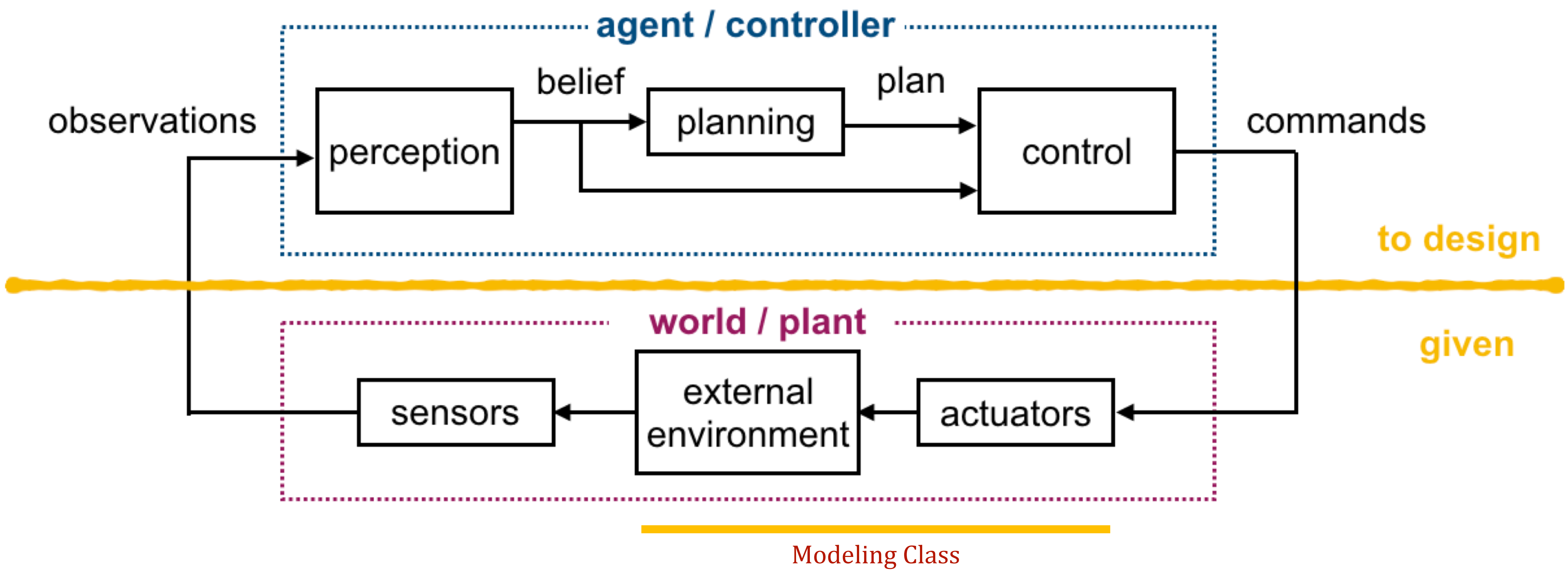
Odometry calibration

Big picture



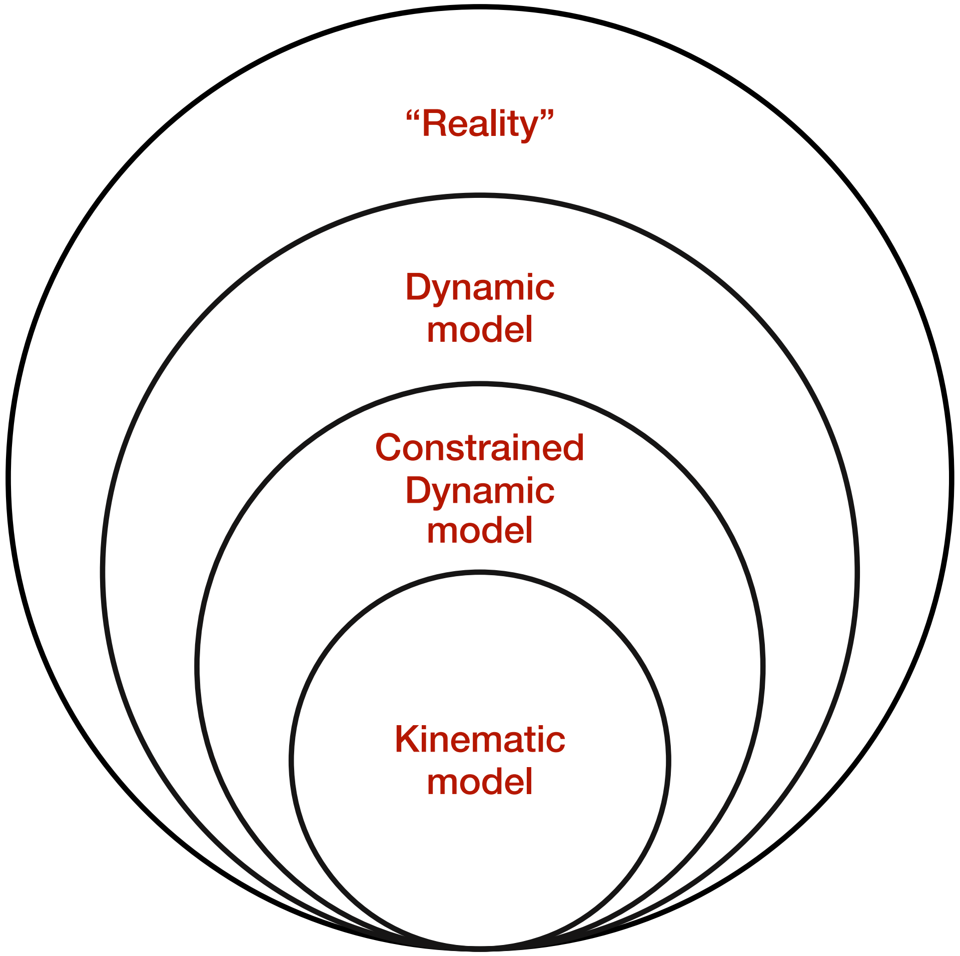
We derived a mathematical model for a differential drive robot

How do we find the parameters of the model?

Intuition

Odometry = ὀδός + μέτρον = measurement of the road/path

Odometry Calibration = Determination of model parameters to "match" predicted motion and measurements



Problem Definition (General formulation)

Given a model of the Duckiebot and a set of discrete measurement from which the output can be estimated, find the most likely calibration parameters.

Calibration parameters

Model of the system:
$$\dot{x} = f(p; x, u)$$
$$y = g(x)$$

Measurements (not necessarily evenly spaced in time)

Set of discrete measurements: $\mathcal{M}_n = \{m_k = m(t_k), t_1 < \ldots < t_k < \ldots < t_n\}$

Set of output estimates: $\hat{\mathcal{Y}}_n = \{\hat{y}_k = h(m_k), k = 1, \ldots, n\}$

Most likely calibration parameters: $\mathrm{p}^* = \arg\max_{\mathrm{p}} \mathrm{prob}(\mathrm{measurements}|\mathrm{p})$

Different cases

Calibration parameters

Model of the system:
$$\dot{x} = f(p; x, u)$$
$$y = g(x)$$

$f(\cdot)$:
- Constrained Kinematic model
- Kinematic model
- Constrained Dynamic model
- More general Dynamic model

$g(\cdot)$:
- Robot pose
- Sensor pose

Measurements (not necessarily evenly spaced in time)

Set of discrete measurements: $\mathcal{M}_n = \{m_k = m(t_k), t_1 < \ \ldots < t_k < \ \ldots < t_n\}$

m_k:
- "Internal" sensors ("interoception")
  Wheel Encoders, IMUs, Compass, …
- "External" sensors ("exteroception")
  Camera, Lidar, Infrared, …

Different cases (Typical solution)

Calibration parameters

Model of the system:

$$\dot{x} = f(p; x, u)$$
$$y = g(x)$$

$f(\cdot)$:
- Constrained Kinematic model
- Kinematic model
- Constrained Dynamic model
- More general Dynamic model

$g(\cdot)$:
- Robot pose
- Sensor pose

Measurements (not necessarily evenly spaced in time)

Set of discrete measurements: $\mathcal{M}_n = \{m_k = m(t_k), t_1 < \ \dots < t_k < \ \dots < t_n\}$

m_k:
- "Internal" sensors (interoception)
  Wheel Encoders, IMUs, Compass, …
- "External" sensors (exteroception)
  Camera, Lidar, Infrared, …

Odometry calibration with encoders

- Encoder measures a "tick" every $\Delta\varphi$, there are N_{tot} ticks in $2\pi R$

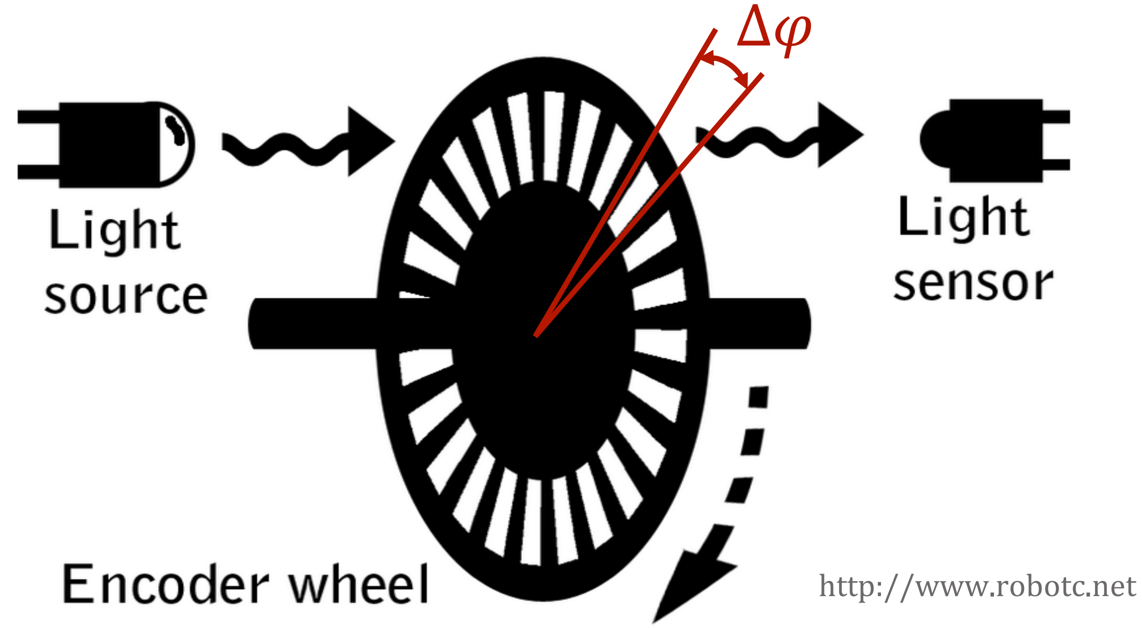
Number of ticks measured in Δt

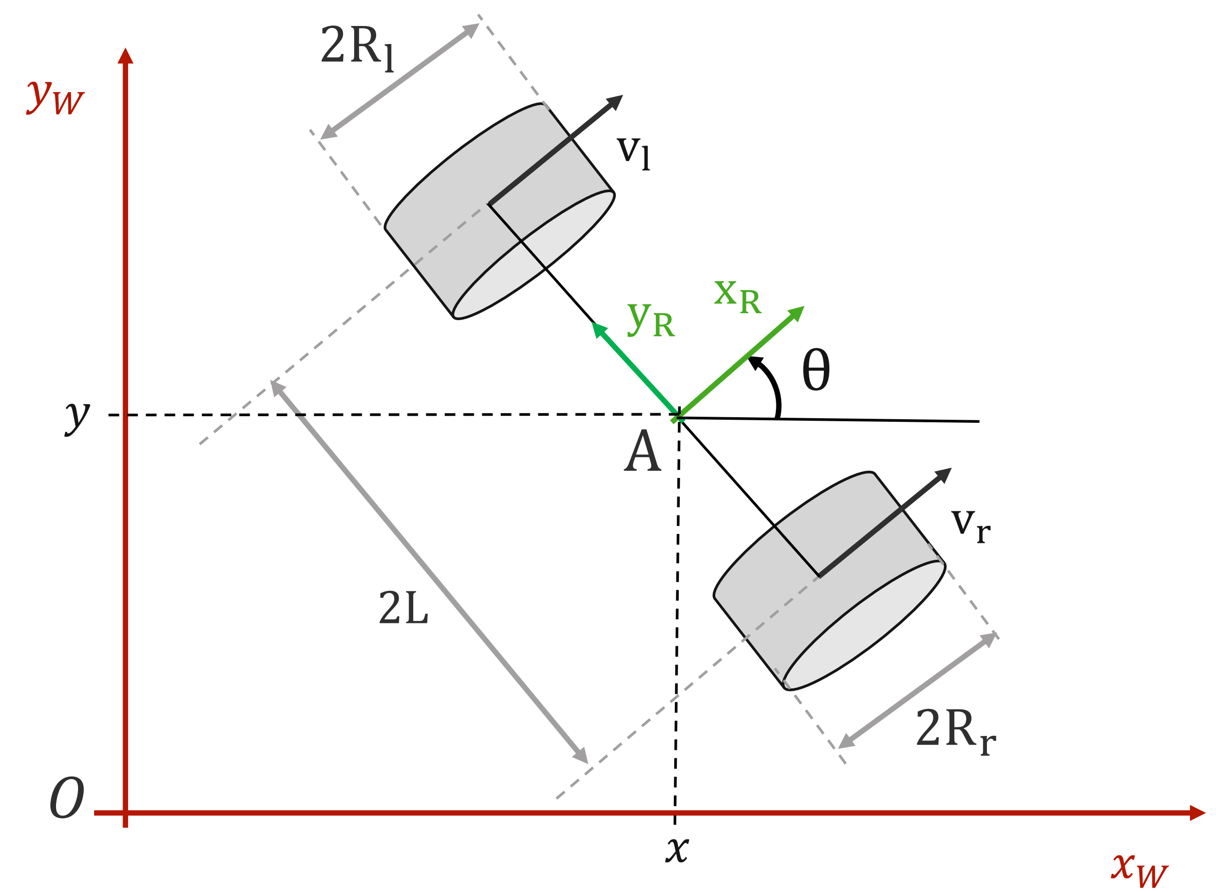
- Travelled distance in Δt: $d_{l/r} = 2\pi R_{l/r} \frac{N_{l/r}}{N_{tot}}$

Distance travelled by A in time interval

- Kinematic model: $\begin{cases} v_A = \frac{v_r + v_l}{2} \rightarrow d = \frac{d_r + d_l}{2} \\ \dot{\theta} = \frac{v_r - v_l}{2L} \rightarrow \Delta\theta = \frac{d_r - d_l}{2L} \end{cases}$

- $\begin{cases} x(t+\Delta t) = x(t) + d\cos\theta \\ y(t+\Delta t) = y(t) + d\sin\theta \\ \theta(t+\Delta t) = \theta(t) + \frac{d_r - d_l}{2L} \end{cases}$





Odometry calibration with encoders

- Calibration parameters: $p = (R_l, R_r, L)$

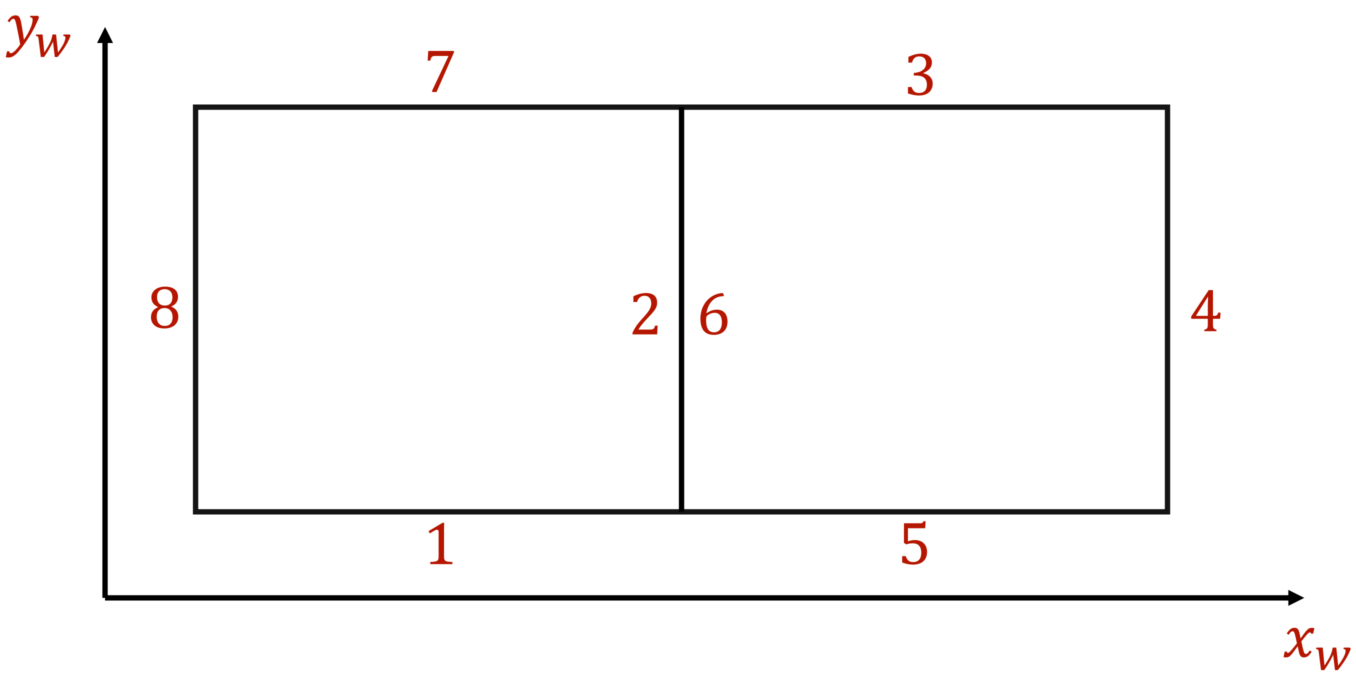
- q from:

  Coordinates of points known in world frame

  Drive parallel to line on points

- $\hat{q}(p)$ from $\begin{cases} x(t+\Delta t) = x(t) + d\cos\theta \\ y(t+\Delta t) = x(t) + d\sin\theta \\ \theta(t+\Delta t) = \theta(t) + \frac{d_r - d_l}{2L} \end{cases}$

- Least squares to determine parameters



But:

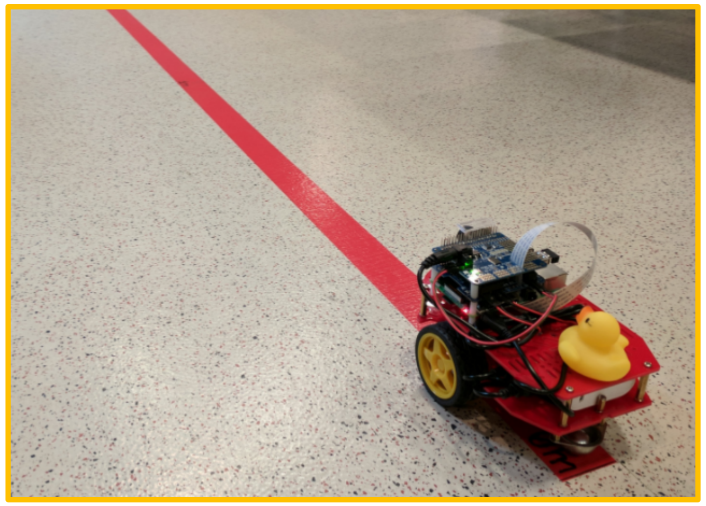
- Adding noise leads to drift
- ~~Duckiebots don't have encoders~~

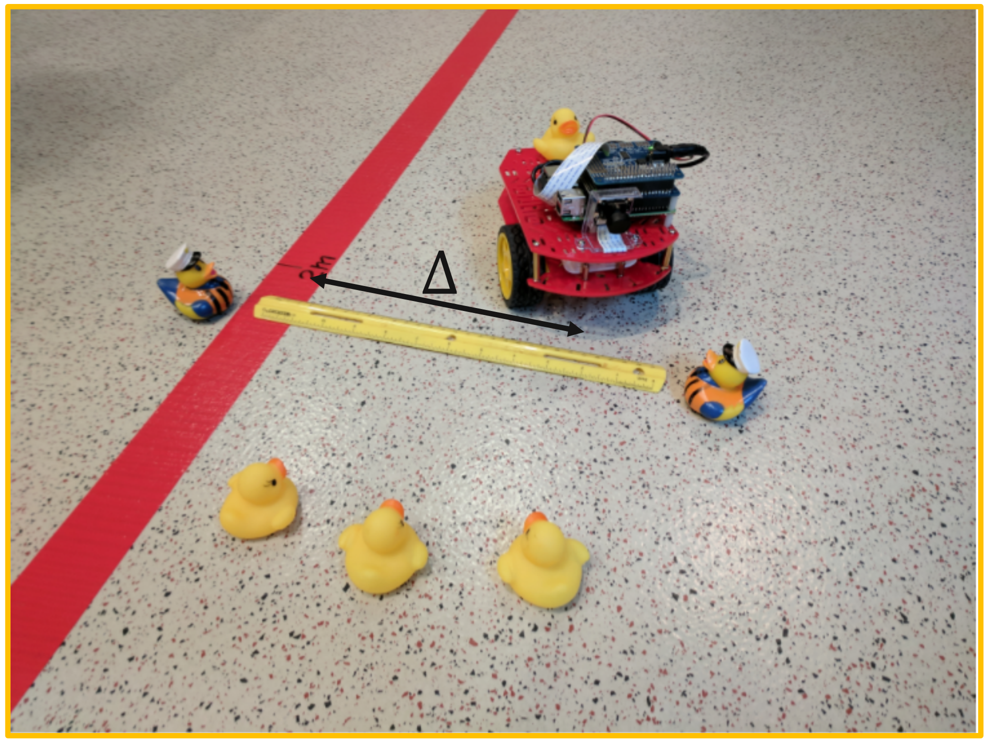
2020 Now they do! DT19, DB-Beta

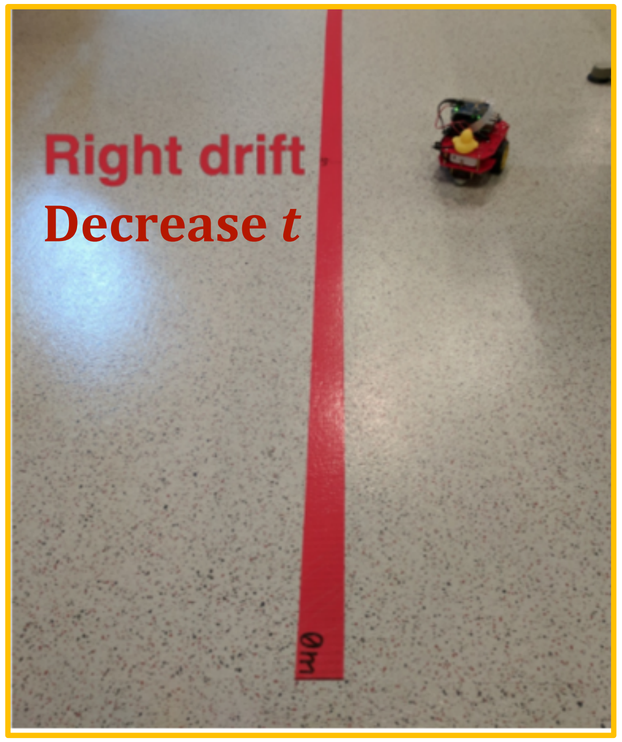
Manual odometry calibration for Duckiebots (gain / trim model)

- Trim and Gain model: $\begin{cases} \overset{\text{Voltage to left/right motors}}{V_l} = (\underset{\text{Gain}}{g} + t)(v_A - \omega L) \\ V_r = (g - \underset{\text{Trim}}{t})(v_A + \omega L) \end{cases}$

- Step 1: Set gain to minimize slipping of wheels ($g \sim 1 - 1.5$)
- Step 2: Drive Duckiebot forward on straight line for ~2 m
- While $\Delta > 10$ cm: Change trim





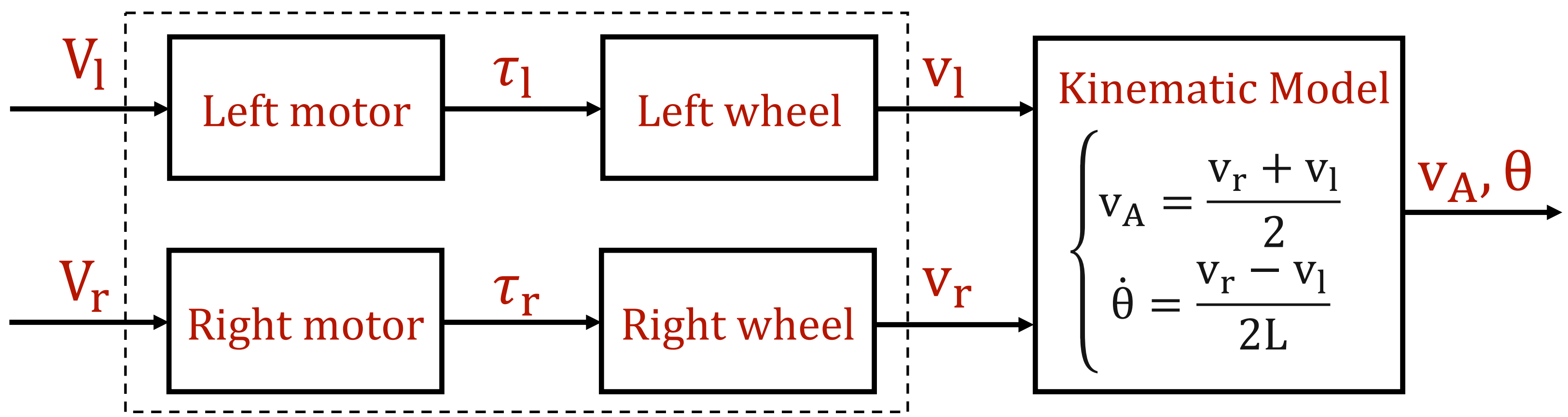


Kinematic model with motors at steady state

- Geometric hypothesis:
1. Identical wheels (of diameter $2R$)
2. Equally spaced wheels (axle length = $2L$)
3. Symmetric along longitudinal axis
4. Center of mass on symmetry axis

- Kinematic hypothesis
5. Rigid body
- Kinematic constraints
6. No skidding
7. No slipping

- Motor dynamics
8. **Steady State**



- At steady state:

"Wheel" calibration parameters

$$v_{l/r} = c_{l/r} V_{l/r}$$

$$c_{l/r} = \frac{R_{l/r}}{k_{b,l/r}}$$

Wheel radii

Motor constants

- Through trim we measure ratio wheel calibration parameters

$$t = \frac{c-1}{c+1}$$

$$c = \frac{c_r}{c_l}$$

The math (for reference)

- $V(t) = R\,i(t) + \cancel{L\frac{di}{dt}(t)} + K_b\,\dot{\varphi}(t) \rightarrow i = \frac{1}{R}\left(V - \frac{K_b}{r}v\right)$
- $\dot{\varphi}(t) = \frac{v}{r}$ Pure rolling
- $\tau(t) = K_i\,i(t) = \frac{K_i}{R}\left(V - \frac{K_b}{r}v\right) \rightarrow \boldsymbol{v = \frac{r}{K_b}V \Rightarrow}$ $\begin{aligned}\boldsymbol{v_l} &\boldsymbol{= \frac{r_l}{K_{b,l}}V_l = c_l V_l}\\ \boldsymbol{v_r} &\boldsymbol{= \frac{r_r}{K_{b,r}}V_r = c_r V_r}\end{aligned}$
- $J\ddot{\varphi} = \frac{J}{r}\dot{v} = \tau - \cancel{\tau_{dist}}$

Ignore disturbance for now

From kinematics:

- $\begin{cases} v_l = (v_A - \omega L) \\ v_r = (v_A + \omega L) \end{cases} \Rightarrow \begin{cases} V_l = \frac{1}{c_l}(v_A - \omega L) \\ V_r = \frac{1}{c_r}(v_A + \omega L) \end{cases}$
- Trim and Gain model: $\begin{cases} V_l = (g + t)(v_A - \omega L) \\ V_r = (g - t)(v_A + \omega L) \end{cases}$

Let $g = g_0 = 1$

$$\begin{cases} \frac{1}{c_l} = g + t = 1 + t \\ \frac{1}{c_r} = g - t = 1 - t \end{cases}$$

$$\boldsymbol{t = \frac{c - 1}{c + 1}}$$

$$\boldsymbol{c = \frac{c_r}{c_l}}$$

Summary

- All models are wrong, some are useful
- Given a model, fit the parameters to the observations to calibrate
- In Duckietown, we can use measurements from the encoders, the camera or do it "manually" (trim and gain)

Try it yourself! https://docs.duckietown.org/daffy/opmanual_duckiebot/out/wheel_calibration.html