: **never** spin the cube with fingers on the corners...you will shear off the corner pixels.

```
-s [time in seconds] sets the spin time in seconds per revolution, e.g. `-s 4.5` will revolve once every 4.5 seconds. Positive values will revolve in the correct direction for the Earth map. Negative values (e.g. `-s -4.5`) in the opposite direction, maybe useful for text, logos or Uranus.
```

```
-a [samples] sets the per-pixel, per-axis number of samples in the texture-mapping operation. Higher numbers yield a smoother image, but lower refresh rate. Default for the C++ code is 2 (i.e. 2x2 supersampling per pixel), default for Python version is 1 (no supersampling, because Python isn’t as quick).
```

```
-t [seconds] limits the program run time, after which it exits. Default is to run indefinitely (stopped with Control+C). Combined with a simple shell script, the time limit allows a simple slide show of different textures. You can try this with:
```

```
sudo sh ./cycle.sh
```

```
-f [seconds] sets a fade in and fade out time, where the image will transition from and to black at the start and finish. The cycle.sh script uses this together with the `-t` argument to make the image changes look smooth and intentional (otherwise there would be a quick flash to black while loading the next image).
```
-b [percent, 0 to 100] sets the peak brightness of the LEDs, if you want less than the maximum. They can be too much in some settings, or you might want to economize on power. Default is 100, full brightness.

The custom 3D-printed frames make for very narrow bezels...the gaps between matrices...about twice the thickness of the PCBs. But perhaps you've built up a cube “manually,” with the original frames still intact, resulting in more prominent bezels. Two arguments, -e [size] and -E [size] (yes, same letter, but lower and upper case), tune the mapping algorithm to compensate for these, so you don’t see a sudden jump as textures move between matrices. -e is the edge-to-edge dimension of one matrix...for example, on the Adafruit 2mm pitch 64x64 matrices, that’s about 128 millimeters. -E is the edge-to-edge measure of opposite faces of the cube...that might be just a slightly larger 134 mm with our custom frames, or 158 if the original frames are used. These can be any units...millimeters, inches, all good...it’s the ratio between them that actually matters. Honestly though, this is all quite esoteric and nobody will notice if you just leave them at their 1:1 ratio defaults.

Conway’s Game of Life

John Conway's Game of Life is a classic cellular automaton that produces interesting patterns from a set of simple rules (). We've devised a special version that correctly wraps around all edges of the cube!

```bash
sudo ./life
```

Or, for the Python version:

```bash
sudo python life.py
```

Although a little bit slower, the Python version does seem more enjoyable to watch. For both programs, press Control+C to stop.

As with the globe program, there are some configurables that work the same with either language:

- **-k [index]** selects a predefined color palette to use. The default, 0, is simply white or black, as in most classic Life implementations. But there's also:

  - **-k 1** A grayscale palette. Pixels slowly fade out as they “age” and “die.”

  - **-k 2** A common “heat map” look, fading from white to yellow to red to black.
Cycles through a spectrum of colors.

-t [seconds], -f [seconds] and -b [percent] work exactly the same as the globe program described above. These set the program run time, fade in/out time, and peak brightness level, respectively.

There’s one more program, if you built your cube with the optional accelerometer. This requires some extra setup and installation, so it gets described separately on the next page. Best to do this with the cube still in pieces. Otherwise, if you don’t have the accelerometer, you can proceed with cube assembly.

Software Part 2

The globe and Game of Life programs are cool, but this last one might be the best...an interactive “snow globe” that reacts to gravity.

This requires the LIS3DH accelerometer, which is an optional part. Let’s assume at this point that you do have this item, and connected it as explained on the “RGB Matrix Bonnet and LIS3DH Wiring” page.

Start by enabling the Raspberry Pi’s I2C bus and installing some diagnostic software:

```
sudo raspi-config nonint do_i2c 0
sudo apt-get install i2c-tools
sudo nano /etc/udev/rules.d/99-com.rules
```

(Substitute your editor of preference in place of nano if you like.)

Look for the following line...it’s in the first few lines of the file:

```
SUBSYSTEM=="i2c-dev", GROUP="i2c", MODE="0660"
```

Change the last digit from a “0” to a “6,” like so:

```
SUBSYSTEM=="i2c-dev", GROUP="i2c", MODE="0666"
```

This makes sure our code can read from the accelerometer later on.

Save changes to the file and quit from the editor, then reboot the system:

```
sudo reboot
```

Log back in, then enter the following command:
i2cdetect -y 1

If all goes well, you should get a response like the following:

```
0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:                         -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- 18 -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
```

That “18” there is the accelerometer on the I2C bus! This tells you the system can detect and communicate with it.

If you do not see the “18” in the program’s output: The most common culprit is having the SDA and SCL wires swapped. Double-check your connections and re-solder if needed.

Then download and compile our code for the falling sand demo. At the moment, this is using a special branch, so the commands are a little peculiar...

```
wget https://github.com/adafruit/Adafruit_PixelDust/archive/refs/heads/pb-multiplane.zip
unzip pb-multiplane.zip
cd Adafruit_PixelDust-pb-multiplane/raspberry_pi
make
```

And if that finishes, go ahead and try it!

```
sudo ./cube
```

If you’re testing on a flattened, unassembled cube…the movement of the sand won’t make much sense yet, but that’s okay. The important thing is that the accelerometer and code work! And if you do have an assembled cube, remember to turn it gently to avoid shearing off edge pixels.

As with the other cube programs, Control+C quits.

If you haven’t assembled the cube into its final form yet, now is the time to proceed with that! Do an orderly system shutdown before taking things apart:

```
sudo shutdown -h now
```
Small Cube Assembly

Cables for Panels
Get the data and power cables ready to install onto Panel #5 first.

Install Cables to Panel #5
Connect data cables to the HUB75 input and output.
Connect the power cable to the power connector on panel #5.
Orient panel #3 and panel #1 with panel #5 by matching the labels on edges of the frames.

Install Cables Through Pi Mount
Insert the wires from the power cable through the bottom of the Pi mount.
Reference image for correct placement.
Fit Pi Mount onto Panel #5
Attach Panel #3 and Panel #1 to Panel #5.

Place Pi mount over the tabs with magnets setting flush together.

Ensure USB cables and power wires are connected to the Raspberry Pi.

Connect Panel #5 to RGB Matrix Bonnet
Plug in the input data cable from Panel #5 to the HUB75 port on the RGB Matrix Bonnet.

Panel #4 Install
Place Panel #4 with edges "B" matching panel #5.

Connect the power cable to Panel #4.

Plug in the data output cable from Panel #5 to the data input on Panel #4.

Connect another data cable to the panels output port.
Panel #3 Install
Carefully separate Panel #3 from Panel #5 and lay it down on your work surface.

Connect a power cable to Panel #3.

Plug in the data output cable from Panel #4 to the data input on Panel #3.

Connect another data cable to the panels output port.

Attach Panel #3 to Panel #5 once again.

Panel #2 Install
Place Panel #2 down next to Panel #5 and match edges "C".

Connect a power cable to Panel #2.

Plug in the data output cable from Panel #3 to the data input on Panel #2.

Connect another data cable to the panels output port.

Attach Panel #2 to Panel #5 when ready.

Panel #1 Install
Carefully separate Panel #1 from Panel #5 and lay it down.

Connect a power cable to Panel #1.

Plug in data output cable from Panel #2 to the data input on Panel #1.

Connect another data cable to the panels output port.

Attach Panel #1 to Panel #5 once again.
Panel #1 Cube Fit
Ensure power and data cables are not being pinched when attaching Panel #1 back onto the panel assembly.

Panel #0 Install
Orient the cube assembly with Panel #1 laying down flat on your work surface.
Place Panel #0 down next to panel #1 and match edges "L".
Connect a power cable to Panel #0.
Plug in the data output from Panel #1 to the data input on Panel #0.

Power Cables
Carefully grab all of the wires from the panels power cables.
VCC Bus
Install all of the red wires from the power cables to the power distribution bus.

Use a screw driver to secure to the cables to the power distribution bus.

GND Bus
Install all of the black wires from the power cables to another power distribution bus.

Use a screw driver to secure to the cables to the power distribution bus.

Install Buses to Cube
Carefully fit the two power distribution buses into the cube assembly.

Adjust wire placement by neatly tucking in access wiring into the cube assembly.
Connect Battery
Plug in the USB-C cable from the Raspberry Pi into the 5V 3A USB Battery first.

Wait a few moments for the Pi to boot.

Then, plug in the USB cable from the RGB Matrix bonnet into the second USB port on the battery.

Raspberry Pi will automatically start up once battery is actively providing power.

Install Battery
Carefully fit the battery into the cube assembly.

Adjust wiring to better fit the USB battery.

Ensure wires are properly secured to power distribution bus when handling.

Final Cube Assembly
Carefully handle the cube assembly and orient Panel #0 with edges "J" matching. Remaining edges should be matching.

Attach Panel #0 with the rest of the cube assembly.

Congratulations on your build!!
Turning it Off

1. Connect to the Raspberry Pi over WiFi and run the shutdown command.
2. Locate panel #0 and use a thin flat prying tool to remove from cube assembly.
3. Disconnect USB cables from battery(s).

Bigger Cube Assembly

Power and Data Cables
Gather up the power and data cables for all six panels.

Install Power Cables
Connect the power cables to the six panels.

Tip: Lay the cube assembly flat to get a better handle of the connector when plugging it into the panels.
Connect Panel 4 to Panel 3
Grab a data cable and connect one end to the output of Panel #4.

Connect the other end of the data cable to the input of Panel #3.

Connect Panel 3 to Panel 2
Grab a data cable and connect one end to the output of Panel #3.

Connect the other end of the data cable to the input of Panel #2

Connect Panel 2 to Panel 1
Grab another data cable and connect one end to the output of Panel #2.

Connect the other end of the cable to the input of Panel #1.
Connect Panel 1 data out
Grab the remaining data cable and connect one end to the output of Panel #1.

Connect Panel 4 to Panel 5
Grab Panel #5 and place it next to Panel #4 with the edges "B" matching.

Connect data input cable from Panel #5 to the HUB75 port on the RGB Matrix Bonnet.

Connect data output cable from Panel #5 to data input on Panel #4.

Install Panel 5
Carefully fit Panel #5 into the cube assembly with all of the edges labels matching.

Orient the cube assembly so Panel #5 is laying face down.

 Arrange the wires from the power cables so they're all grouped and ready to secure to the power distribution buses.
Power Bus
Gather up all of the red wires and begin securing them to one of the power distribution buses.

Tip: Lay the cube assembly on its side to get better handling on wires.

Note: The power cable from Panel #0 has been removed to better reach the power distribution bus.

Ground Bus
Repeat the process above and secure all of the black wires to the second distribution bus.

Panel 0 Connect
Get Panel #0 laying face down and reconnect power cable.

Connect the data out cable from Panel #1 and plug it into data in on Panel #0.
Connect Batteries
First connect the USB power cable from the Raspberry Pi to one of the two USB batteries.

Wait a few moments for the Raspberry Pi to boot up.

Then, connect the USB power cable from the RGB matrix bonnet to the second USB battery.

Install Battery
Carefully fit the battery into the cube assembly.

Adjust wiring to better fit the USB battery.

Final Cube Assembly
Carefully handle the cube assembly and orient Panel #0 with edges "L" matching. Remaining edges should be matching as well.

Attach Panel #0 with the rest of the cube assembly.

Congratulations on your build!!

Turning it Off

1. Connect to the Raspberry Pi over WiFi and run the shutdown command.
2. Locate panel #0 (the top face) and use a thin flat prying tool, not your fingers, to remove it from cube assembly. A guitar pick or plastic card work well.
3. Disconnect USB cables from battery(s).
Auto-Start on Boot

The preferred way to do this nowadays is via the systemd service. Begin by creating a unit file for this service...

```
sudo nano /lib/systemd/system/cube.service
```

(Substitute your editor of preference in place of nano if you like.)

Copy-and-paste the following into the new file, editing the “ExecStart” line to reference which of the programs you’d like to run. Can also change the directory name, if you’ve moved or called it something else:

```
[Unit]
Description=Cube Service
After=multi-user.target

[Service]
Type=idle
ExecStart=/usr/bin/python /home/pi/Pi_Matrix_Cube/life.py -k 1

[Install]
WantedBy=multi-user.target
```

When running programs in a service like this, it’s best to specify full, absolute pathnames for everything. In the example above, that includes python and the script name. If specifying images for the image viewer or globe programs, those should have a full path to the image files too.

Save this file and exit the editor.
A few commands then enable this new service and will launch it on startup:

```
sudo systemctl daemon-reload
sudo systemctl enable cube.service
sudo reboot
```

To change which program is loaded on startup, just edit `/lib/systemd/system/cube.service` and change the ExecStart line, and reboot. No need to repeat the systemctl commands.

---

**Designing Images and Code**

If designing new images for the imageviewer.py script, take a look at the file `images/minecraft.png` for insights on how to orient things so that edges correctly wrap around.

This image is 384 pixels wide (64x6) and 64 pixels tall:

![Image of minecraft.png](imageviewer.py)  
Or here's [John Park's](https://www.johnparks.com) head, similarly formatted:

![Image of John Park's head](imageviewer.py)

If you’re wondering why each face is twisted in this peculiar order, it’s because of the way we must fold an otherwise straight chain of matrices...

```
  5  4  3  2  1  0
  4  3  2  1  0  5
  3  2  1  0  5  4
  2  1  0  5  4  3
  1  0  5  4  3  2
  0  5  4  3  2  1
```

...into a cube shape:
These orientations are not as random as they seem! We carefully iterated on this design for the 3D-printed frames to accommodate parts clearances on the 2mm pitch matrix PCBs (fitting around certain capacitors, etc.), and for fairly consistent cable distances throughout (e.g. no connections between opposite faces—the standard ribbon cable couldn’t reach if making a DIY 3mm pitch cube).

The globe program, meanwhile, uses simple UV texture maps for input...a common way of “unwrapping” a sphere into a plane, such as:

You can find lots of other examples online just by including “UV texture” in Google image search terms.

For designing new programs: if at all humanly possible, use one of the included examples as a starting point. Creating the impression of spatial coherence...where something moving off one edge appears on the correct edge of an adjacent matrix...is one of the most unpleasant things we’ve ever encountered. It’s why we designed the
matrix frames to specifically assemble only one correct way, so most users never need to face this.

“What’s in the box?”

“What Pain."

To the matrix-driving library on the Raspberry Pi, everything is just one long 384x64 pixel image, and “up” is always up. But to the code...as you can see in the images at the top of this page...every face is twisted a different way. No edge crossing leads where you would intuitively expect it. As explained above, physical constraints in the cabling and matrices themselves dictated this arrangement, and code then has to fix up the resulting mayhem.

There is no one-size-fits-all solution to this problem. You can see each of the programs takes a slightly different approach as best suits the task. There’s usually one or more tables near the top of the code trying to express the topology in that program’s context.

---

Field Notes

Here are some further insights learned in the building of one particular cube. This one was heading to a maker convention and would be seen and handled by many over the course of a day, thus needs to be extra durable and with extended run time.

Miscellany

For added robustitude, before anything else, the edge frame magnets were set in place using 5-minute epoxy. Be super extra careful that the polarity is correct...there’s no do-overs! Proceed in small batches a few at a time and keep the pieces spread
well apart. Even from a few inches away, these magnets will pull each other from their sockets! Also, “5-minutes” is the working time, but the epoxy is still pliable, so allow 30–60 minutes between groups.

Four magnets were intentionally left off the last face. This would serve as the “access door” and needed to be less grabby. It’s still very strong, and even two magnets near opposite corners might suffice.

The end plug from a spare ribbon cable was sacrificed to create a pull tab for opening the cube. This plugs into the “out” header of the last matrix in the chain — our access door.

Looked at using the Electrodragon RGB LED Matrix Panel Drive Board — which can drive three matrix chains in parallel (e.g. 2 matrices each), reducing flicker — but it hangs quite a lot over the edge of the Raspberry Pi board and wouldn’t fit inside the smaller cube. For a larger DIY cube, it’s something to consider!
Power

With some patience, careful ribbon cable folding and right-angle USB cables, it’s possible to fit two USB power banks inside the cube. Over six hours of run time is possible!

CAUTION: some USB cables (like the braided one here) have metal plug housings, which could cause a catastrophic short if it shifts and contacts the wrong part of a matrix. This will get covered with tape, or better yet replaced with a plastic-bodied cable.

Originally one battery was dedicated to the matrices, the other to the Pi. Turns out the matrices use substantially more power. Moving two matrices over to the Pi battery (via the Bonnet’s screw terminals) made for a fairly balanced load, improving run time by 50%.

Consider this just a napkin-sketch schematic…neither a precise layout nor complete bill of materials…it’s merely to illustrate how power is split from the two battery banks to the Pi and matrices. Some improvised cobbling of parts and wires is needed!
We can get away with splitting power like this because the ribbon cables along the matrix chain include several ground lines; the two circuits have a common point of reference.

To save space, instead of bus bars, a section of Perma Proto board was used to distribute power to the matrices, then sealed in heat-shrink tube. Wires were clipped extra short...just long enough to reach across the cube...but one pair was left a bit longer for the access door to swing open.

This photo is from the early “all matrices on one battery” stage, hence the large number of wires. Two matrices were later moved to the screw terminals on the Matrix Bonnet.

A second USB-to-DC-barrel cable was used, connecting to a DC-socket-to-screw terminal adapter to power four of the matrices. Heat shrink tubing and hot glue provide insulation and strain relief.

Cutting a few inches from a USB cable and connecting that to the matrix power distribution is also an option, but I didn’t feel my soldering and heat-shrink skills were up to the task.

A few strategic dots of hot melt glue keep cables in their sockets. Normally hot glue is “the wrong adhesive for any job,” but a redeeming property is easy removal later...touch it with a cotton swab dipped in alcohol, and it pops clean off.

Raspberry Pi Setup

Running in its default configuration, and seated between those LED matrices, the Raspberry Pi 4 can get incredibly hot. This leads to thermal throttling (the Pi slows way down to cool off), causing the LEDs to visibly flicker. And it’s nearly hot enough to burn fingers when reaching inside to power off. Underclocking the Pi just a little — 1.2
GHz instead of the default 1.5 — plus some LED tweaks below, are sufficient to keep it running at a steady rate, and safe to touch (warm, but not dangerously so).

These lines went in /boot/config.txt:

```text
arm_freq=1200
gpu_freq=400
```

If there’s an arm-boost line in there, comment it out!

```text
#arm_boost=1
```

These settings are specific to the Pi 4. Earlier boards aren’t likely to run so hot...or if they do, would need different clock settings. This would take some experimentation.

With the CPU throttled down, the `--led-gpio-slowdown` argument for the software can be adjusted down to keep the matrix refresh peppy.

One more step both to reduce the interior temperature and improve battery life is simply to turn down the brightness...sometimes a little, sometimes a lot. In fact, when initially testing with a single battery, even half brightness with certain globe images would lock up the Pi due to brownouts. The fade-in effect available to some of the demos...originally there just to look cool...actually helps us here by avoiding sudden surges.

Here’s a demo loop script that worked, cycling between sand, life and globes:

```bash
#!/bin/sh
set -- --led-slowdown-gpio=2 --led-pwm-bits=9 --led-brightness=80
while true; do
  timeout 20s /home/pi/Adafruit_PixelDust-pb-multiplane/raspberry_pi/cube $@
  /home/pi/Pi_Matrix_Cube/globe $@ -e 128 -E 132 -a 3 -t 15 -f 1 -i /home/pi/
  Pi_Matrix_Cube/maps/earth.jpg -v --led-brightness=60
  /home/pi/Pi_Matrix_Cube/life $@ -t 15 -f 1 -k 2
  /home/pi/adafuit/PixelDust-pb-multiplane/raspberry_pi/cube $@
  /home/pi/Pi_Matrix_Cube/globe $@ -e 128 -E 132 -a 3 -t 10 -f 1 -i /home/pi/
  Pi_Matrix_Cube/maps/boing.jpg --led-brightness=20 -s 5
  /home/pi/Pi_Matrix_Cube/life $@ -t 15 -f 1 -k 1
  /home/pi/Pi_Matrix_Cube/maps/adafruit.jpg -s -5
  done
```

This was placed in the file /home/pi/Pi_Matrix_Cube/cycle.sh

The “set” line configures a few things globally for all programs, then each one provides its own additions or overrides.
Notice the absolute paths to everything, even the image files. That’s because it’s going to run as root (not pi) on startup, using the systemd service.

This process is already explained on the Auto-Start on Boot() page...so if you’ve already done that, you just need to modify the service file. The only change here is we’re calling a shell script rather than one specific demo. If you don’t yet have that, create a service file like so:

```
sudo nano /lib/systemd/system/cube.service
```

...containing the following lines:

```
[Unit]
Description=Cube Service
After=multi-user.target

[Service]
Type=idle
ExecStart=sh /home/pi/Pi_Matrix_Cube/cycle.sh

[Install]
WantedBy=multi-user.target
```

Again, absolute path to the cycle script.

Save changes to the file, then activate the service using:

```
sudo systemctl daemon-reload
sudo systemctl enable cube.service
```

On next boot, the script should start automatically. If it doesn’t, it’s likely a typo in the file or commands.

After ample testing and tweaking, the Pi was then configured for read-only mode, as explained in this guide(). This allows simply cutting power (or battery to fully run down) without a formal system shutdown. Do this as a penultimate step after demo changes are finalized, because files can’t simply be edited afterward. That requires unwinding the read-only steps, rebooting, editing files, then reinstating read-only mode. A chore!

The last last step was to make a backup of the SD card. These things do fall out if jostled, and a ready spare might save your bacon!
Acrylic Box for Smaller Cube

These LED matrices are notoriously fragile, and one must handle an assembled cube carefully with flat palms, otherwise corner and edge pixels could get sheared off. You and I understand that, but the public at large does not.

So...a custom laser-cut acrylic box holds this cube, providing an unobstructed view while protecting those delicate edge pixels. There’s a couple millimeters clearance all around, so the LED cube can slide in without shearing off pixels. There’s nothing “clever” in the box’s design, but it does the job.

The files below are in Adobe Illustrator format, but other software (such as the free Inkscape) can also read these.

This box is for the smaller cube only, but with a little editing might accommodate the larger cube by rubber-band selecting the points along each edge and scooting them 16 millimeters outward (bigger cube is 32 mm larger on each axis). The scoot (rather than scaling) keeps all the slots and tabs aligned and the internal clearance and 3mm thickness intact.

This file is for the sides of the box. Cut four copies; they’re identical. 141×141 mm. One edge has a protruding tab that aligns with a corresponding notch on an adjacent piece.

box-sides.ai

This file is the top and bottom of the box. 141×141 mm. Laser cut two copies. One will be affixed to the four side walls, the other is left unconnected.

box-top-bottom.ai

Our box used bog standard 3mm (1/8") clear acrylic. Consider anti-glare or scratch-resistant acrylic if you want to get “fancy”. As much as we love black LED acrylic, it would be a poor choice here! There’s a necessary clearance to the box dimensions, and the resulting shifting of the cube inside would cause some faces to be blurry.

Assembly

Laser-cut, peel, rinse and fully dry all six faces. Set aside the top piece so you don’t inadvertently seal the whole thing shut. Work with just the acrylic for now; the LED cube slides in later.
Puzzle out how the remaining five pieces fit. The “+”-shaped piece is the bottom, while the four sides have keys that align one to the next. Hold everything together temporarily with rubber bands or masking tape. As you start joining pieces, you'll need to move some bands to a different direction or remove tape from one edge at a time.

Some options for joining the edges:

- Transparent sticky tape. Not the most durable but it's cheap and might suffice. Run a strip down a whole edge, trim the ends with a hobby knife, then rearrange the bands or tape to expose the next edge, one at a time.
- Acrylic cement. This is considered the Good and Proper Way to join acrylic. It's a little toxic (follow precautions on bottle) and takes patience, as each edge should be done one at a time and allowed to fully set up. Use a sable brush or tiny syringe and capillary action. You MUST clear any bands or tape away from the edge being glued, else it will seep into anything and ruin the pristine face of the cube.
- UV-curing resin. A kit with a bottle of resin, ultraviolet flashlight and UV safety glasses runs about $20. Squeeze a pea-size drop near an inside corner and patiently allow gravity to do the work, then cure for one minute using the UV flashlight before moving on to the next edge. Like the acrylic cement, keep bands and tape away from the edge being glued. Once all eight edges are done, the bands/tape can be removed and the box set out in the sun for an hour or two to ensure it's fully set up.

Carefully slide the assembled cube into the acrylic box, stopping if there's any hint of resistance. The “access door” face of the cube should align with the open face of the box. It might be easier to turn the cube over and lower the box over it.

The top face is then held on with clear sticky tape. A long piece along one edge forms a hinge, and a couple shorter pieces on the opposite edge hold it shut temporarily. More durable would be to run long strips down all four edges, understanding that it will take some work to get back inside and power it down. Bring the tape dispenser with you and keep an eye on that tape, in case it wears down with all the handling. Can also use the tape to temporarily repair glued edges if they should crack apart.

Clear rubber feet on the bottom face are nice but optional.
Presentation

“Are they heavy? ... Then they’re expensive, put ‘em back.”

— Donald Genarro, Jurassic Park

The accelerometer sand demo in particular just screams to be handled, and no passers-by are denied that pleasure. With the acrylic enclosure, the cube is reasonably well protected. There’s a little extra step we can take though...

Might carry it around with white inspection gloves. These aren’t really there to protect the cube, which will be handed around to bare-fingered folks. The gloves are for two things...

Rubbing off fingerprints so the cube’s always appealing and shiny. Without a word, it tells others “this is a treasure, please handle it kindly.” Theatrics, really.

The cube contains around $500 in parts, and that’s before the Raspberry Pi 4 got jacked up in short supply. There’s also tremendous work and love putting this together.

Not expecting it to walk away at an upstanding convention…but other times, like in a parked car, smash-and-grab crime is rising. As an experiment, trying an Apple AirTag inside the cube...these are tiny and can fit in any remaining crevice. It remains to be seen whether this can even function through the LED matrix PCBs, so this will be a Learning Experience. Since the AirTag is metal on one side, it goes in a small ziploc bag to avoid shorting anything inside the cube.