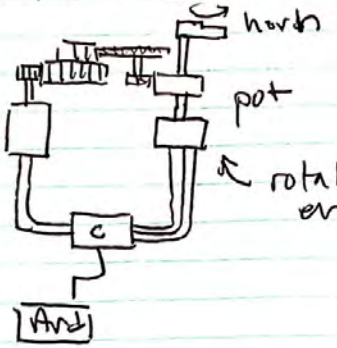
COGS 300 control 01 Sep 23/25

Today we start on the control module.

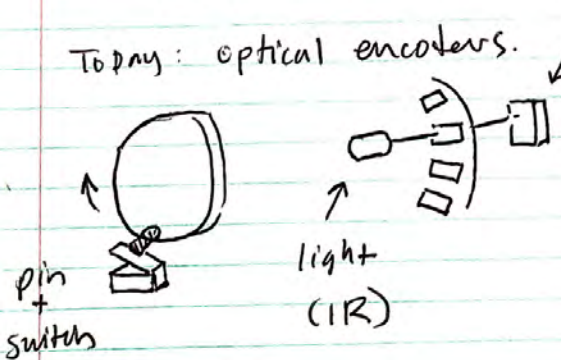
We have a couple of sensors yet, but we're moving a level up in scope to simple behaviours, basic intelligent control.

Last time: servo



- project + meetings
- setup.

Today: optical encoders.



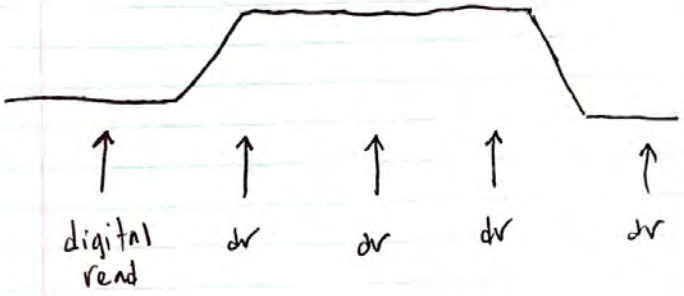
model as a switch

Q: How do we get distance from a "switch" style encoder?

every "click" = $\frac{360}{\#\text{ of holes}}$ degrees

of $C = \pi d$ so $\frac{360}{\#\text{ of holes}} \cdot \pi d$

But let's zoom in:

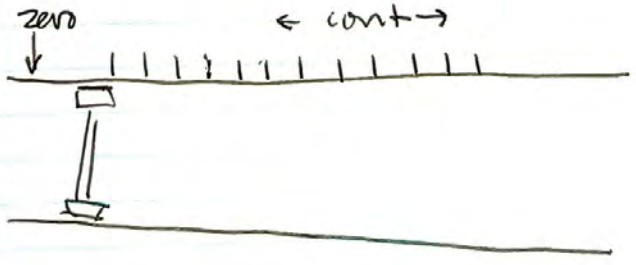


can't "just" do switch on ...
also, backspin.

★ Activity
↳ build circuit
→ figure out lead/fall
→ backspin.

· The general problem we're trying to solve is robot positioning.

Let's zoom out.

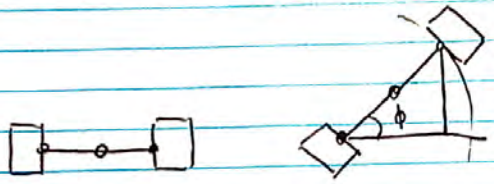


1D motion

★ Encoder → simulation activity.

Telemetry
↓ "remote"
↳ metrics, ie. sensing.

simulation problem:



need liner algebra!

Instead, we will use the encoders to detect movement + speed, not run full simulations.

Instead, we are going to use

Sensor integration

ie. use an external sensor (ultrasonic) To triangulate position.

But, ultrasonics are noisy ... not perfect!

If time:

Design a 1D triangulation system:

two sources of "truth" both with problems...
how to reconcile?

PID control

set point = set = 5cm

measured = pos = 10 cm

error = set − pos

output = p(error)

↑ some constant.

e.g. 10 cm − 5 cm = −5 cm

$10 \cdot (-5) = -50$

5cm ... etc.

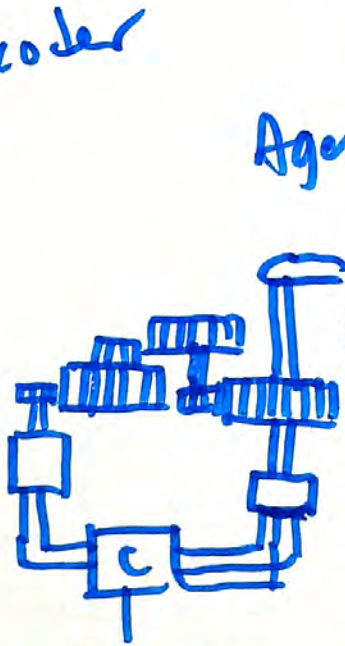
cobs 300 **Control 01** Sep 23/25 @5

Warm up: Draw a series of straight lines

Now, do it with your **eyes** closed.

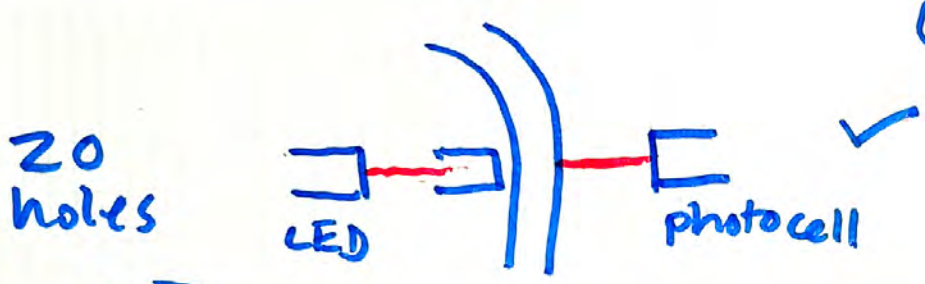
can you match length with your eyes closed?

encoder



-50
10x
-5cm
5cm
10cm
set
pos
err = set - p
out = p(err
prop.

(2)

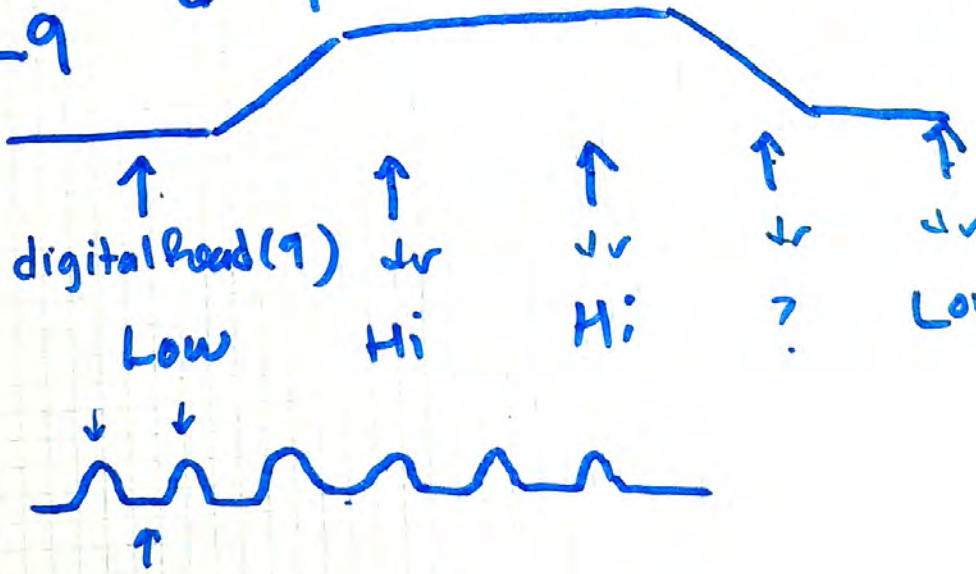


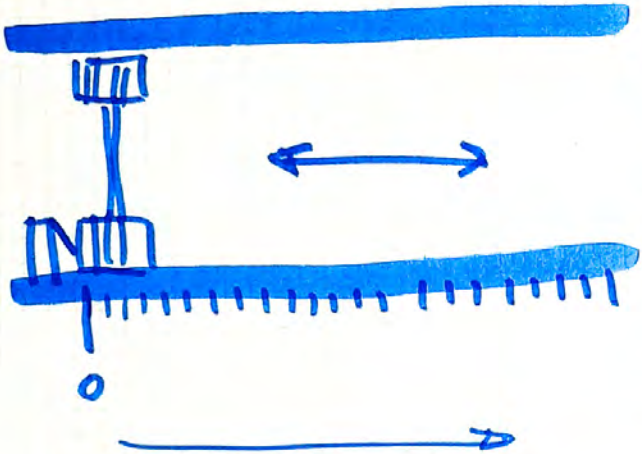
$$\frac{360}{20} = 18 \frac{\text{deg}}{\text{hole}}$$

$$2r \cdot pi = C \qquad \frac{C}{20} = \frac{\text{dist.}}{\text{hole}}$$

$$d \cdot pi = C$$

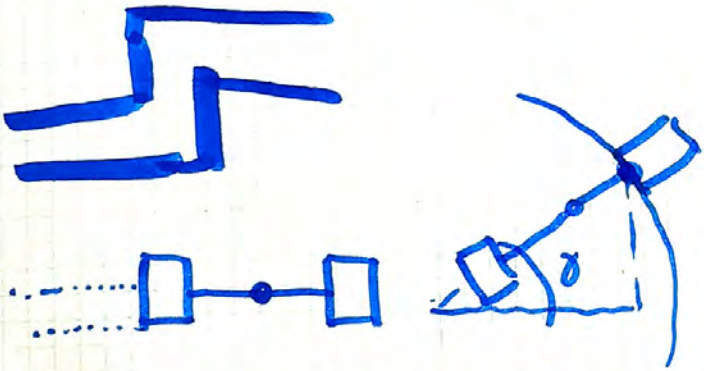
D9
D-9





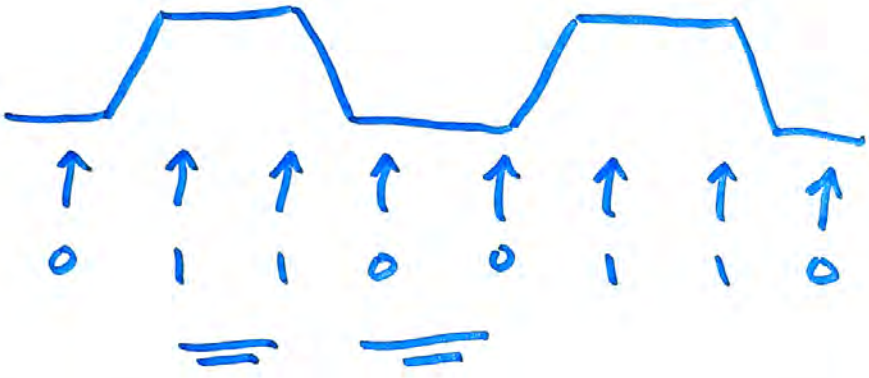
error
resolution

Design ex.



difficult to simulate

Sensor integration



lm=last_measurement= 0;

void loop {

if (lm == 1) {

...

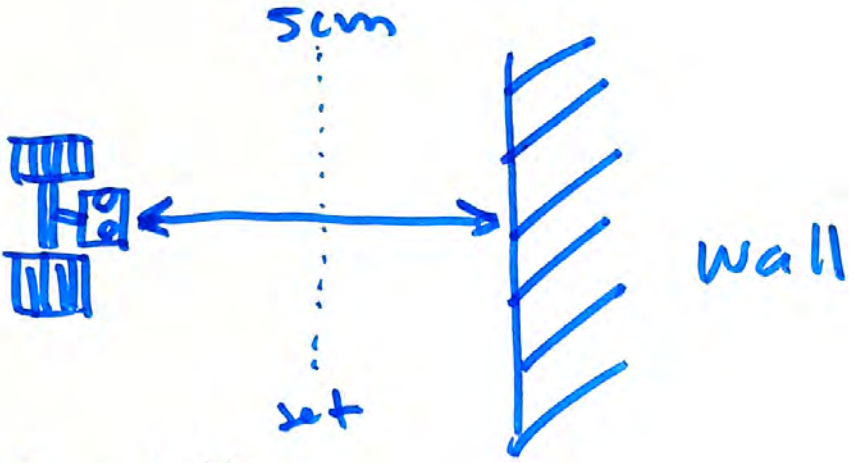
}

if (lm == 0) {

dm = abs(tm - lm

}

~~dm = (tm - lm) & abs~~



triangulation

wheel_1_dist
wheel_2_dist
ultrasonic_dist

prob(d_1)
prob(d_2)
⋮

ud = 10cm
set = 5cm
er = ud − set
out = p(er)

} loop

guess = d_3
wheel = d_4
us = d_5