drawing the line

prof. dr. nicolas meseth

1 a world without steps
2 two cuts and a price
3 drawing your own lines

part 1

a world without steps

is this digital?



image: ai-generated (gpt-image-2)

digital is not the same as electronic

digital



flip scoreboard


light switch


abacus


a die

analog


slide rule


dial thermometer


dimmer

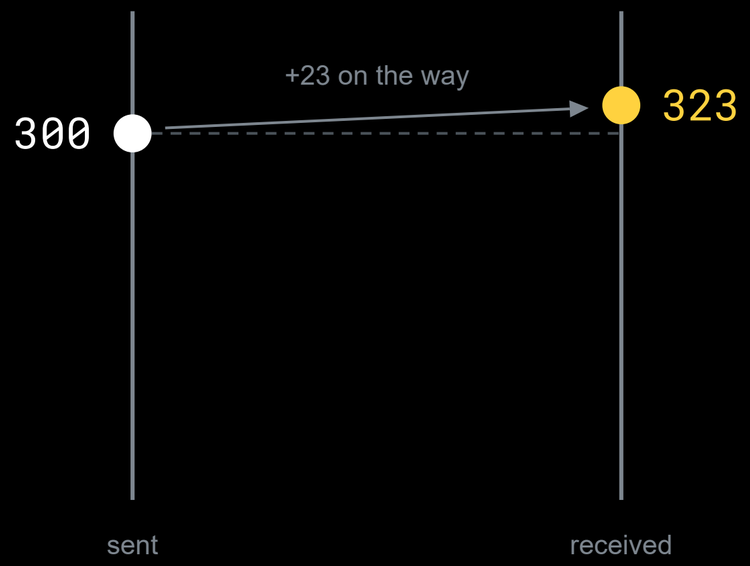

hourglass

not one of them contains a chip, so digital cannot mean electronic.

images: ai-generated (gpt-image-2)

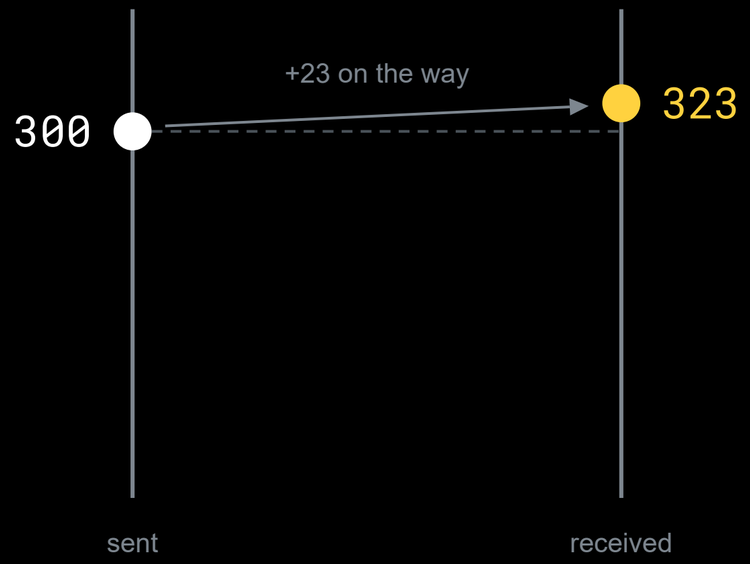
digital means finitely many states,
and nothing in between counts.

analog: every value counts



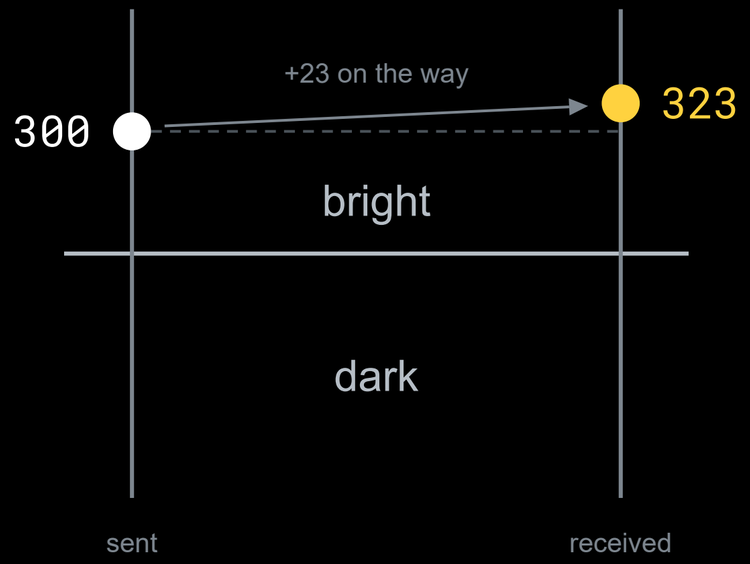
a different value

analog: every value counts



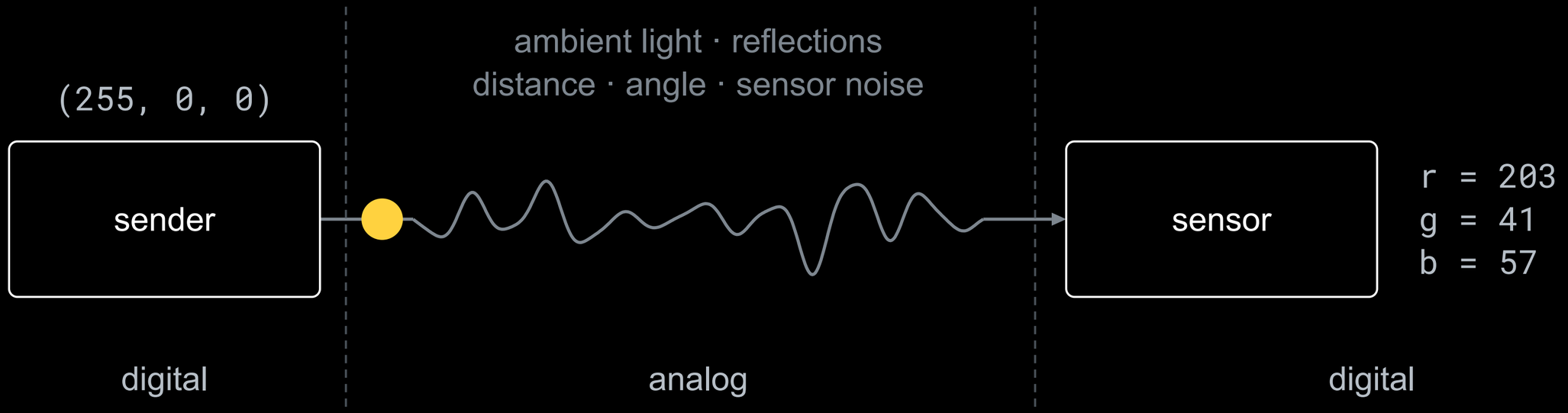
a different value

digital: only the region counts



still bright, the same state

a small disturbance changes the value, but not the state.


ambient light · reflections
distance · angle · sensor noise
(255, 0, 0)
sender
sensor
r = 203
g = 41
b = 57
digital
analog
digital

light: between any two brightnesses lies another



light: between any two brightnesses lies another



your program: a few states, clearly apart

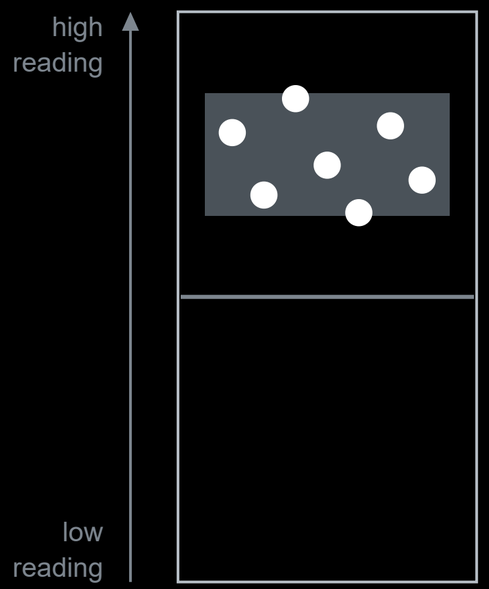


digitising turns the stepless light into a few states.

the same scatter, cut into more regions

● one reading while the same colour is sent

▭ how far they scatter

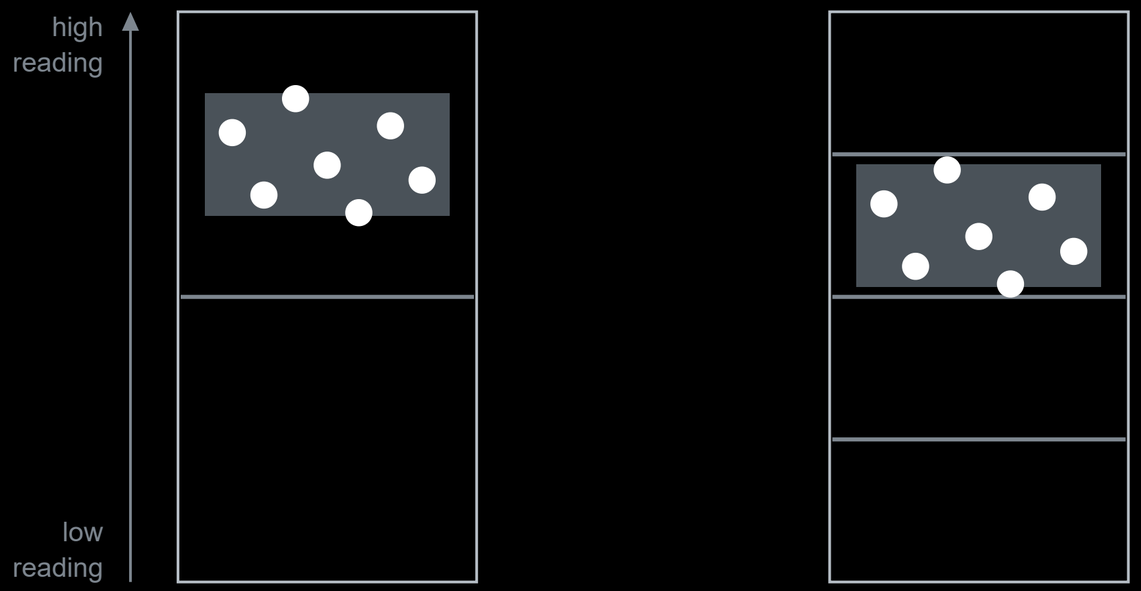


2 regions

the same scatter, cut into more regions

one reading while the same colour is sent

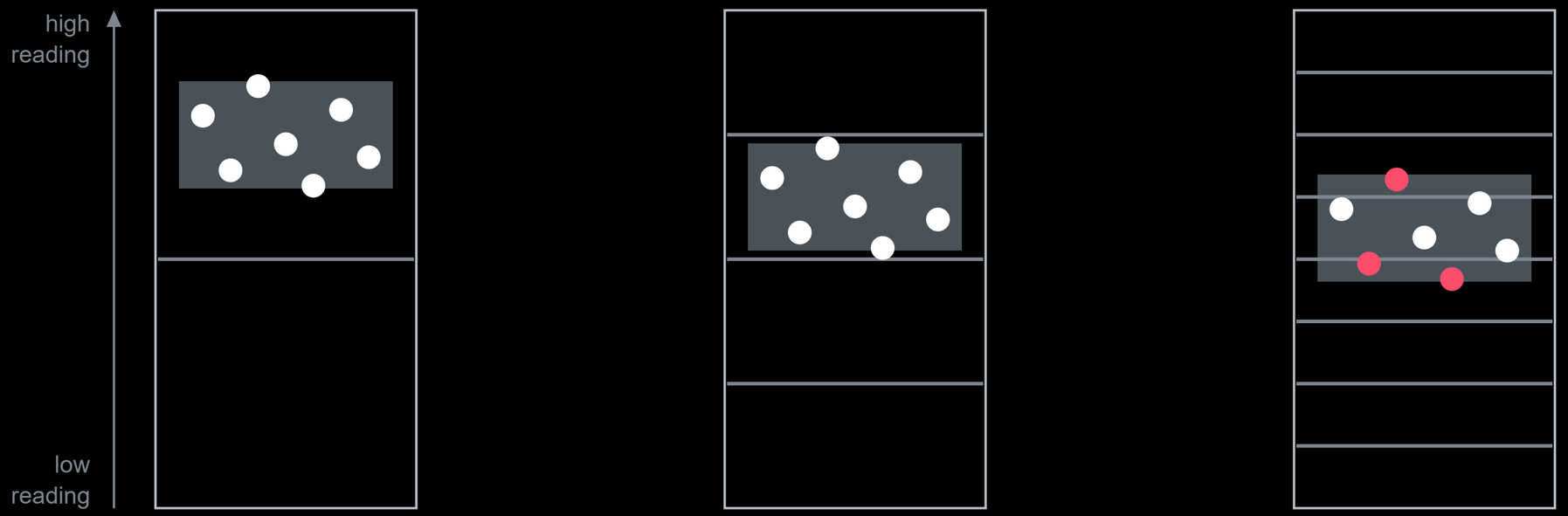
how far they scatter



the same scatter, cut into more regions

one reading while the same colour is sent

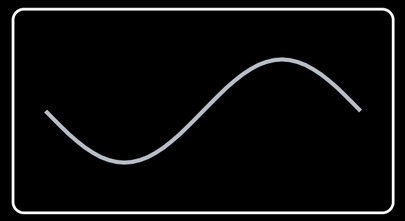
how far they scatter



the readings scatter as much as before, but with more regions some land in the wrong one.

what the thrown-away precision buys

original

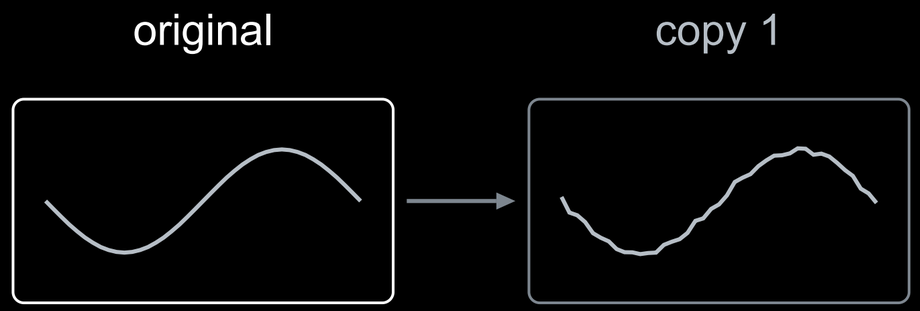


tape: every copy takes over the noise of the last one and adds its own

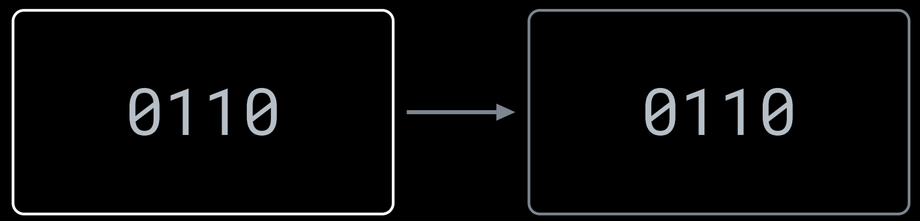
0110

file: every copy is read and set again from clean bits

what the thrown-away precision buys

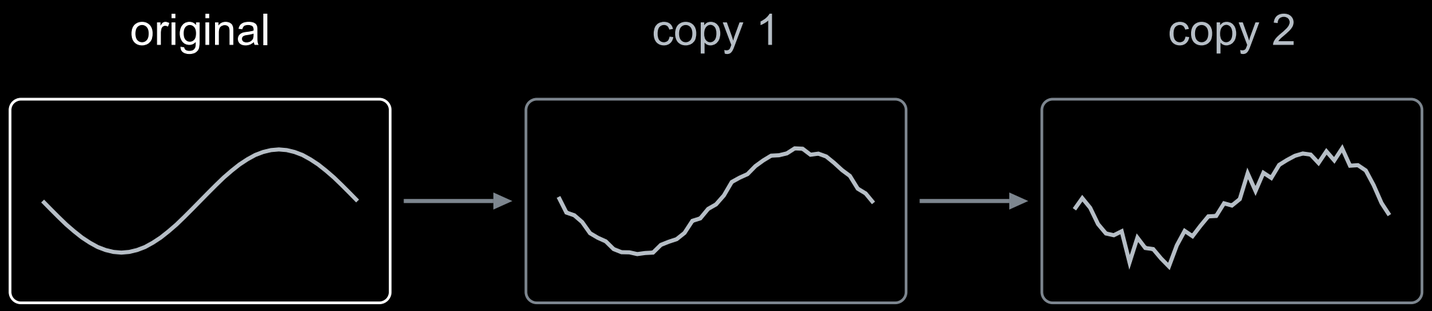


tape: every copy takes over the noise of the last one and adds its own



file: every copy is read and set again from clean bits

what the thrown-away precision buys

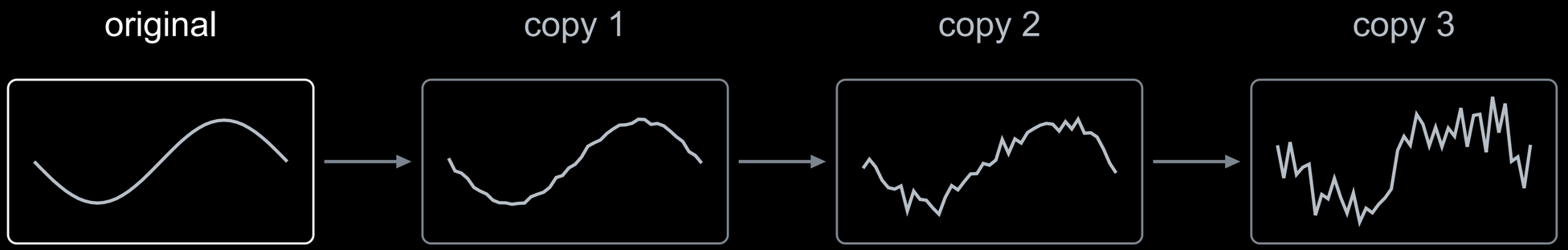


tape: every copy takes over the noise of the last one and adds its own

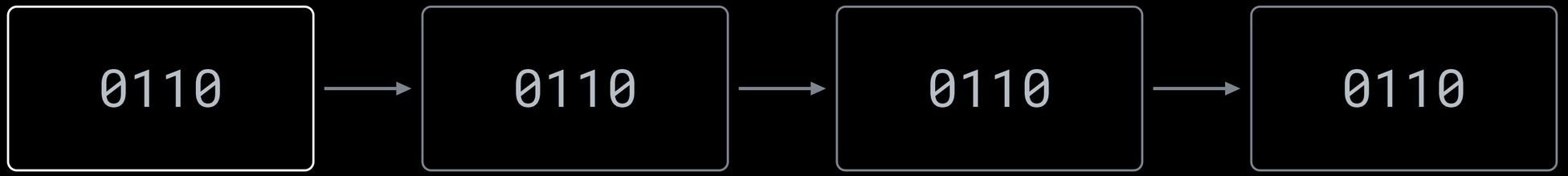


file: every copy is read and set again from clean bits

what the thrown-away precision buys



tape: every copy takes over the noise of the last one and adds its own



file: every copy is read and set again from clean bits

every digital copy is reborn from clean states, so it never inherits the noise.

part 2

two cuts and a price

digitising throws information away, and it does so on purpose.

which exact value inside a region it was is gone for good.

digitising takes two cuts

digitising takes two cuts

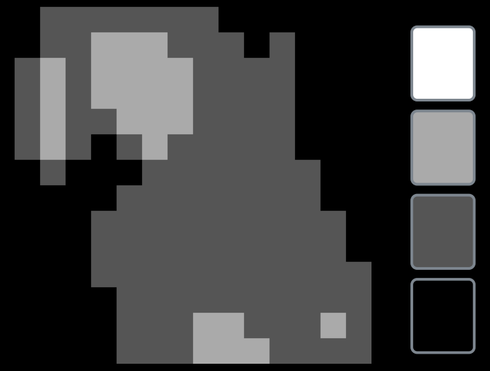
the analog picture



16 × 16 points



4 grey levels



cut 1: sampling
how many points: the resolution

cut 2: quantization
how many levels: the colour depth

sampling cuts the picture into points, and quantization cuts each point's value into levels.

which cut was too coarse?



too few points

blocks



too few levels

hard patches instead of soft shading

```
64 × 64 points × bits per point = size
```

points	bits per point	size	over the light link at 30 bit/s
4,096	24 bit	12 KB	about 55 minutes
4,096	1 bit	512 bytes	about 2 minutes

what you keep, you pay for in minutes.

how far can you go?

how far can you go?

how far can you go?

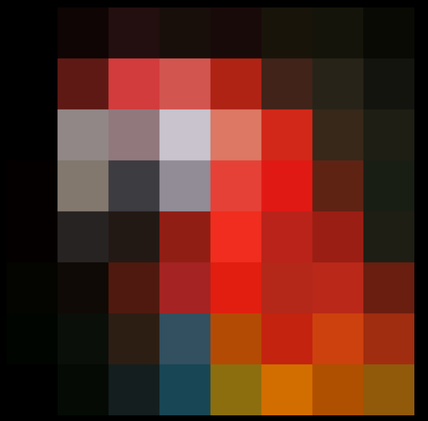
how far can you go?



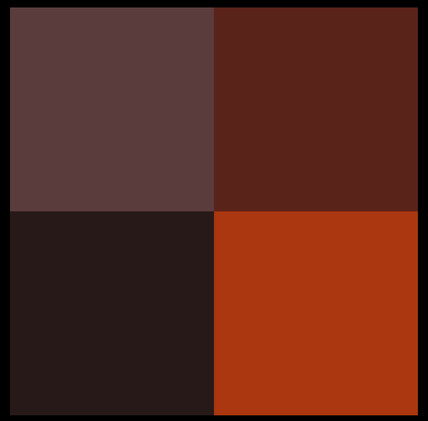
128 × 128
49,152 bytes



32 × 32
3,072 bytes



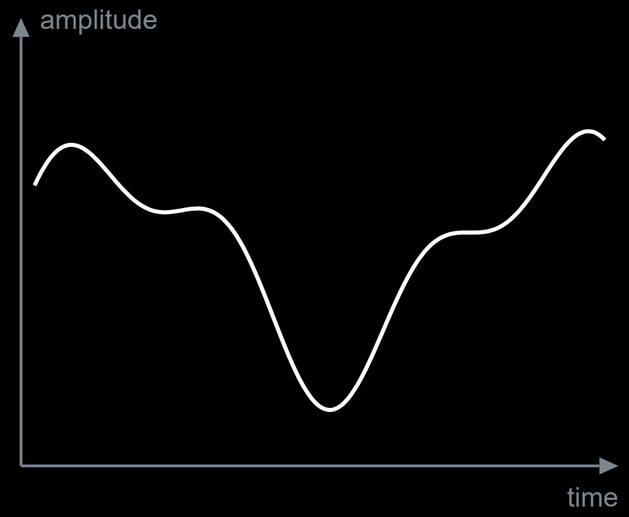
8 × 8
192 bytes



2 × 2
12 bytes

where would you stop?

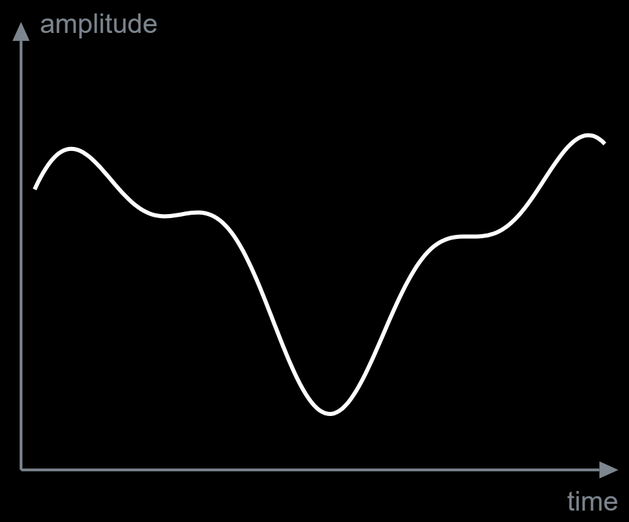
the same two cuts, for sound



the sound wave

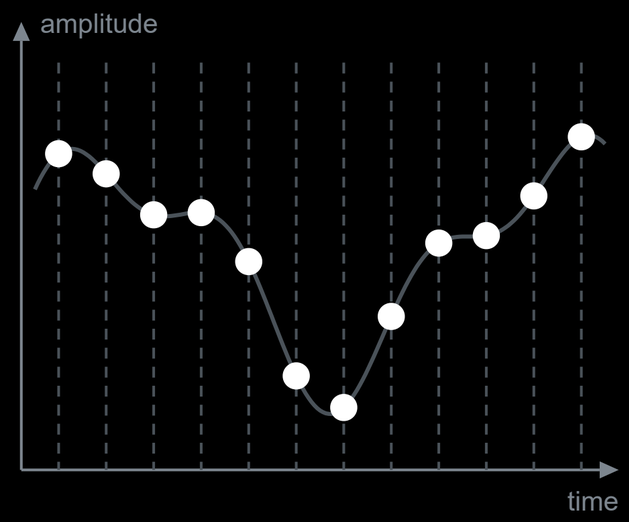
stepless

the same two cuts, for sound



the sound wave

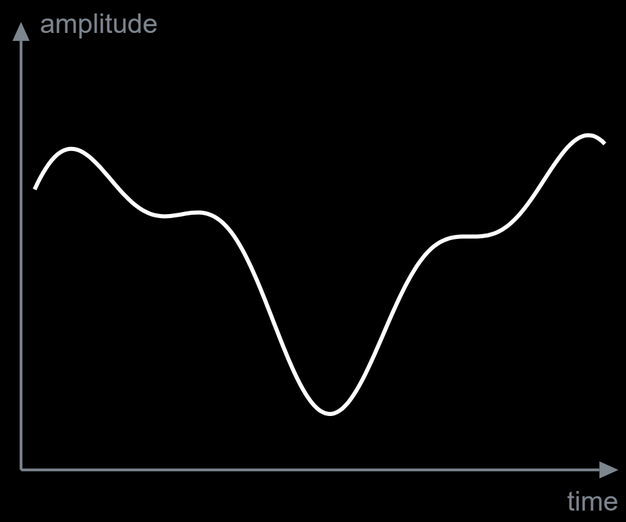
stepless



cut 1: sampling

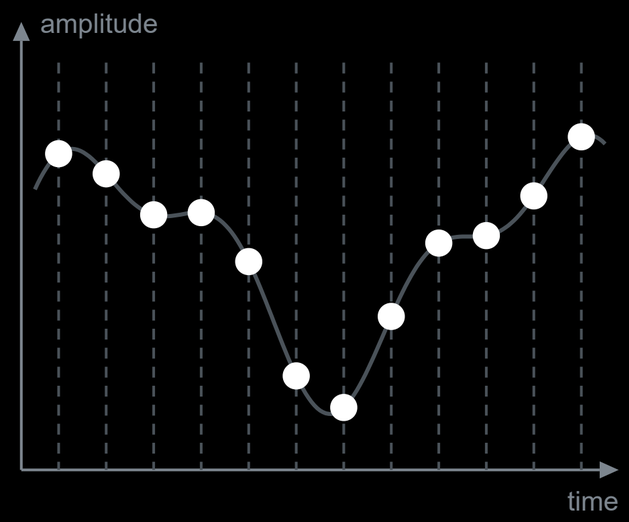
12 points in time

the same two cuts, for sound



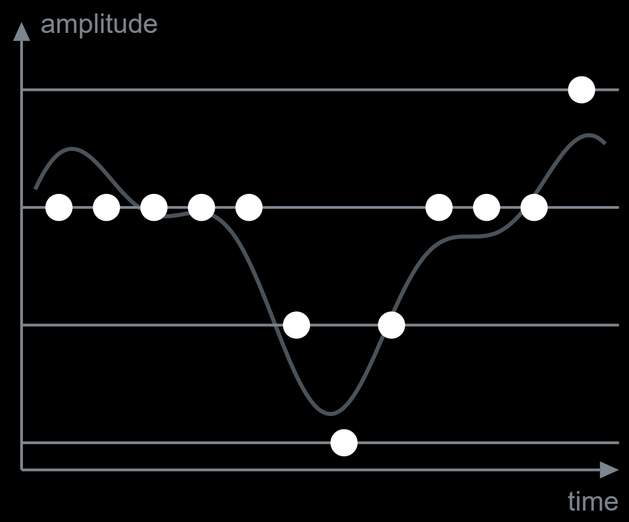
the sound wave

stepless



cut 1: sampling

12 points in time



cut 2: quantization

4 amplitude levels

sampling slices time, and quantization slices the amplitude.

part 3

drawing your own lines

```
from lifi_hardware import sensor
r, g, b, c = sensor.read()

if c > 250:
    symbol = "bright"
else:
    symbol = "dark"
```

this one line turns thousands of possible readings into just two symbols.

how many colours are safe?

two colours are safe, and eight are fast.

how many colours are safe?

two colours are safe,
and eight are fast.

somewhere in between sits your alphabet, and only your own measurements know where.

you decide where the lines go.

you decide where the lines go.

from there on, your decision is the truth, not the physics.