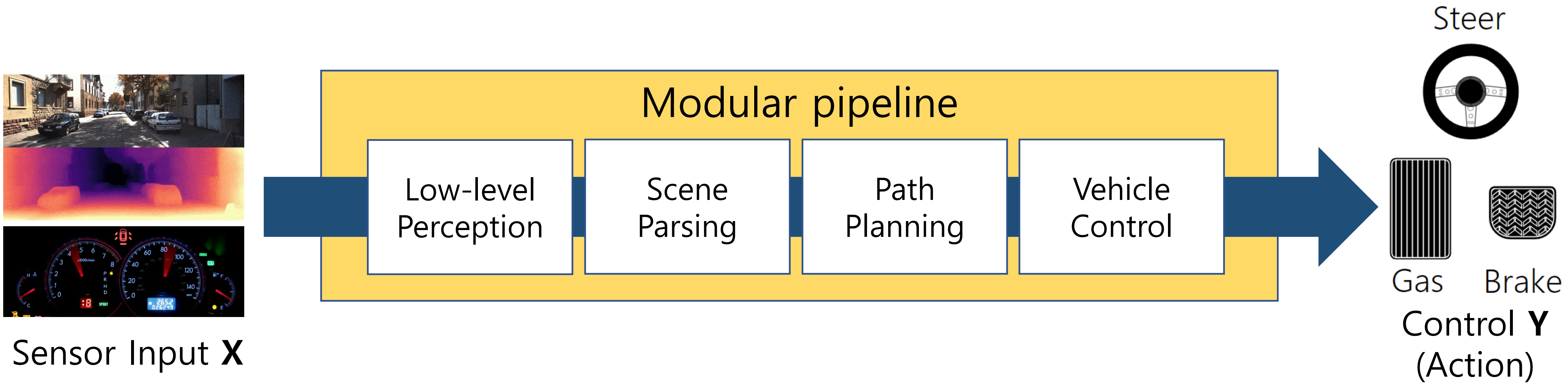
Advanced Programming Practice
Vehicle Control

2023 Fall, CSE4152

Sogang University

Slide credit: Andreas Geiger

Modular Pipeline



- Low-level Perception & Scene Parsing: Lecture 1
- Path training: Lecture 2
- **Vehicle Control: Lecture 3**

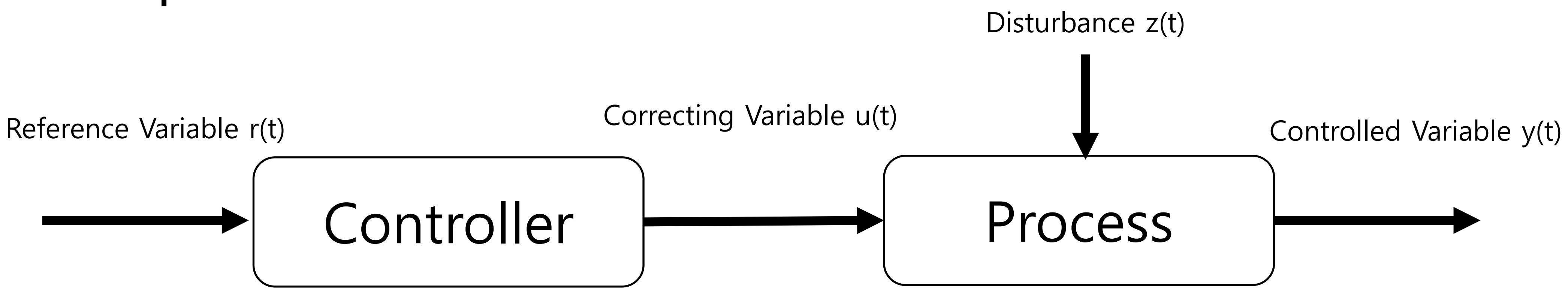
Vehicle Control

- Key objects: Controller and Object
  - Controller will give an action to the object for given status.
  - Object (vehicle) takes and processes the action and the state of it changes.

- Goal of vehicle control
  - Achieve the target state of the vehicle comfortably.
    - No oscillation, less damping.
  - Keeping speed at 60 km/h
  - Driving the curve regularly
  - Follow the way points

Two Major Types of Controller

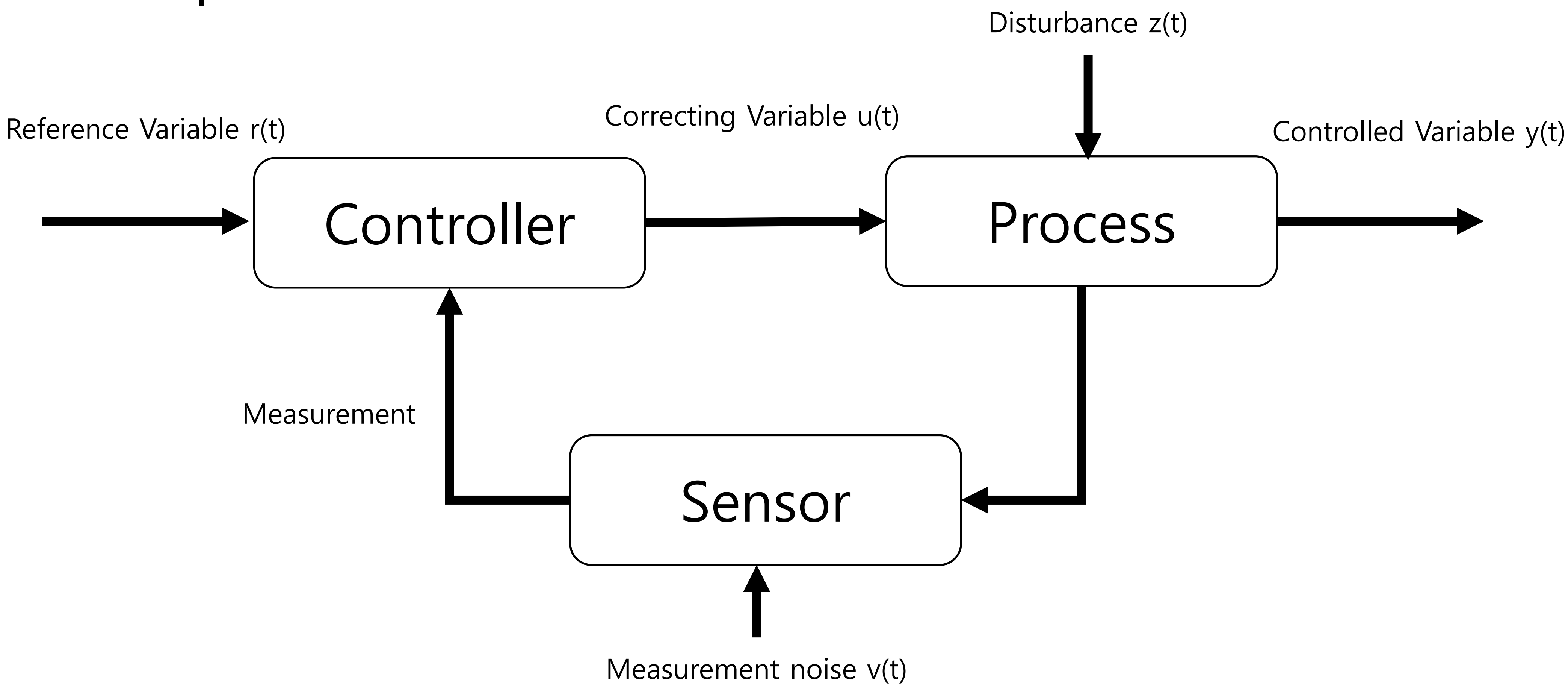
- Open-loop Control



- No feedback → off guard for unknown disturbance → severe **drift**
- The controller must be thoroughly calibrated.
- E.g. Immersion water heater and toaster

Two Major Types of Controller

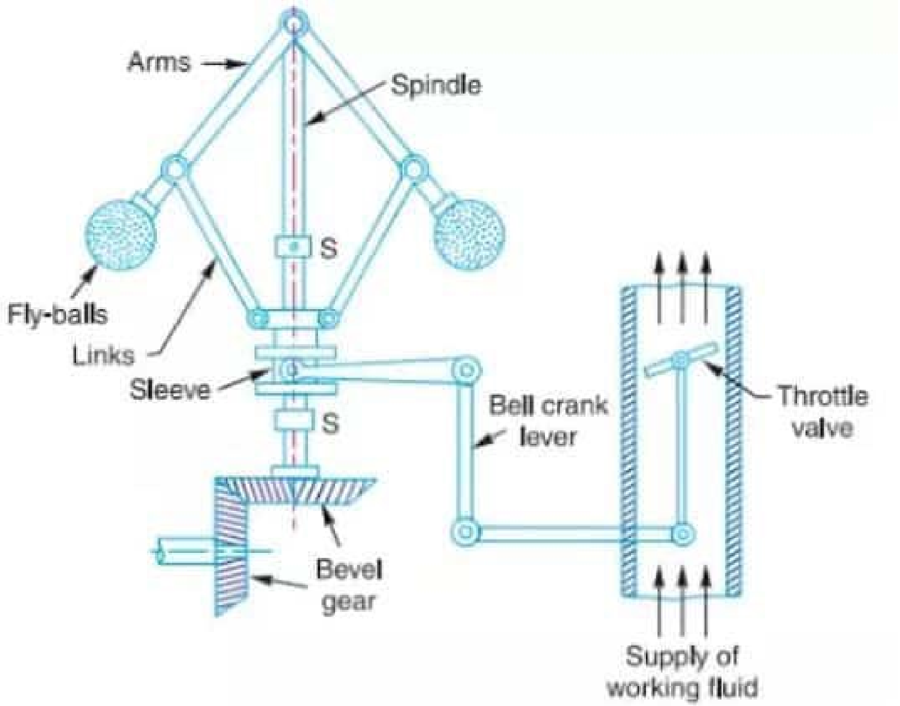
- Closed-loop Control



- Minimize error between observation and reference.

Closed-Loop Control: Centrifugal Governor

- Classic closed-loop control
- Controls the speed of engine by regulating the flow of fuel or working fluid to keep a near-constant speed.



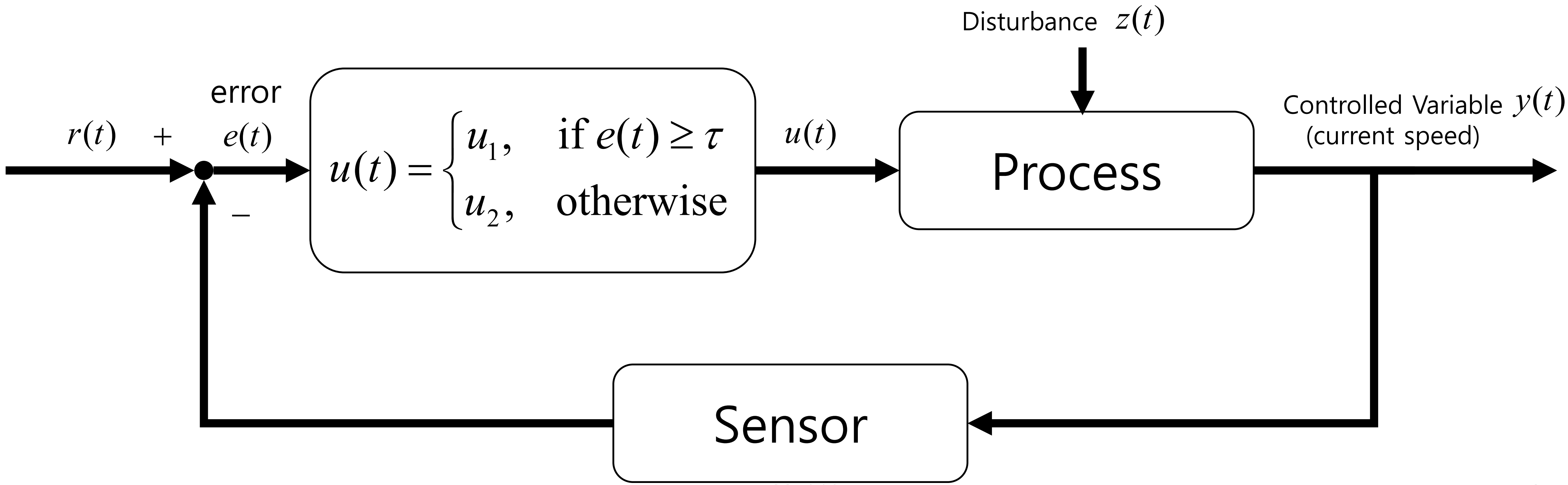
Black-Box Control

Black-box controllers don’t know anything about processing

Bang-Bang Control

- Often used like house heater or conditioner.
- Mathematical formation

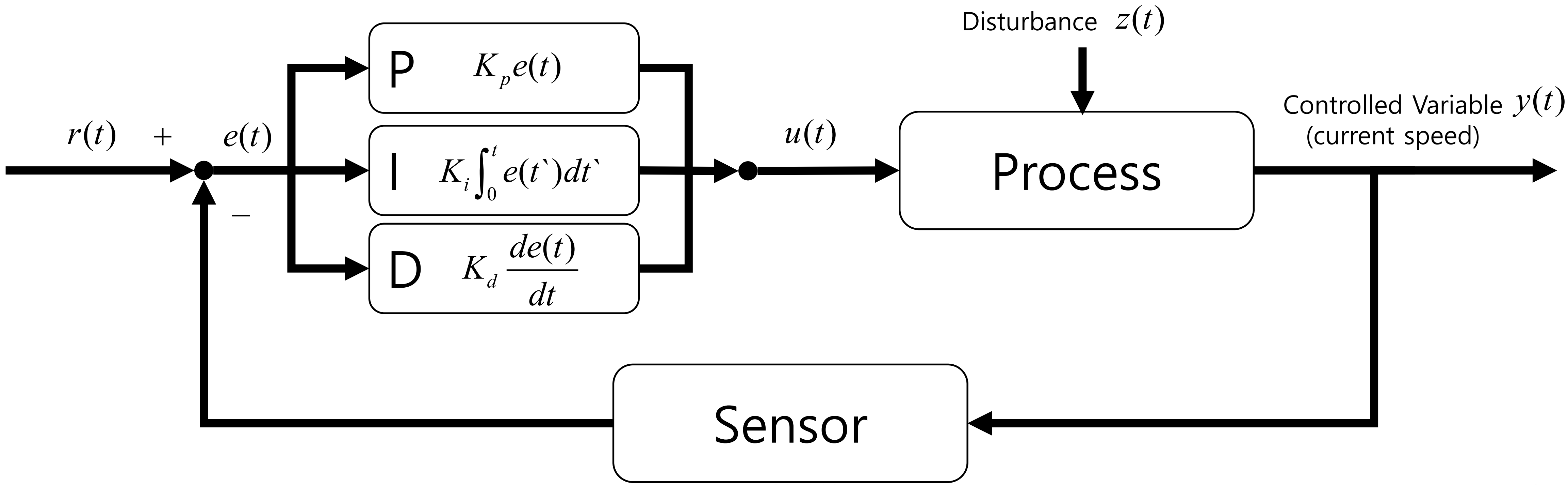
$$u(t) = \begin{cases} u_1, & \text{if } e(t) \geq \tau \\ u_2, & \text{otherwise} \end{cases}$$



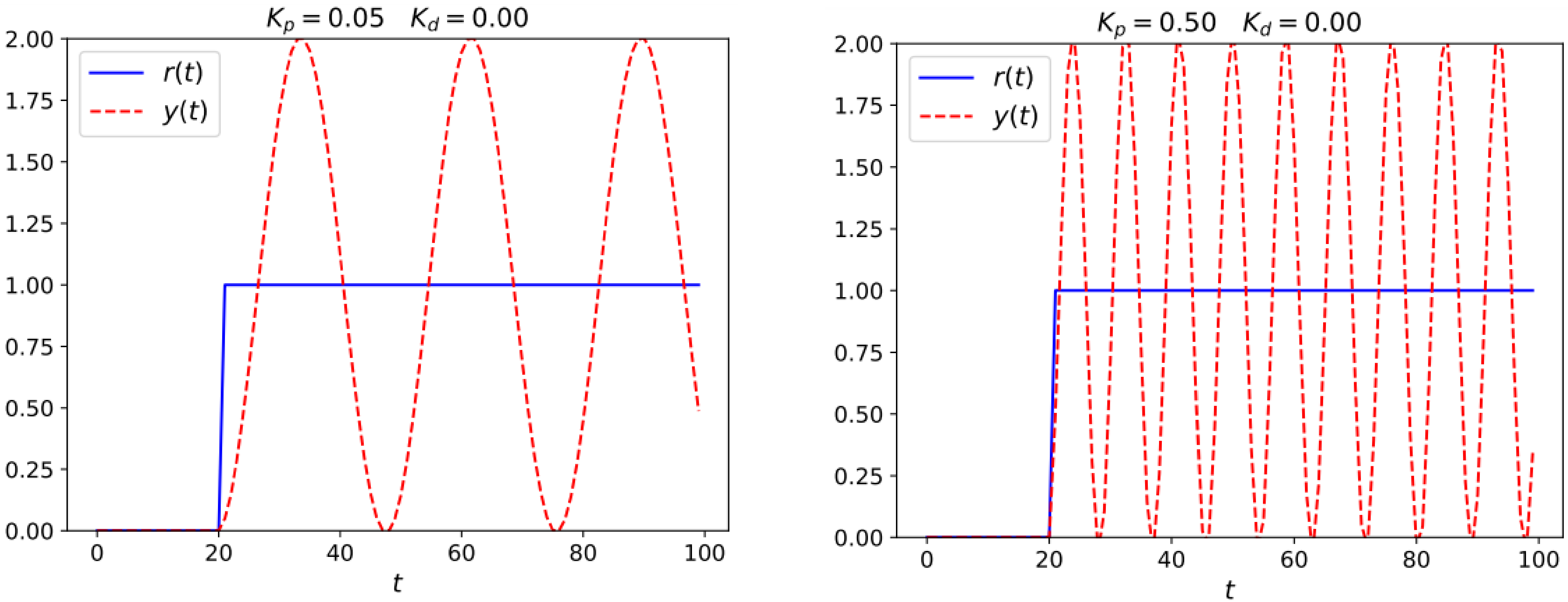
PID Control

- Mathematical formation of Proportional-Integral-Derivative (PID) control

$$u(t) = K_p e(t) + K_i \int_0^t e(t`)dt` + K_d \frac{de(t)}{dt}$$

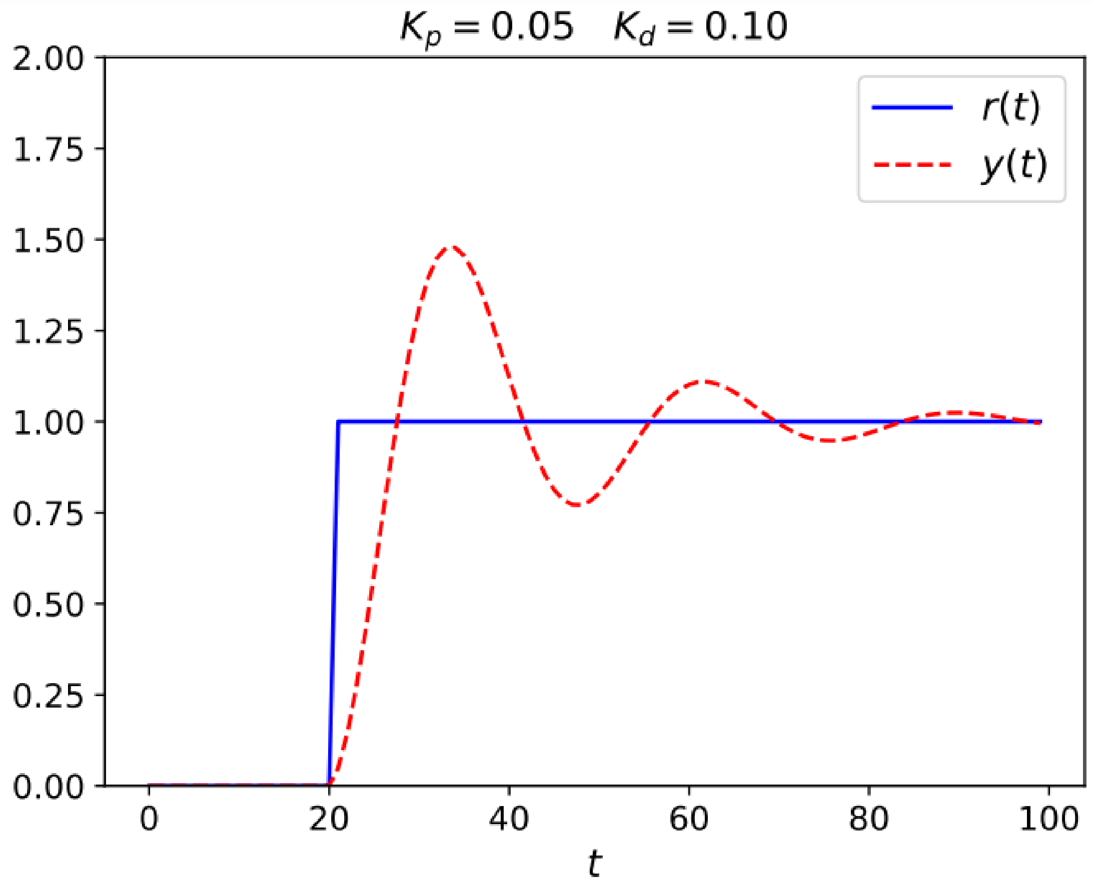


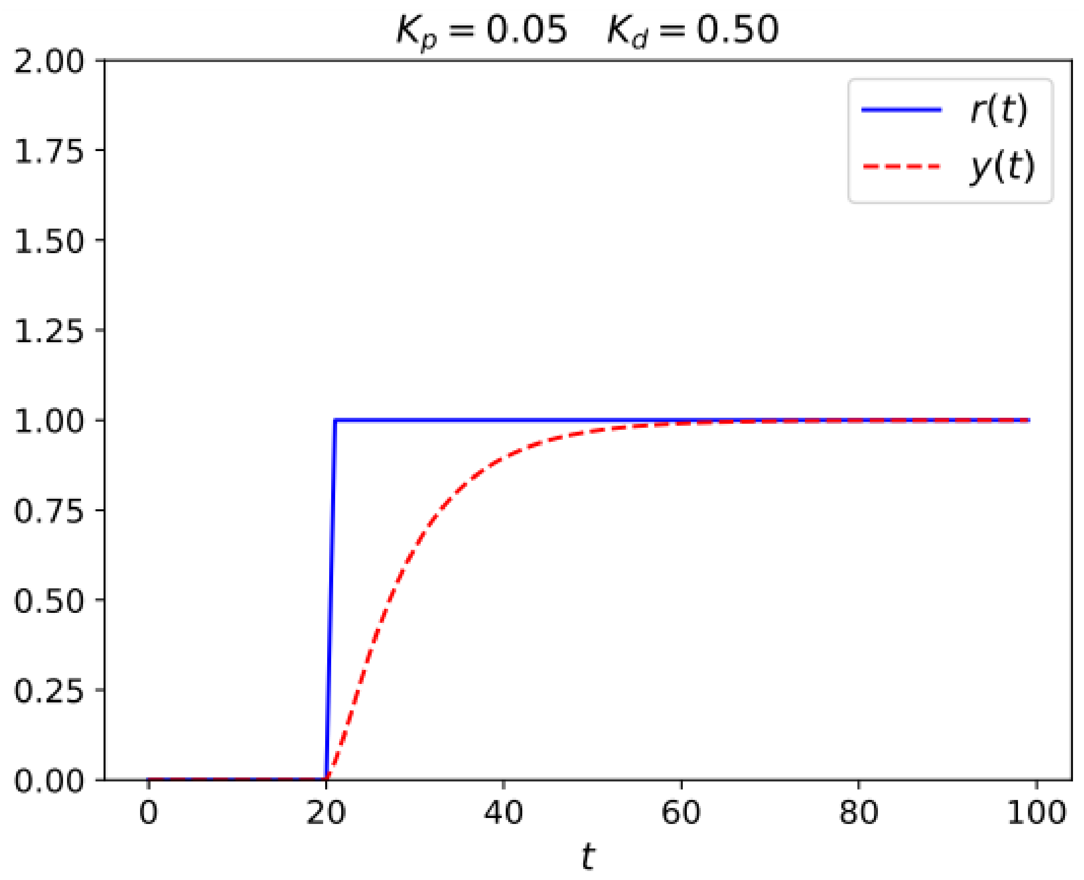
P Control only



- Controlled Variable: Position $y(t) = x(t)$
- Correcting Variable: Acceleration $u(t) = a(t) = \ddot{x}(t)$

PD Control



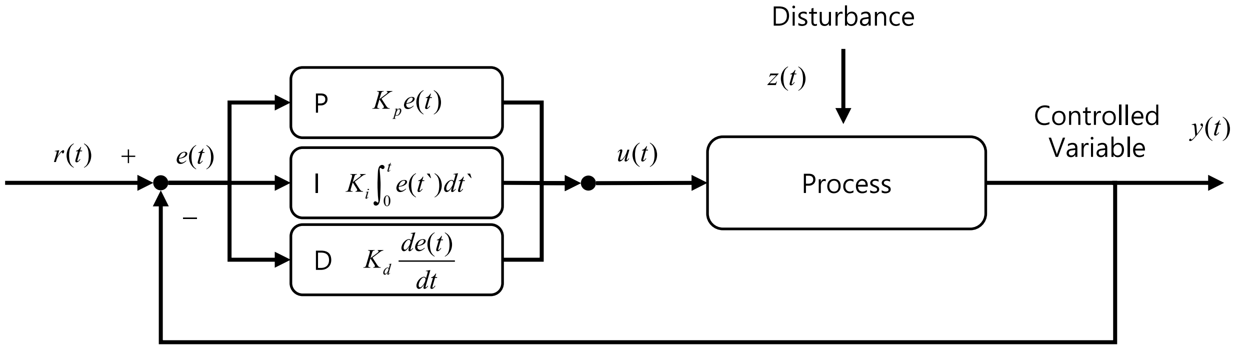


- Controlled Variable: Position $y(t) = x(t)$
- Correcting Variable: Acceleration $u(t) = a(t) = \ddot{x}(t)$

PID Control

- Longitudinal Vehicle Control

$$v(t) = v_{\max}\left(1 - \exp\left(-\theta_1 d(t) - \theta_2\right)\right)$$



$v(t)$: target velocity at time t

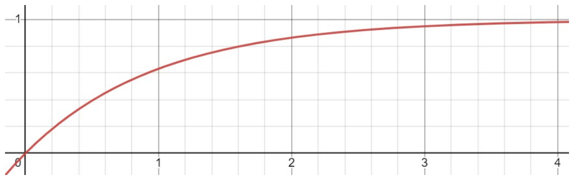
$d(t)$: distance to preceding car

reference variable: $r(t) = v(t) =$ target velocity

correcting variable: $u(t) =$ gas/brake pedal

controlled variable: $y(t) =$ current velocity

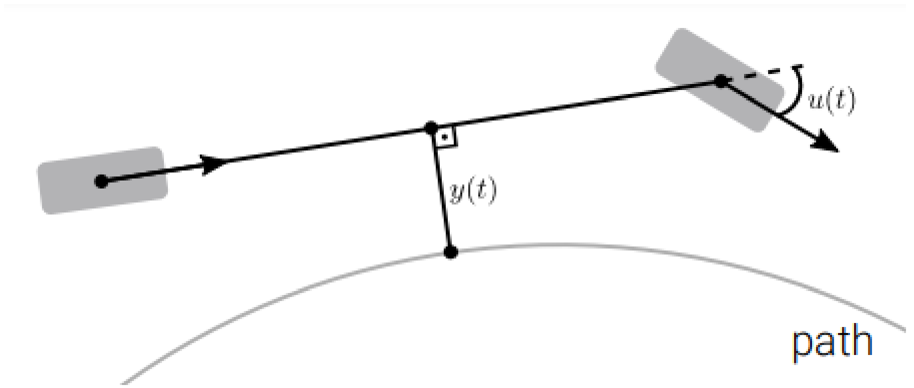
error: $e(t) = v(t) - y(t)$

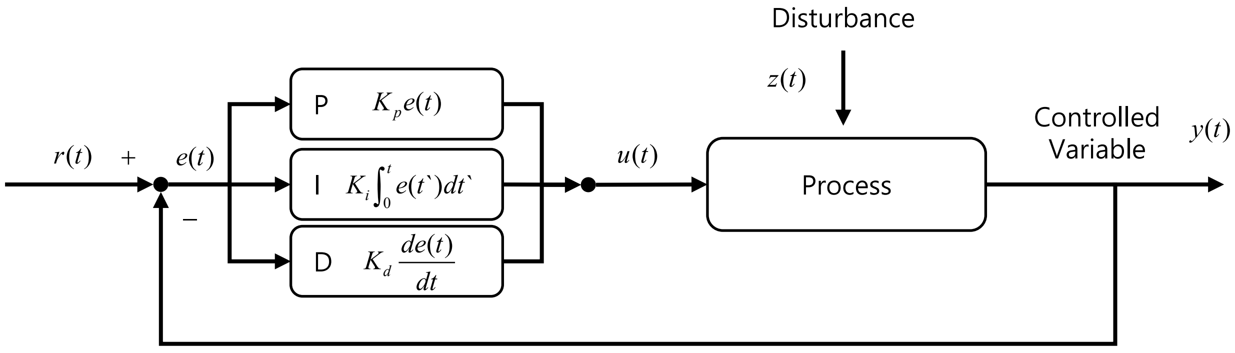


1-exp(-x) graph

PID Control

- Lateral Vehicle Control





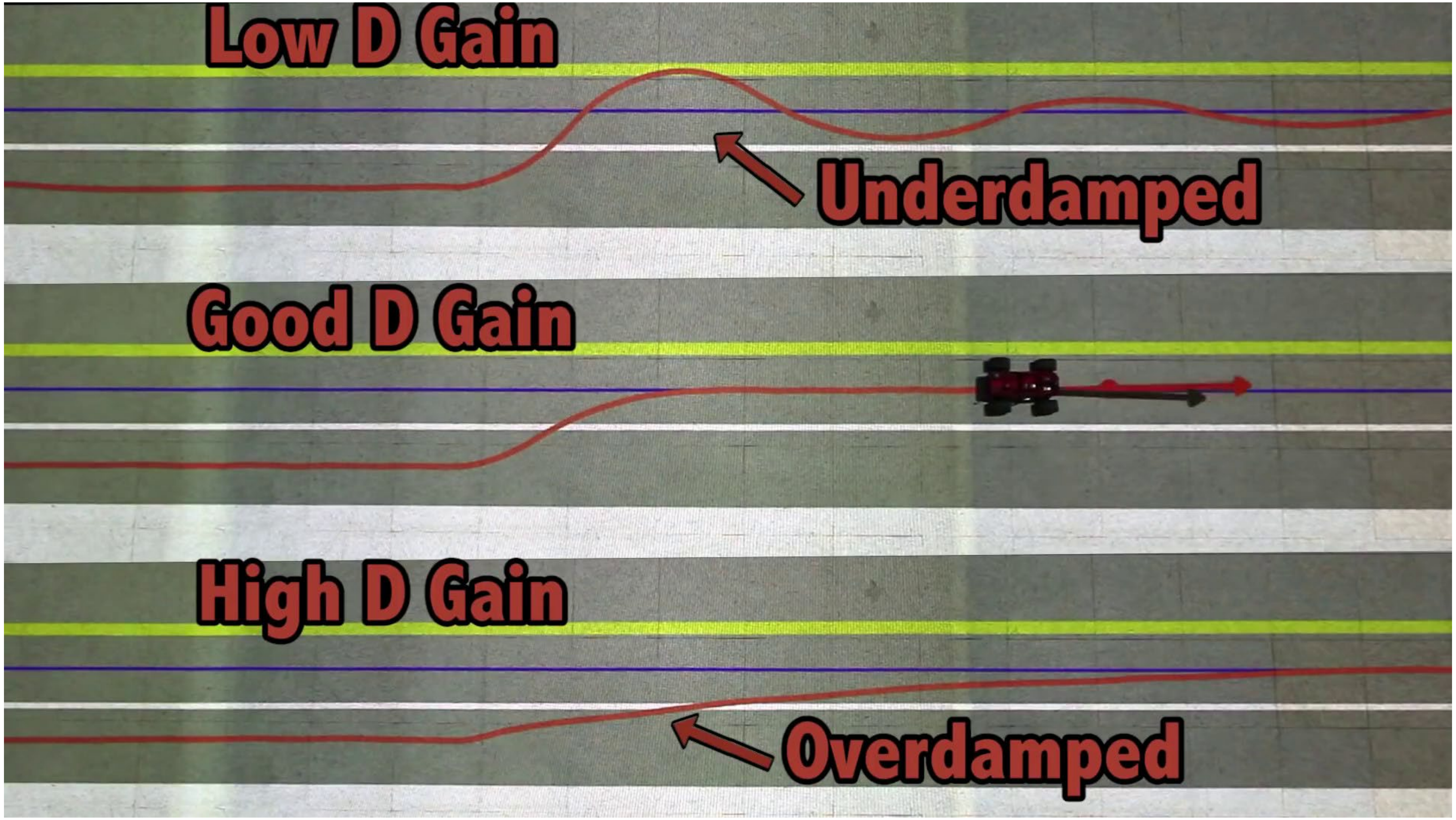
Reference variable: $r(t) = 0 =$ no cross-track error

Correcting variable: $u(t) = \delta =$ steering angle

Controlled variable: $y(t) =$ cross track error

Error: $e(t) = -y(t) =$ cross track error

PID Control



Geometric Control

exploit geometric relationships between the vehicle and the path

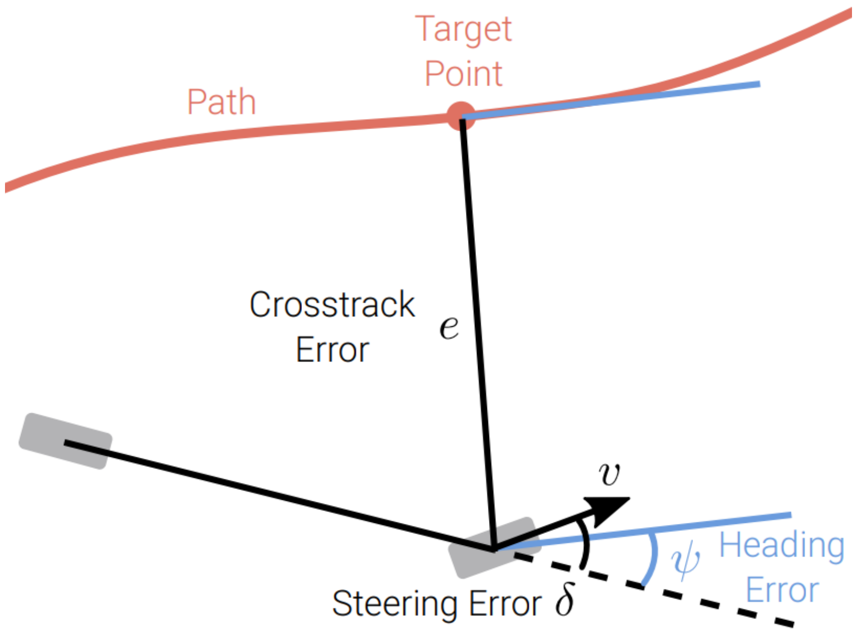
Stanley Control

- Control Law used by Stanley in DARPA Challenge:

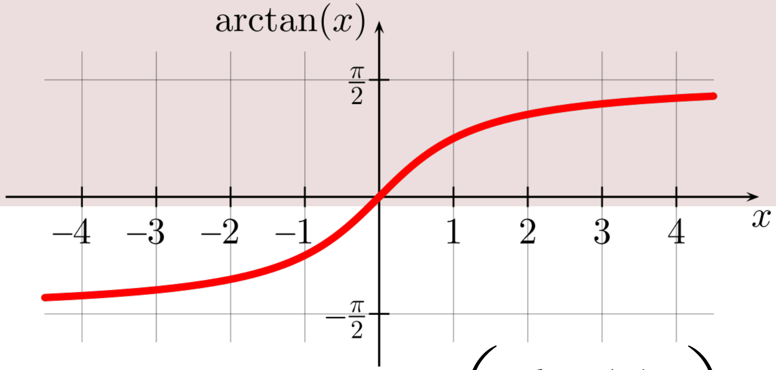
$$\delta = \psi + \tan^{-1}\left(\frac{ke}{v}\right)$$

$v =$ speed, ψ=heading err., e=crosstrack err.

- Combines heading and cross-track error.

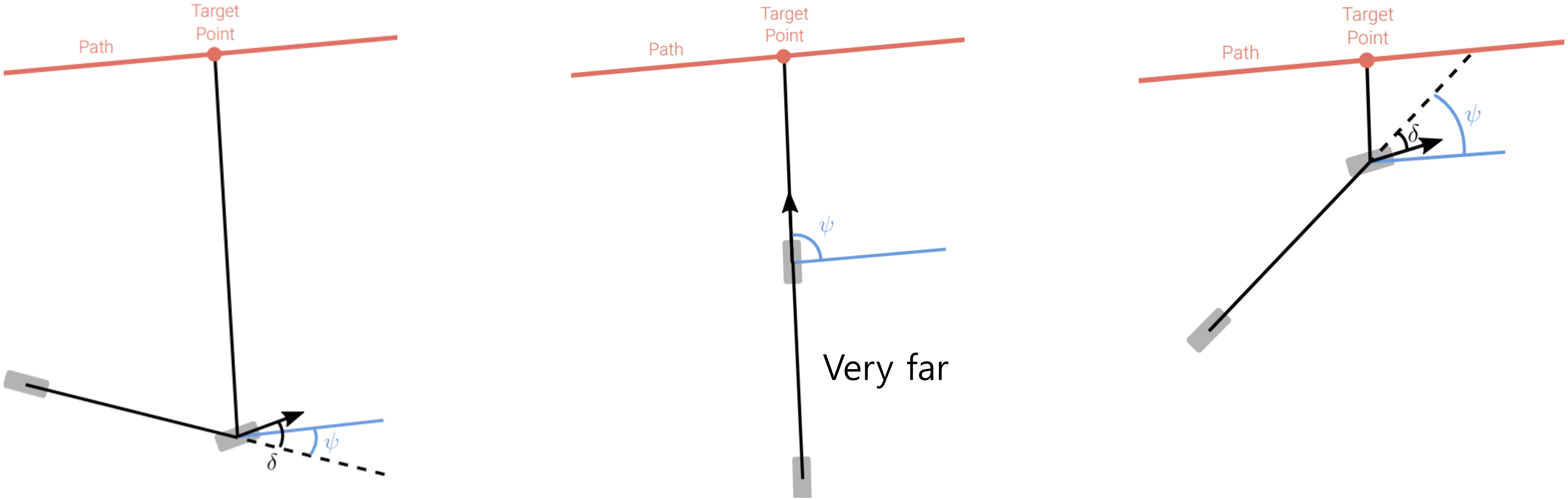


Stanley Control



- First term – heading error
- Second term – considering cross-track error

$$\delta = \psi + \tan^{-1}\left(\frac{ke(t)}{k_s + v}\right)$$



Assignments

Ex1. Implement and Compare Controls

- Implement **bang-bang** control, P control, and PD Control in discrete version.

- Our discrete simulation uses the explicit Euler method.
  - https://en.wikipedia.org/wiki/Euler_method
  - https://research.ncl.ac.uk/game/mastersdegree/gametechnologies/previousinformation/physics2numericalintegrationmethods/2017%20Tutorial%202%20-%20Numerical%20Integration%20Methods.pdf

- Reference position is given and correcting variable is acceleration.

$$v_{i+1} = v_i + a_i \Delta t \qquad r_i = x_{ref}$$

$$p_{i+1} = p_i + v_i \Delta t \qquad p_0 = x_0 \qquad u_i : \text{acceleration}$$

- Describe how each control behave.

Ex 2. Autonomous Driving: Lateral Control

- Template
  - lateral control.py
  - test lateral control.py for testing
- a) Stanley Controller:
  - Read section 9.2 in the Stanley paper http://isl.ecst.csuchico.edu/DOCS/darpa2005/DARPA%202005%20Stanley.pdf
  - Understand the parts of the heuristic control law

$$\delta = \psi + \tan^{-1}\left(\frac{ke}{v + \varepsilon}\right)$$

Ex 2. Autonomous Driving: Lateral Control

- Stanley Controller:
  - Implement controller function given waypoints and speed
    → LateralController.stanley()
  - Orientation error $\psi(t)$ is the angle between the first path segment and the car orientation
  - Cross track error $e(t)$ is distance between desired waypoint at a spline parameter of zero and the position of the car
  - Prevent division by zero by adding as small epsilon
  - Check the behavior of your car

Ex 2. Autonomous Driving: Lateral Control

- c) Damping:
- Damping the difference between the steering command and the steering wheel angle of the previous step

$$\delta = \delta_{SC}(t) - D \cdot \left(\delta_{SC}(t) - \delta(t-1)\right)$$

- Describe the behavior of your car

Ex 2. Autonomous Driving: Longitudinal Control

- Template
  - longitudinal control.py
  - test longitudinal control.py for testing
- PID Controller:
  - Implement a PID control step for gas and braking
  - Use a discretized version:

$$e(t) = v_{\text{target}} - v(t)$$

$$u(t) = K_p e(t) + K_d \left[e(t) - e(t-1) \right] + K_i \left[\sum_{t_i=0}^{t} e(t_i) \right]$$

Ex 2. Autonomous Driving: Longitudinal Control

- PID Controller
  - Due to integral windup, implement an upper bound for integral term.
  - From control signal to gas and brake action values

$$a_{gas}(t) = \begin{cases} 0, & u(t) < 0 \\ u(t), & u(t) \geq 0 \end{cases} \qquad a_{brake}(t) = \begin{cases} 0, & u(t) \geq 0 \\ -u(t), & u(t) < 0 \end{cases}$$

Ex 2. Autonomous Driving: Longitudinal Control

Parameter Search

- Run test lateral control.py and have a look at plots of the target speed and the actual speed
- tune parameters (K_p, K_i, K_d) and (v_{max}, v_{min}, K_v)
- Start with ($K_p = 0.01$, $K_i = 0$, $K_d = 0$) and ($v_{max} = 60$, $v_{min} = 30$, $K_v = 4.5$)
- Only modify a single term at a time!