Reactive Navigation

Michael Yuhas – PhD Candidate

Energy Research Institute @NTU

Learning Objectives

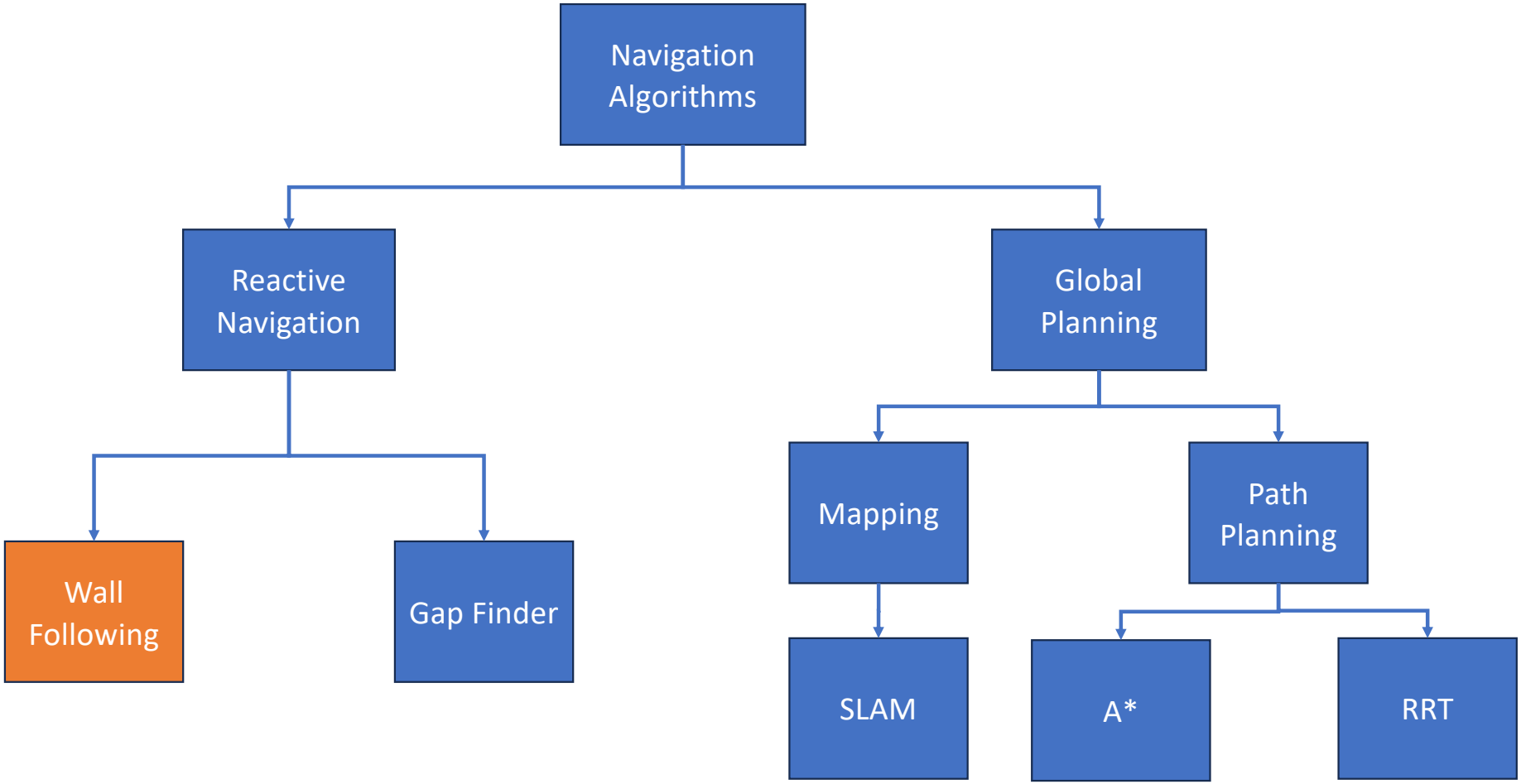
- Identify the different types of navigation algorithms used in robotics and their relative strengths and weeknesses
- Describe the effect of proportional, integral, and derivative control coefficients in a PID controller
- Apply PID control to allow a robot to autonomously navigate a closed circuit

Given some sensor data, what control actions should the robot apply to advance toward its goal?

Top-down projection of lidar data from F1Tenth car

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Taxonomy of Navigation Algorithms



Reactive Navigation – Strengths and Weaknesses

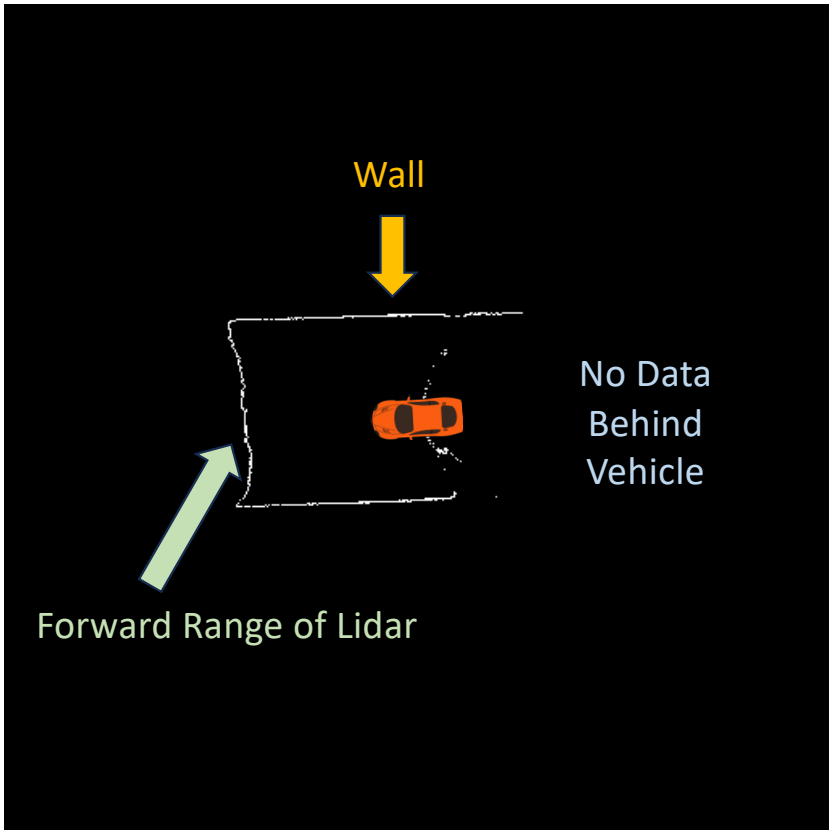
Strengths

- Simple to implement
- By definition reacts to changes in the environment
- Does not require a mapping phases
- Usually computationally inexpensive

Weaknesses

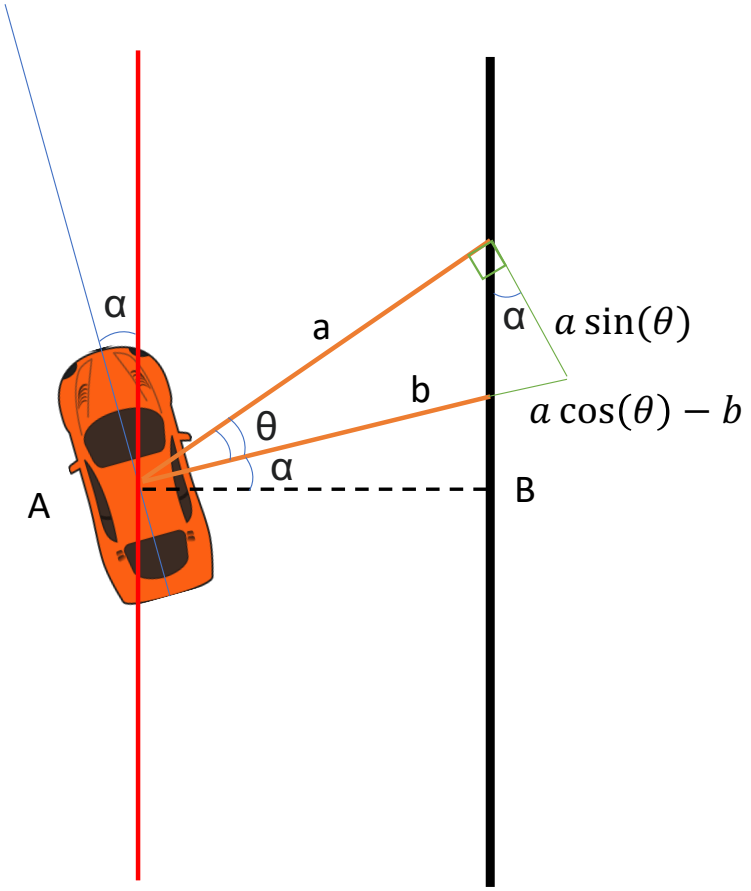
- Cannot implement globally optimal strategies
- Only operates on data locally available

A Simple Goal: Wall Following



- A car following one wall will eventually complete a lap of a closed circuit
- If it can maintain a safe distance from the wall, it won't crash
- How can we implement this?

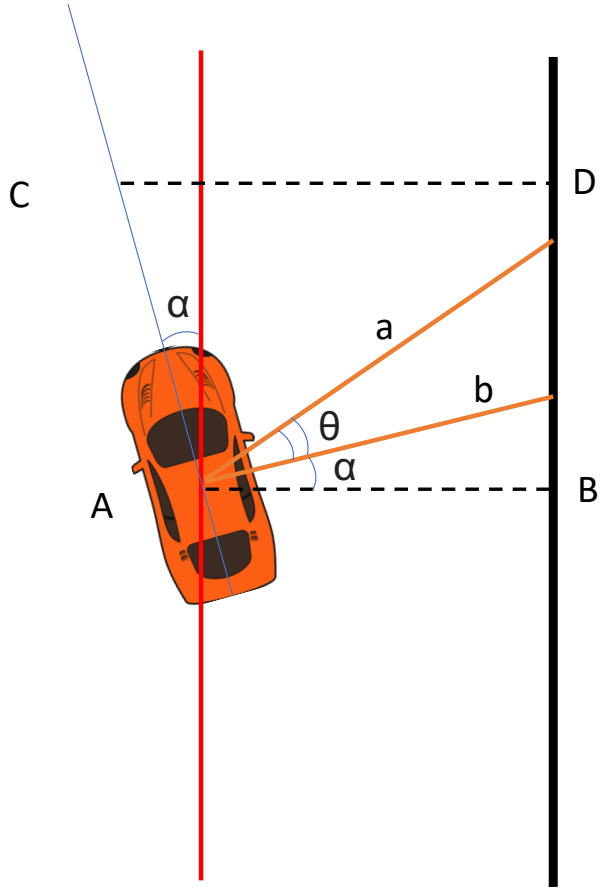
Calculating Current Distance from the Wall



$$\alpha = \tan^{-1}\left(\frac{a\cos(\theta) - b}{a\sin(\theta)}\right)$$

$$\overline{AB} = b\cos(\alpha)$$

Correcting for Distance Traveled



$$\alpha = \tan^{-1}\left(\frac{a\cos(\theta) - b}{a\sin(\theta)}\right)$$

$$\overline{AB} = b\cos(\alpha)$$

$$\overline{CD} = \overline{AB} + \overline{AC}\cos(\alpha)$$

Naïve Control Algorithm – Bang-Bang Control

$$u(t) = \begin{cases} u_{max} \text{ if } e(t) \geq 0 \\ -u_{max} \text{ if } e(t) < 0 \end{cases}$$

- Commonly used in systems where equipment operates more efficiently at some u
  - E.g. Furnace, Aircon
- Will lead to overshoot
- Can lead to rapid cycling

Proportional Control

$$u(t) = k_p e(t)$$

- Apply a control action proportional to the error
  - If error is negative, we automatically steer in the reverse direction
  - As we approach to set point, control action decreases
- Still likely to have overshoot
- Settling time may not be as fast as possible
- https://chev.me/pid-demo/

Derivative Control

$$u(t) = k_p e(t) + k_d \frac{de(t)}{dt}$$

- If the change in error is large, we should apply a larger control action
- As we approach the set point $\frac{de(t)}{dt}$ goes negative, slowing the rate of correction
- Can help prevent overshoot
- https://chev.me/pid-demo/

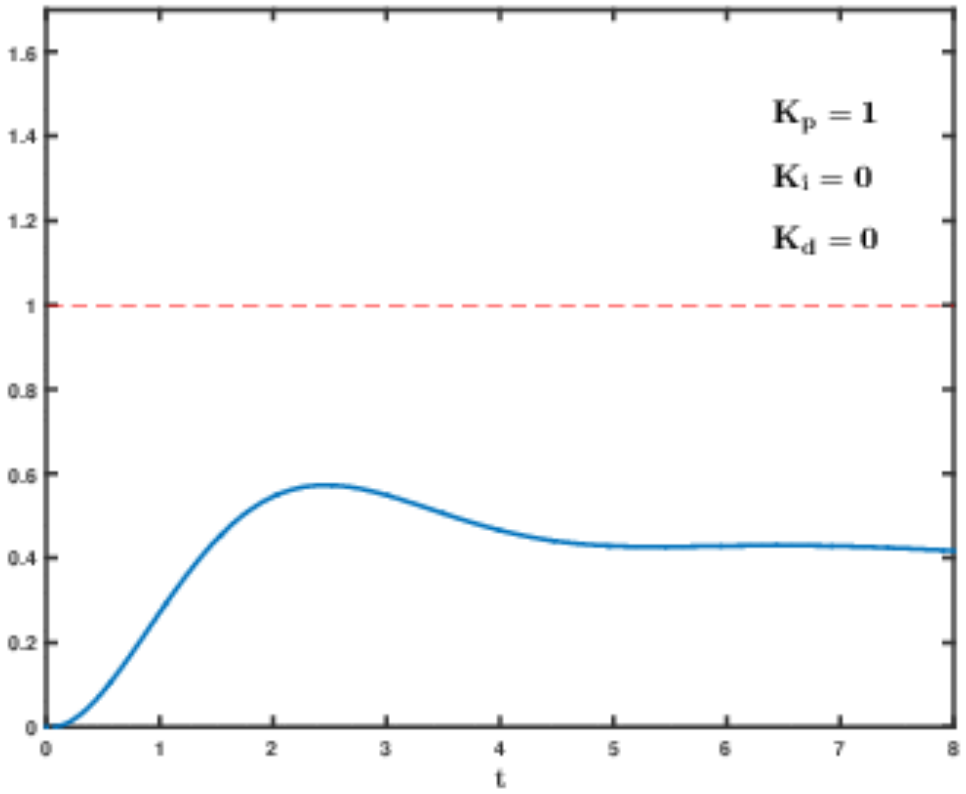
NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Integral Control

$$u(t) = k_p e(t) + k_d \frac{de(t)}{dt} + k_i \int_0^t e(t)dt$$

- If a constant force is applied against the controller, adjust the gain proportional to that force
  - E.g. navigation in a cross-wind
- Integral accumulates the longer an error is present
- Unwinding the integral can take some time: risk of instability
- https://chev.me/pid-demo/

How to Select the Optimal Control Coefficients?



"Effects of varying PID parameters (K_p,K_i,K_d) on the step response of a system" by Physich, licensed under CC0

- Ziegler-Nichols Tuning Method
  1. Set all coefficients to 0
  2. Adjust k_p until stable, consistent oscillations occur (ultimate gain, k_u)
  3. Find oscillation period (T_u)
  4. Set:
     - $k_p = 0.6k_u$
     - $k_d = 0.075k_uT_u$
     - $k_i = 1.2k_u/T_u$
  5. Fine-tune as needed
- Fine tuning
  - Bigger k_p: less rise-time, more overshoot
  - Bigger k_d: less overshoot, less settling time
  - Bigger k_i: less steady state error, more overshoot

Now It’s Your Turn…

Further Reading

- F1Tenth Course Material:
  - University of Pennsylvania. (2022). PID Control for Wall Following. [Online]. Available: https://f1tenth-coursekit.readthedocs.io/en/latest/lectures/ModuleB/lecture04.html.
- PID Tuning:
  - V. V. Patel, "Ziegler-nichols tuning method: Understanding the pid controller." *Resonance,* vol. 25, no. 10, pp. 1385-1397, 2020. Available: https://www.ias.ac.in/public/Volumes/reso/025/10/1385-1397.pdf.
- PID Demo:
  - O. Kalachev. *PID Controller Demo*. (2023). [Online]. Available: https://github.com/okalachev/pid-demo/tree/1cb6830e53779ff9d7e34b29ff2c74ec4eb79c15.