



EERO AF HEURLIN

RGB EFFECTS ON LONG LED STRIPS  
WITH NUMPY AND RASPBERRYPI

## YOURS TRULY

Electronics hobbyist since childhood

Active in Finlands hackerspace scene

Currently working at <http://anders.fi>

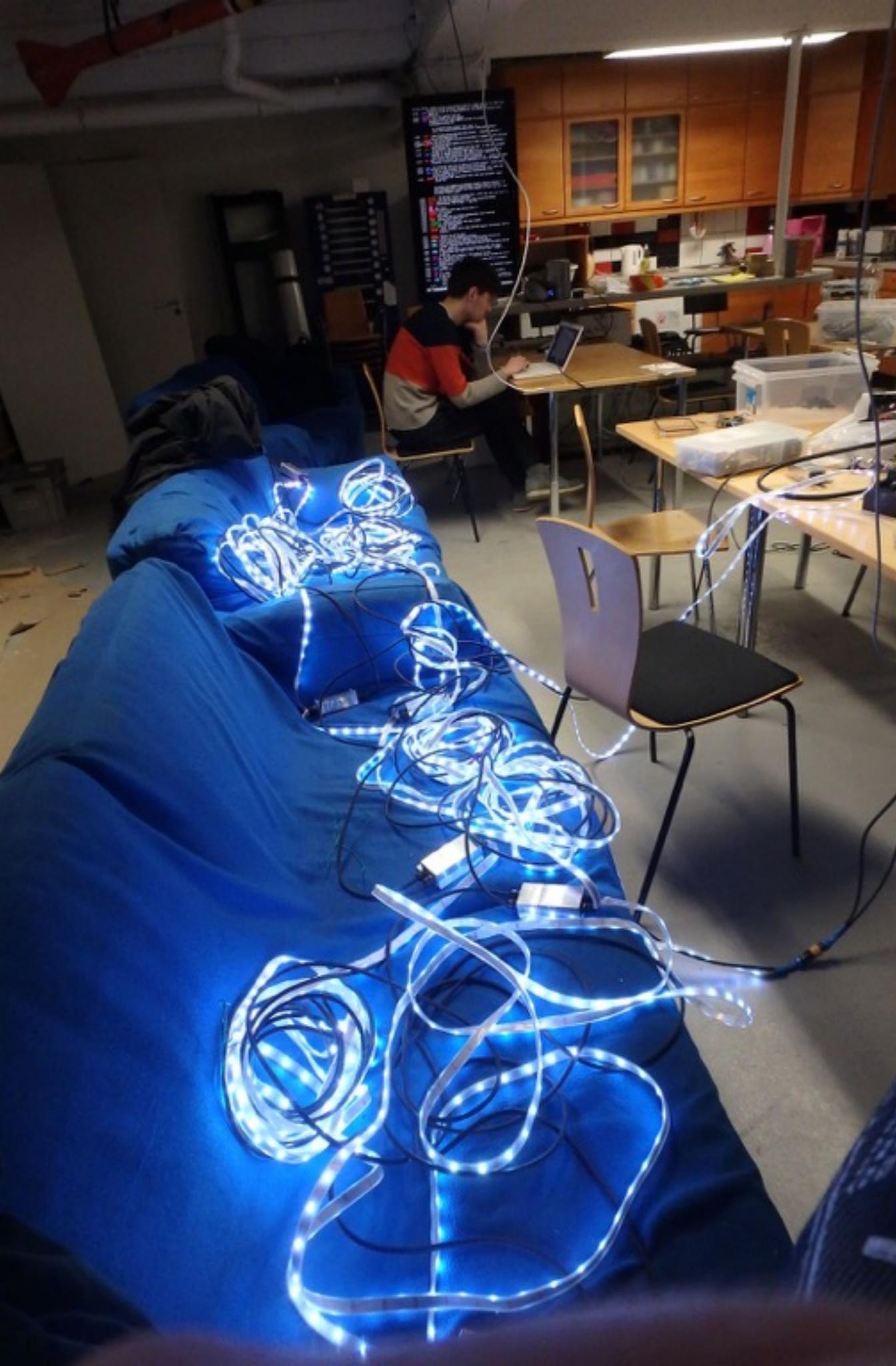
**Believer:** Open Source & Open Hardware

<https://github.com/rambo>

<http://fi.linkedin.com/in/eeroafheurlin/>



# SOME SPECS



- $50\text{m} \times 30\text{LEDs/m} = 1500\text{LEDs}$   
RGB
- in 5m strips using APA102 superLEDs with integrated controller
- SPI-like control protocol
- 5V working voltage (a problem with long strips)
- 10 point-of-load regulators with custom milled cases

# PROJECT BACKGROUND

# ARTISTS' RENDITION OF THE WALK TO HACKLAB DOOR DURING AUTUMN

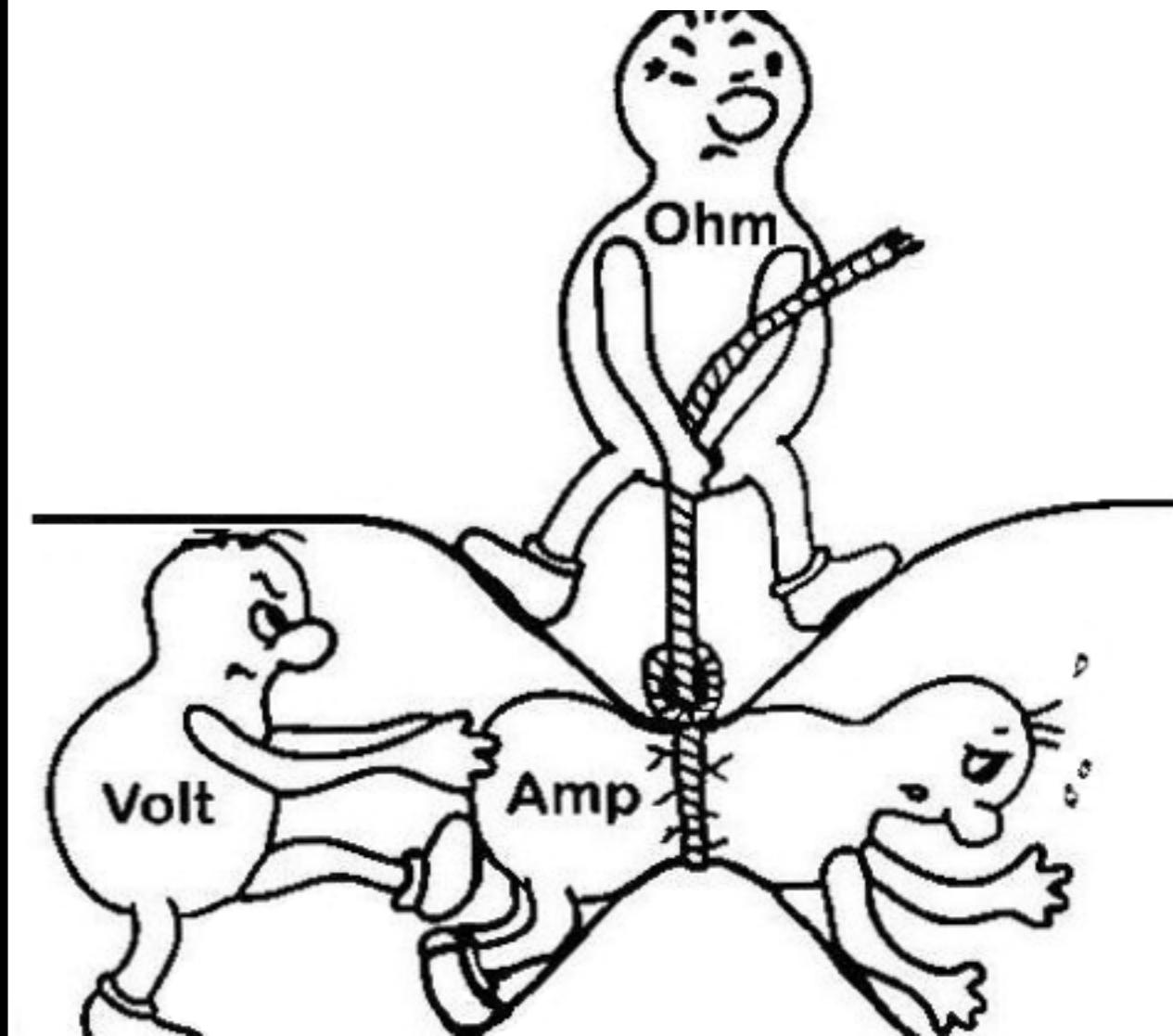
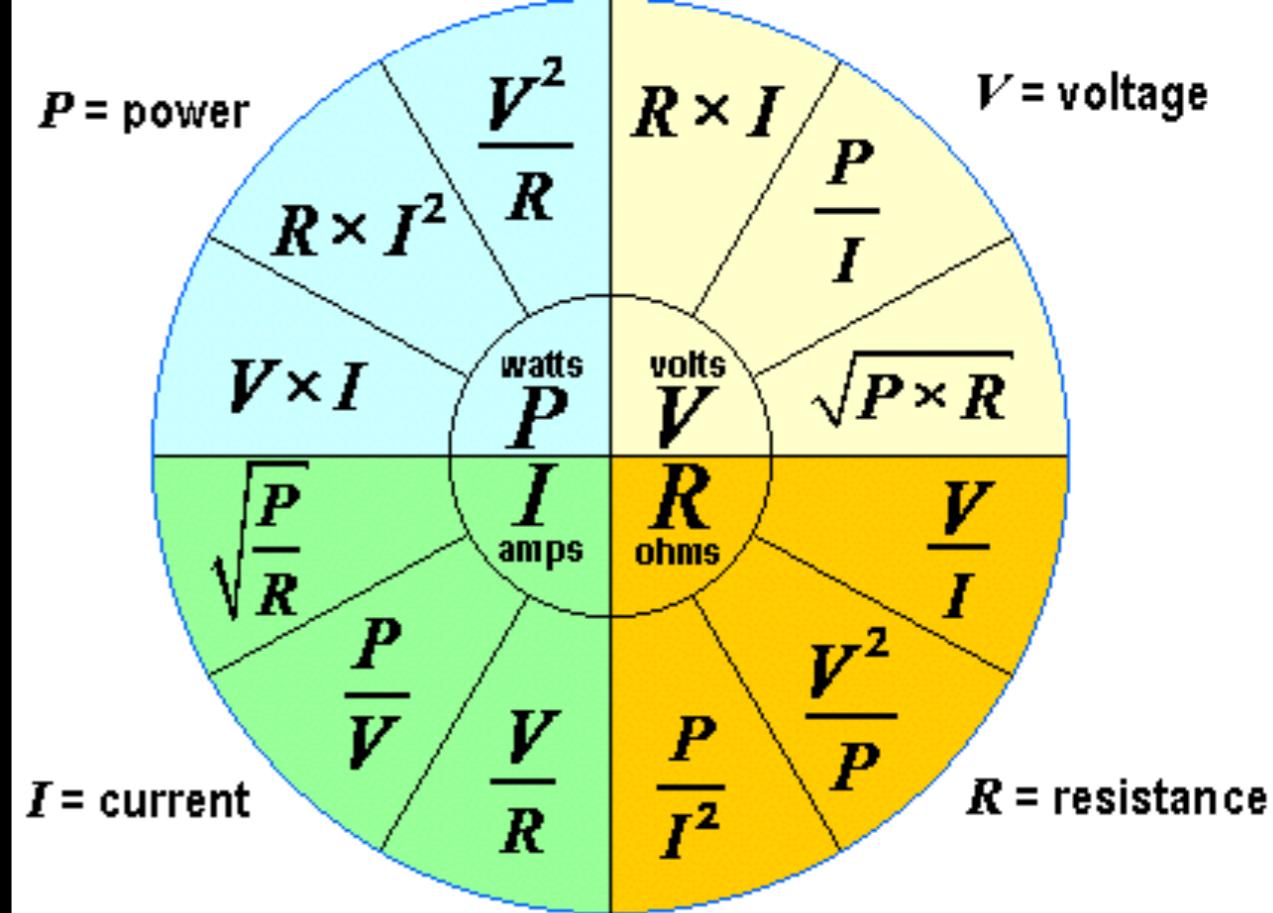


# FINDING THE LED STRIPS

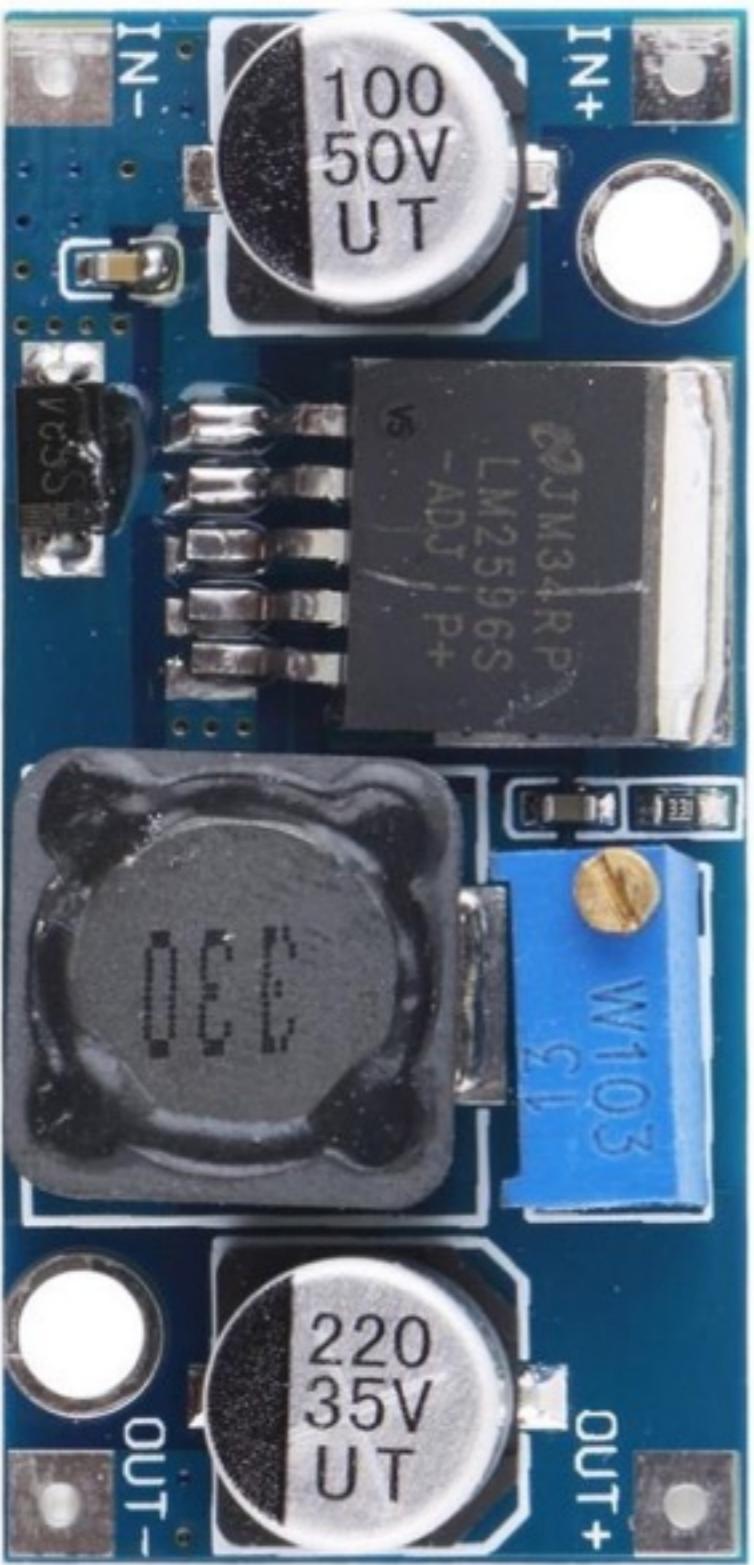
- WS2812 is no-go, the one-wire control protocol is really fiddly with timing and very sensitive to electrical noise
- APA102 is the new hotness, a bit expensive though.
- Alibaba to the rescue.
- After shipping and taxes ~400EUR for 10 \* 5m reels.

# ELECTRONICS INTERLUDE

- $P = U * I$  and  $I = R * U$
- 5V is not a good voltage to deliver significant power
- At full brightness each led takes about 30-40mA -> 45A for 1500 LEDs
- Still "only" 225W of power
- Add point-of-load regulators every 5m fed with 24V -> ~10A for 225W



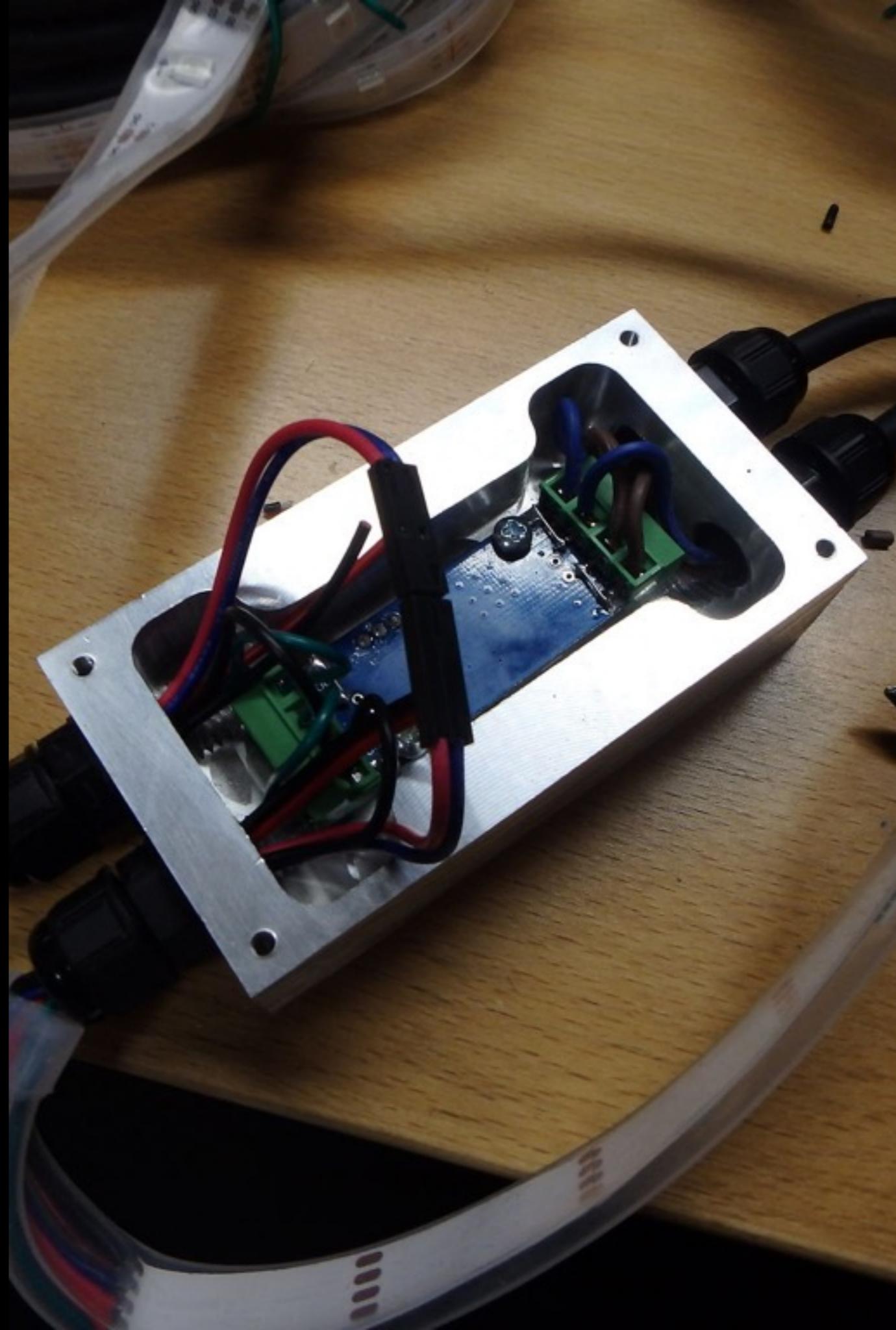
# THE REGULATORS

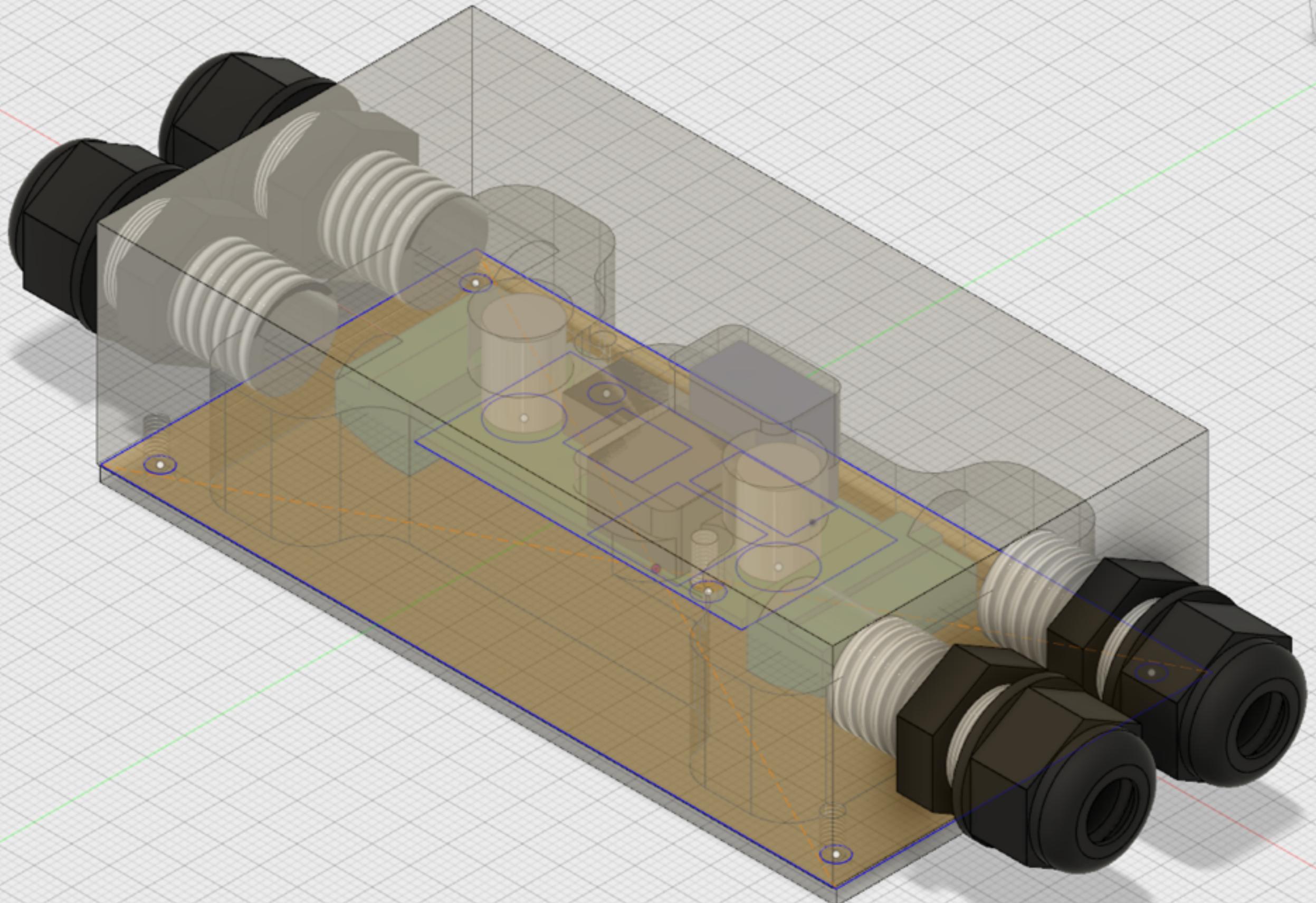


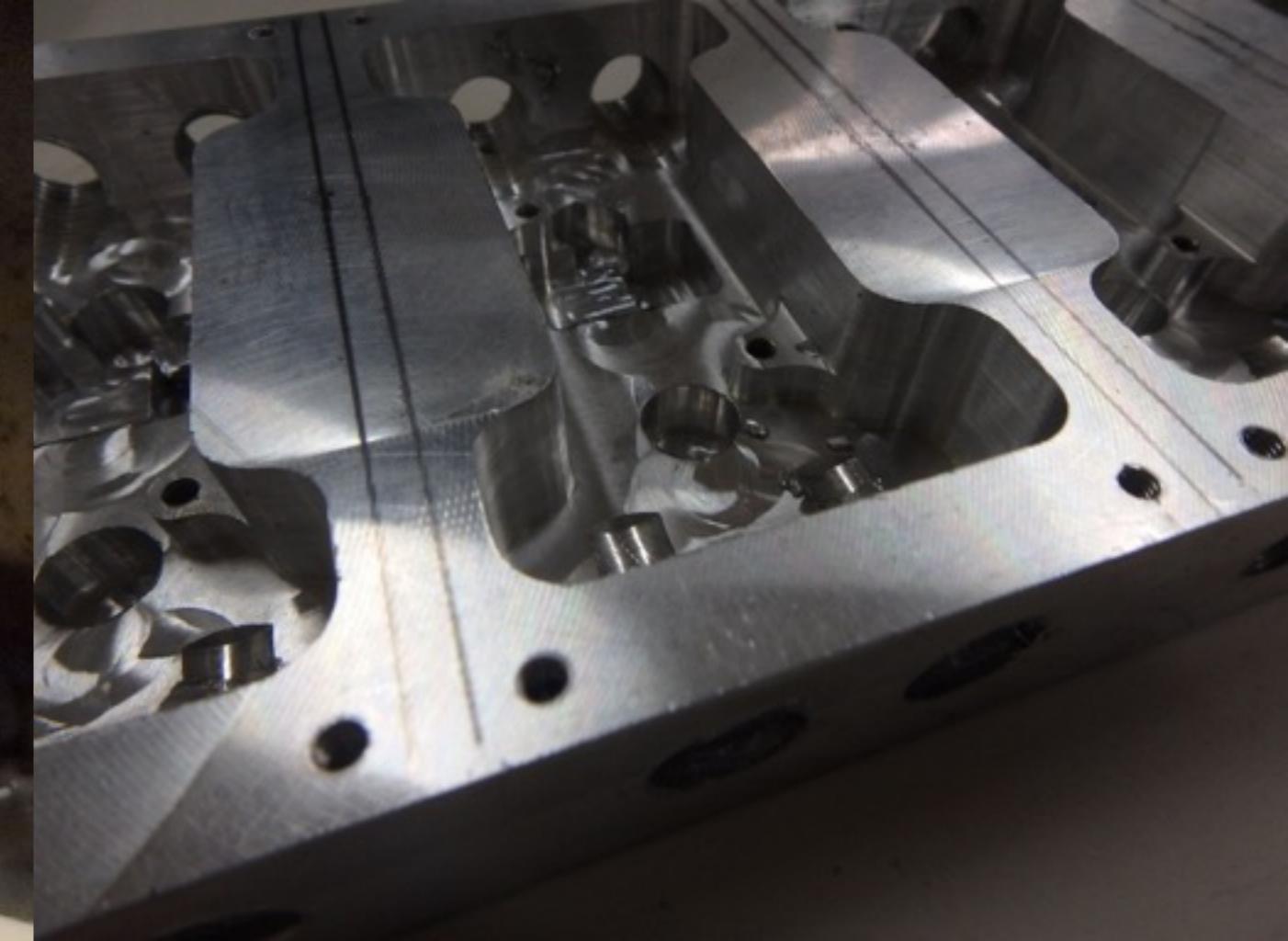
- This design (and a tens of small variations) is small, ubiquitous and cheap
- Had some in my own stock, got a lot more from Aliexpress
- Rated for 2A, some do not reach even that
- Total price maybe 10EUR
- Calibrated output on all to 5.0xxV under load with my HP6632Bs

# CASES

- Waterproof
- Acts as heatsink
- Designed in Fusion360
- Used 3D printed versions to test for parts fit
- Using cheap cable glands, proper connectors would have cost a fortune.

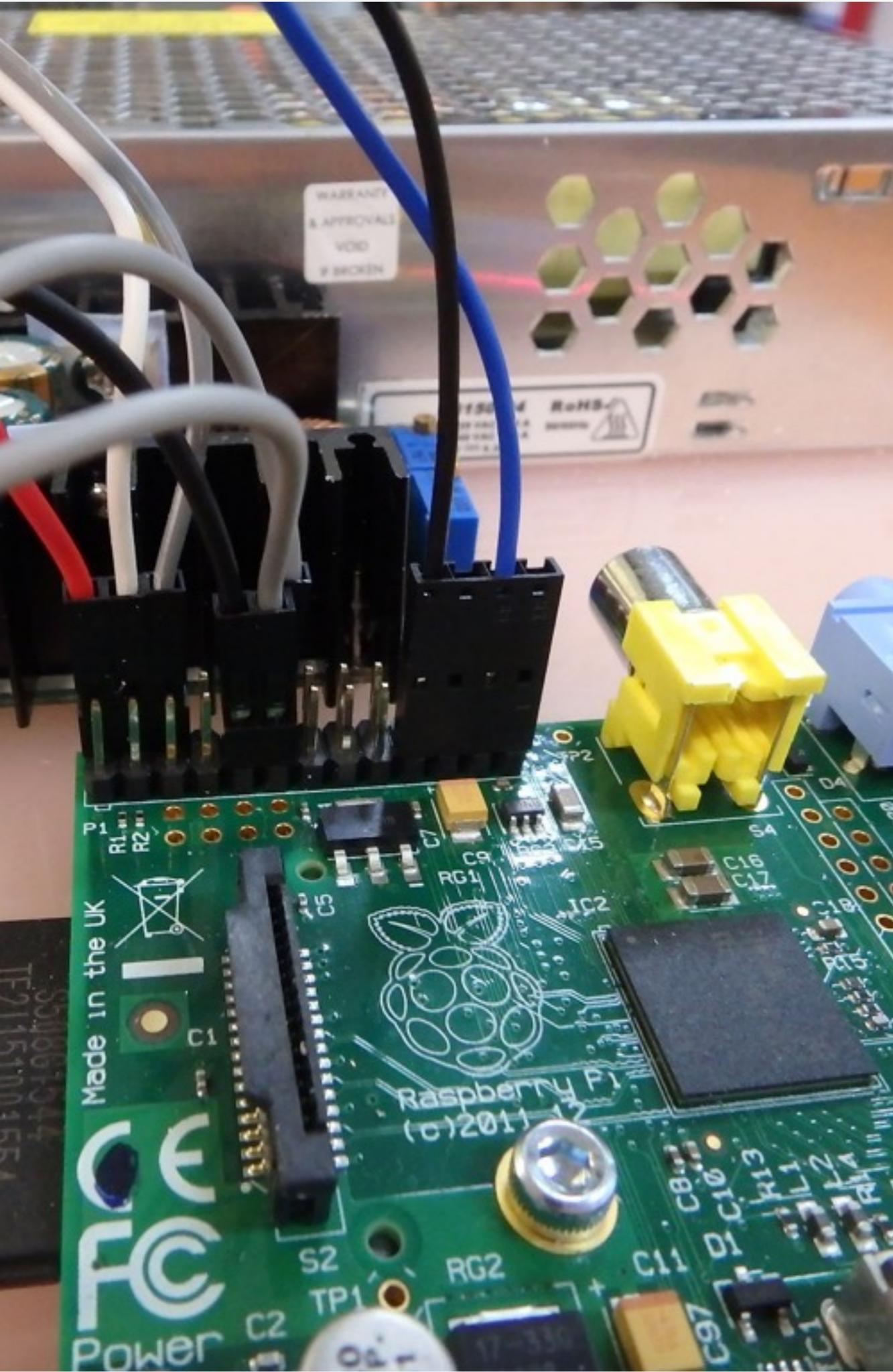






OH YEAH, PYTHON...

# GETTING SPI OUT



- RPi has usable SPI hardware
- Libraries “problematic”
- Wrote a ZMQ<->SPI bridge in C (a while back)
- But default kernel settings do not allow over 4k SPI buffer, we send over 6k...

# FIRST IMPLEMENTATION

- Very simple effect
- Naive
- Pure Python
- Way too slow on the RPi

```
# -*- coding: utf-8 -*-
from __future__ import print_function
import math
from utils import rgb_floats2bytes

def colors2frame(colors):
    """The colors param is array chrome
    # start frame
    frame = bytearray([0x0, 0x0, 0x0,
    # Leds
    for color in colors:
        frame.append(0xff) # we do not care about the order
        rgbbytes = color.rgb256
        frame.append(rgbbytes[2])
        frame.append(rgbbytes[1])
        frame.append(rgbbytes[0])
    # End frame
    eofbits = len(colors) / 2.0
    eof = bytearray([0xff] * int(math.ceil(eofbits)))
    return frame + eof
```

```

def recalculate_chase_mix(self):
    self.chasepattern = np.zeros([1 + self._tailsize, 4], np.uint8)
    # TODO: Gamma-correct ??
    rgbbbytes = self._chasecolor.rgb256
    self.chasepattern[-1] = [0xff, rgbbbytes[2], rgbbbytes[1], rgbbbytes[0]]
    for x in range(self._tailsize):
        tailcolor = chroma.Color(self._chasecolor)
        tailcolor.alpha = 1.0 / self._tailsize * x
        blended = self._basecolor + tailcolor
        rgbbbytes = blended.rgb256
        self.chasepattern[x] = [0xff, rgbbbytes[2], rgbbbytes[1], rgbbbytes[0]]


@property
def numleds(self):
    return self._numleds

@numleds.setter
def numleds(self, value):
    self._numleds = value
    self.pattern = np.zeros([value, 4], np.uint8)
    self.initialize_frame()

def initialize_frame(self):
    eofbits = self._numleds / 2.0
    eofbytes = int(math.ceil(eofbits / 8.0))
    self.frame = np.zeros(self.pattern.size + APA102_FRAME_START_SIZE + eofbytes, np.u
    self.frame[-eofbytes:] = 0xff

def next(self):
    return self.__next__()

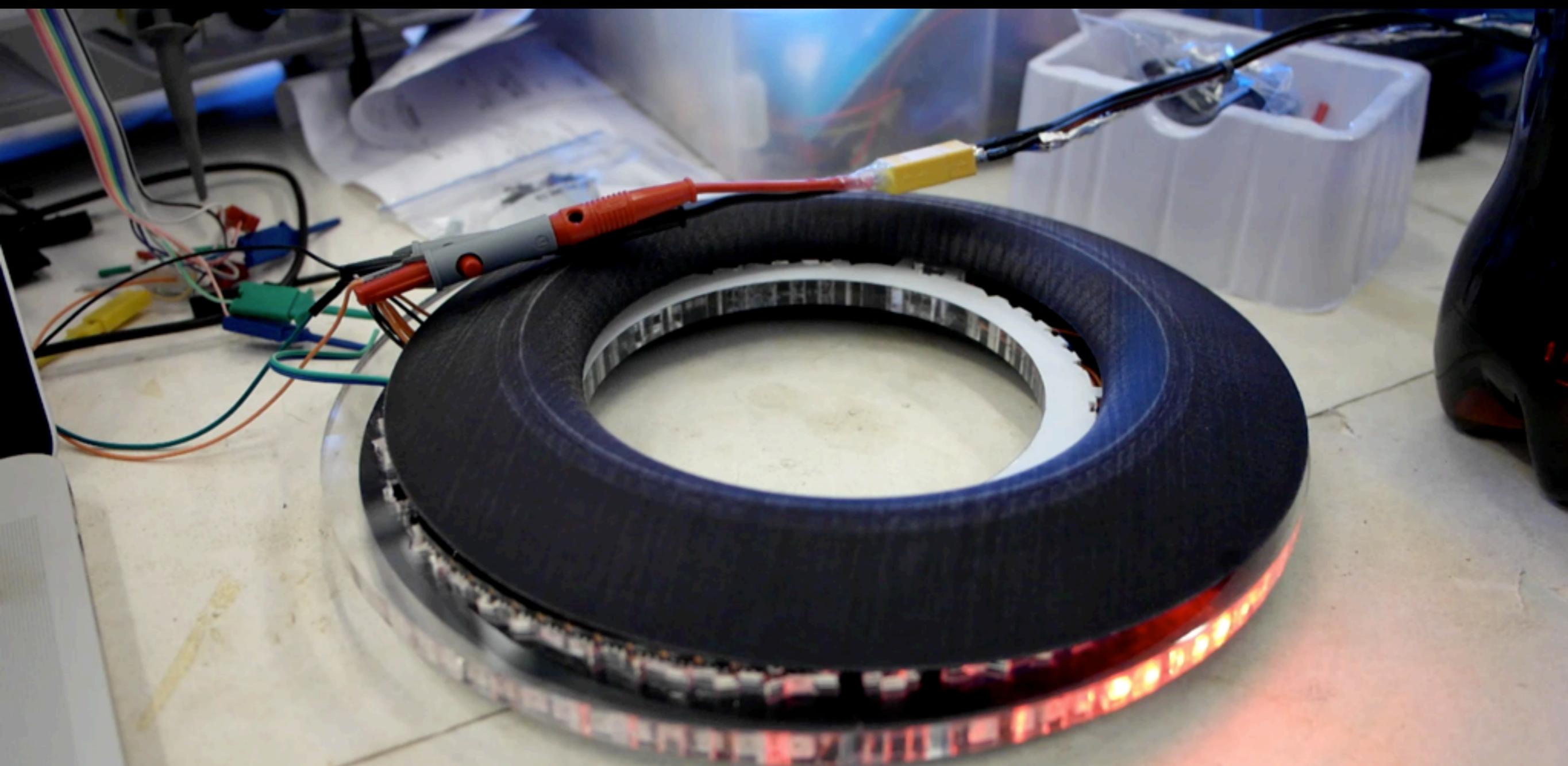
def __position_single_tail(self, chasepos):
    # TODO make this case work too.
    if chasepos < 0:
        return
    tailend = chasepos - len(self.chasepattern)
    # TODO: Gamma-correct ??
    rgbbbytes = self._basecolor.rgb256
    self.pattern[tailend - 1] = [0xff, rgbbbytes[2], rgbbbytes[1], rgbbbytes[0]]
    if tailend < 0:
        self.pattern[tailend:] = self.chasepattern[0:abs(tailend)]
        self.pattern[0:chasepos] = self.chasepattern[abs(tailend):]
    else:
        self.pattern[tailend:chasepos] = self.chasepattern

def __next__(self):
    chasepos = self.i % (self._numleds - 1)
    for x in range(self._numtails):
        self.__position_single_tail((chasepos + x * self._tailinterval) % (self._numleds

```

# GO NUMPY

- Use NumPy arrays instead of list
- Use partial array updates
- Use NumPy vector operations
- Pre-calculate color mixing



# “You mentioned a simulator”

-MEMBER OF THE AUDIENCE

The image shows two terminal windows side-by-side. The left window is titled "Simulating 150 LEDs" and displays a command-line session for a "ledchaser" project. The right window is titled "rambo@ubuntu1404dt: ~/ledchaser" and shows a similar session. Both windows show code for generating and running a LED chase pattern.

```
Simulating 150 LEDs
```

```
rambo@ubuntu1404dt: ~/ledchaser/simulator
chasegenerator.py      play.py          setup.cfg
chasegenerator.pyc    README.md        simulator
devel-requirements.txt requirements.txt start.sh
rambo@ubuntu1404dt:~/ledchaser$ cat start.sh
#!/bin/bash
source /root/.virtualenvs/ledchaser-34/bin/activate && cd /usr/src/ledchaser/ &
& nohup python play.py ipc:///tmp/zmqspi 1500 6 &
rambo@ubuntu1404dt:~/ledchaser$ d
d: command not found
rambo@ubuntu1404dt:~/ledchaser$
rambo@ubuntu1404dt:~/ledchaser$ cd simulator/
rambo@ubuntu1404dt:~/ledchaser/simulator$ ls
apa102parse.py  apa102parse.pyc  requirements.txt  simulator.py
rambo@ubuntu1404dt:~/ledchaser/simulator$ workon led
ledsim   leddsim-34
rambo@ubuntu1404dt:~/ledchaser/simulator$ workon led
ledsim   leddsim-34
rambo@ubuntu1404dt:~/ledchaser/simulator$ workon leddsim
(leddsim)rambo@ubuntu1404dt:~/ledchaser/simulator$ ./simulator.py ipc:///tmp/zmqspi 150
Running
```

```
rambo@ubuntu1404dt: ~/ledchaser
rambo@ubuntu1404dt:~$ cd ledchaser/
rambo@ubuntu1404dt:~/ledchaser$ workon leddsim
(leddsim)rambo@ubuntu1404dt:~/ledchaser$ ls
chasegenerator.py      play.py          setup.cfg
chasegenerator.pyc    README.md        simulator.py
devel-requirements.txt requirements.txt start.sh
(leddsim)rambo@ubuntu1404dt:~/ledchaser$ ./play.py
^C(leddsim)rambo@ubuntu1404dt:~/ledchaser$ vi play.py
(leddsim)rambo@ubuntu1404dt:~/ledchaser$ ./play.py
```

# SIMULATOR

- Accepts exact same frame data as is sent to SPI over ZMQ
- My first VPython application
- Doesn't really scale to hundreds of LEDs as-is.

```
lass ledscene(object):

    def __init__(self, numleds, zmqstream, spacing=33, ledradius=15):
        self.numleds = numleds
        self.stream = zmqstream
        self.stream.on_recv_stream(self.frame_recv)

        visual.scene.width = 1500
        visual.scene.height = 80
        visual.scene.autoscale = True
        visual.scene.title = "Simulating %d LEDs" % self.numleds
        self.strip = visual.frame(pos=(self.numleds / 2 * spacing * -1, 0, 0))
        self.leds = []
        for x in xrange(self.numleds):
            self.leds.append(
                led(ledradius, frame=self.strip, pos=(x * spacing, 0, 0))
            )

    def frame_recv(self, stream, msg):
        ledquads = apa102parse(np.fromstring(msg[0], dtype=np.uint8))
        i = 0
        for quad in ledquads:
            self.leds[i].color = (quad[1] / 255.0, quad[2] / 255.0, quad[3] / 255.0)
            i += 1
            if i >= self.numleds:
                break
        stream.send("")

    def run(self):
        print("Running")
        try:
            visual.scene.visible = True
            visual.scene.autoscale = False
            ioloop.IOLoop.instance().start()
        except KeyboardInterrupt:
            self.quit()

    def quit(self):
        print("Quitting")
        ioloop.IOLoop.instance().stop()
```

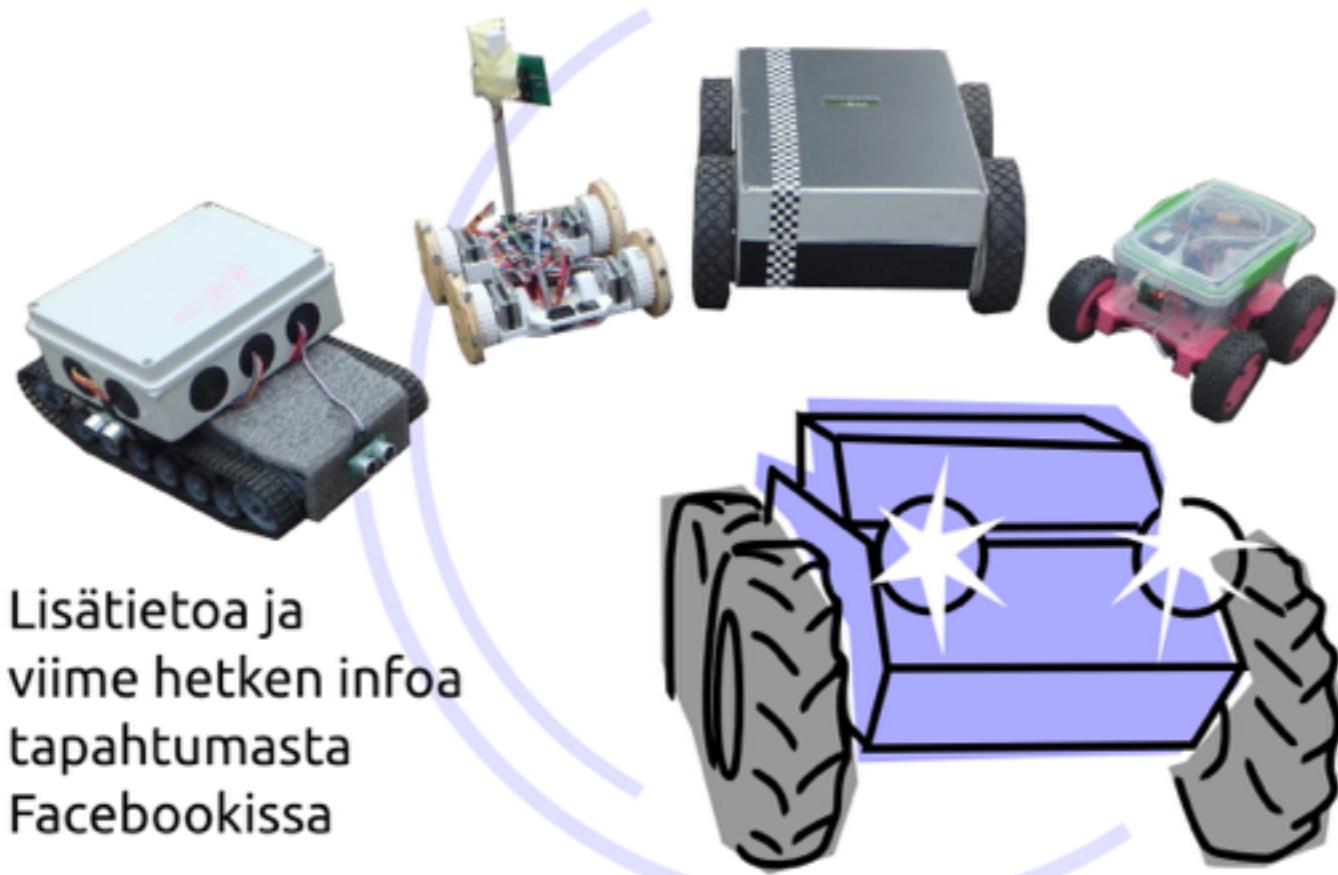
SHAMELESS PLUG

# Robotit Strömbergin puistossa

Ia 11.6. klo 16

Pienoisrobotit kohtaavat sokkeloradalla  
Strömbergin puistossa

Avoin ja maksuton tapahtuma kaikenikäisille!

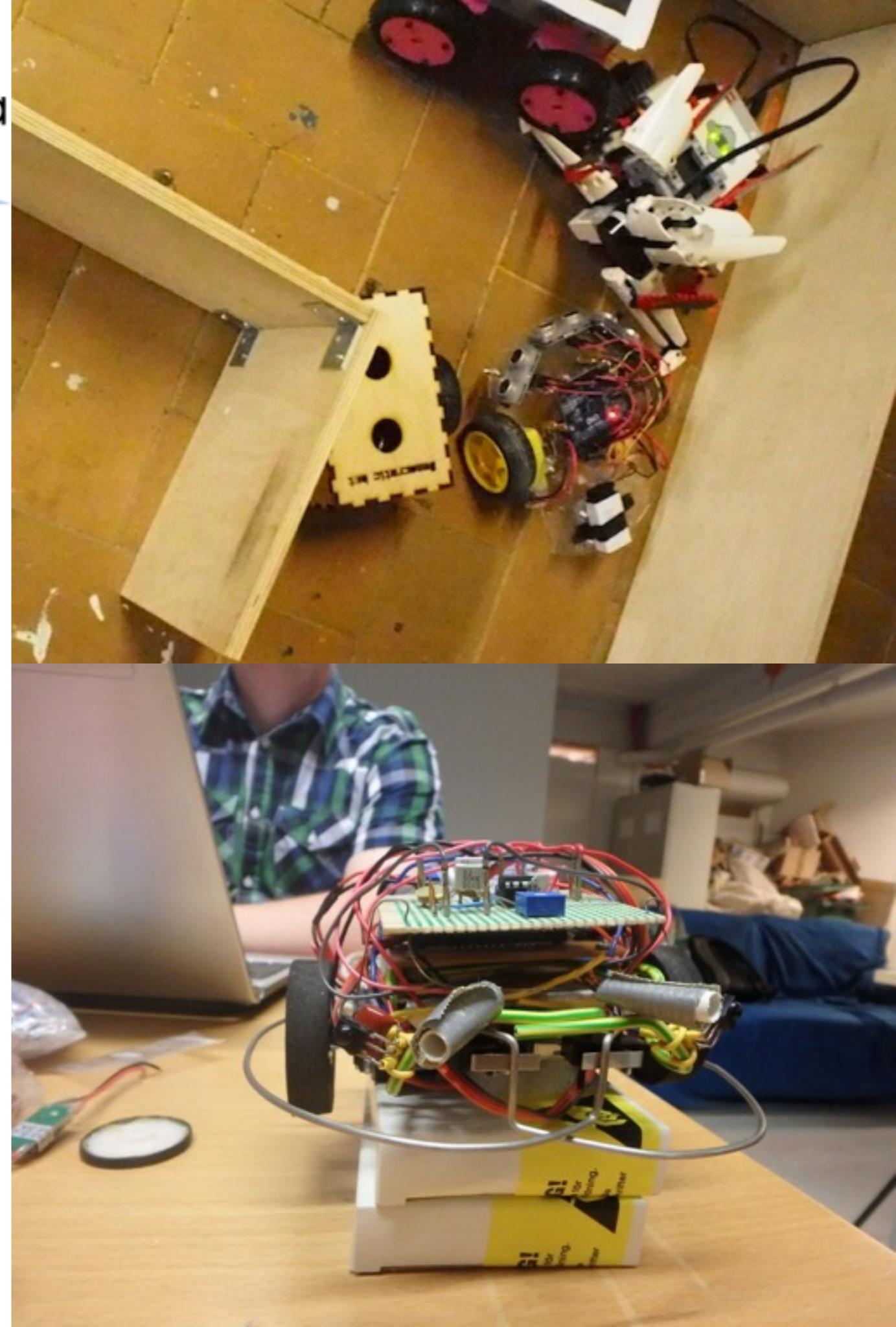


Lisätietoa ja  
viime hetken infoa  
tapahtumasta  
Facebookissa



Tapahtuman järjestäjä  
**hacklab.fi**

Tapahtuma Facebookissa: Robotit Strömbergin puistossa!  
[http://tiny.cc/robotit\\_puistossa](http://tiny.cc/robotit_puistossa)



QUESTIONS ?  
COMMENTS?

AND WE'RE DONE

THANK YOU

