Fall 2022 ME424 Modern Control and Estimation

Lecture Note 6
Control Design and Testing in Drake with Python

Prof. Wei Zhang
Department of Mechanical and Energy Engineering
SUSTech Institute of Robotics
Southern University of Science and Technology

zhangw3@sustech.edu.cn
https://www.wzhanglab.site/

Outline

- **Short introduction to Drake**
- Example 1: Observer and Controller Design
- Example 2: Cart-Pole Balancing
- From regulation to tracking control
- Example 3: DC Motor speed tracking control

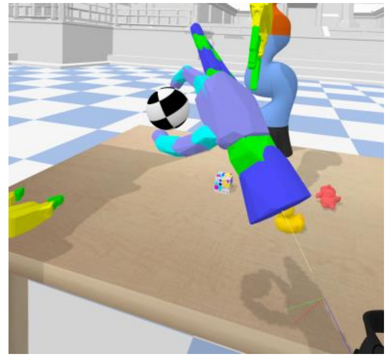
- **What is Simulation?**
  - Real-world physics are often described by functions, ODE or PDE
  - All simulators essentially solve the ODEs and/or PDEs corresponding to a physical process of interest
- **Three pillars of a simulator:**

1. Constructing the differential equations/models

2. Solving differential equations
3. Visualization of the simulation results

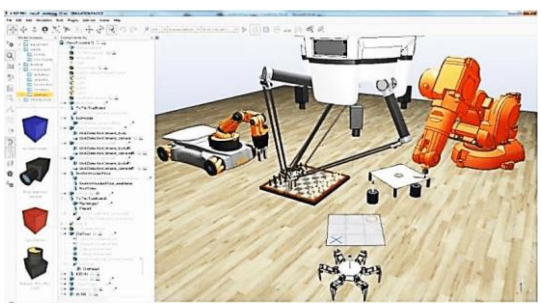
- **Popular simulators in robotics**

Mujoco (Roboti LLC)

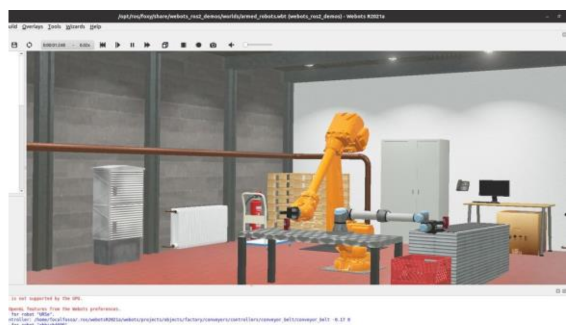


PyBullet (open source)

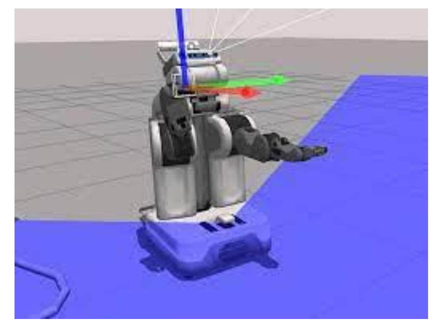
ISAAC (NVIDIA)



V-REP (CoppeliaSim)

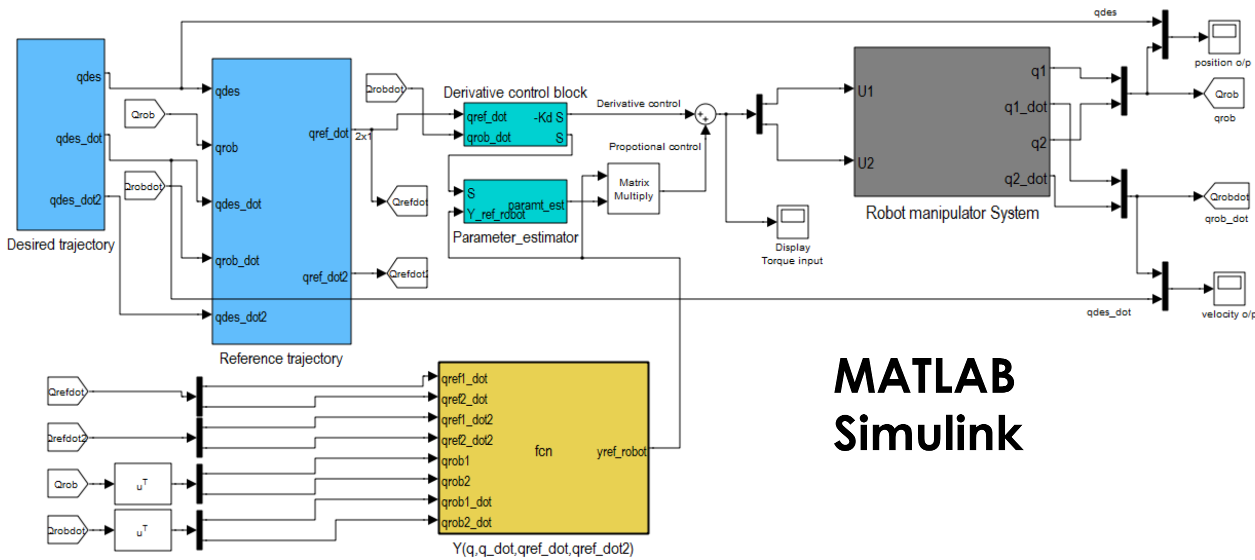


Webot (Cyberbotics）



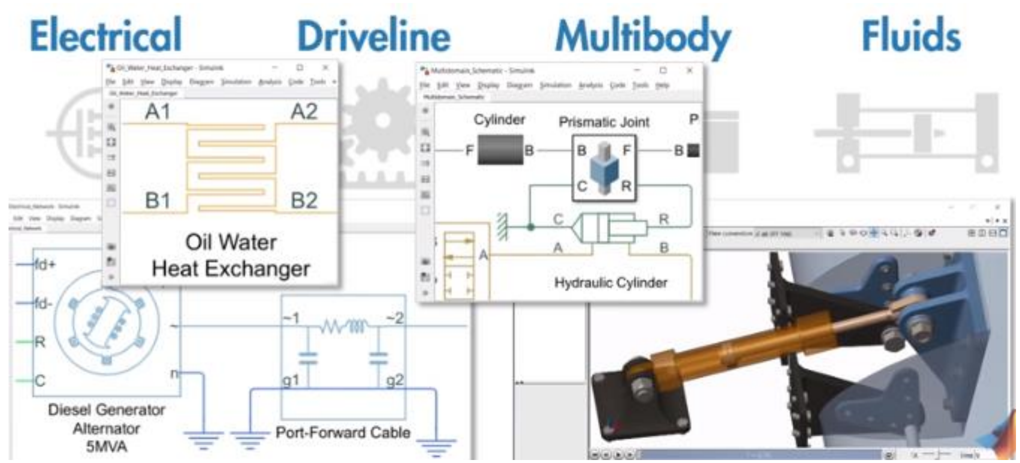
Gazebo

▪ Popular simulators for control systems



MATLAB Simulink

MATLAB Simscape



- **Drake:** Model-Based Design and Verification for Robotics

  - Happy marriage between MATLAB Simulink with and robotic simulators

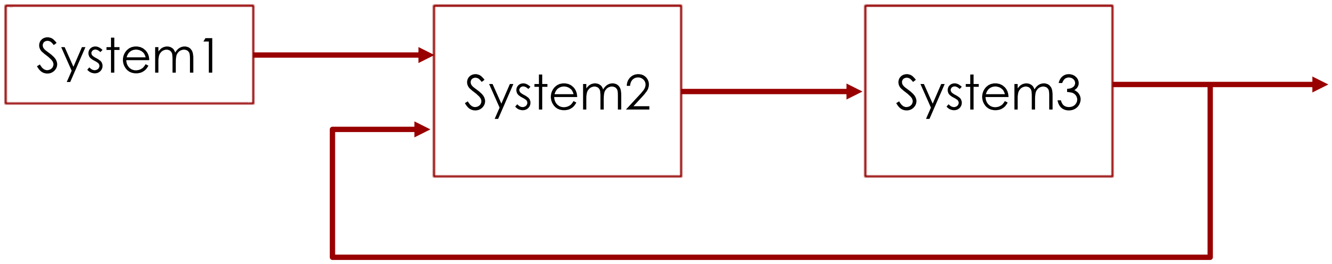
  - Great support for dynamic system modeling, optimization, robotic kinematics and dynamics

  - Very accurate simulation (contact handling)

  - Visualization is not great but good enough

  - Open-source and support Python

▪ Drake: Block Diagram Overview



- **How to define a system block (static or dynamic)?**

- **How to connect blocks to make the overall diagram?**

- **How to simulate?**

▪ Drake: How to Define a Static System?

```
class StaticSysExample(LeafSystem):
    def __init__(self, myParameter):
        LeafSystem.__init__(self)
        self.DeclareVectorInputPort("u1", BasicVector(num_input1))
        self.DeclareVectorInputport("u2", BasicVector(num_input2))
        self.DeclareVectorOutputPort("y", BasicVector(num_output), self.CalcOutputY)

    def CalcOutputY(self, context, output):
        u1 = self.get_input_port(0).Eval(context)
        u2 = self.get_input_port(1).Eval(context)
        y = "your output function"
        output.SetFromVector(y)
```

BasicVector:

- a = BasicVector(3)   # 3-d vector
- a.SetFromVector([arrray]) # from numpy array to Basic vector
- a.CopyToVector()   # from BasicVector to numpy array

▪ Drake: How to Define a Continuous-Time Dynamic System?

```python
class CTSysExample(LeafSystem):
    def __init__(self, myParameter):
        LeafSystem.__init__(self)
        self.DeclareContinuousState(num_state)
        self.DeclareVectorInputPort("u", BasicVector(num_input))
        self.DeclareVectorOutputPort("y", BasicVector(num_output), self.CalcOutputY)

    def DoCalcTimeDerivatives(self, context, derivatives):
        x = context.get_continuous_state_vector().CopyToVector()
        u = self.get_input_port(0).Eval(context)
        xdot = "your vector field"
        derivatives.get_mutable_vector().SetFromVector(xdot)

    def CalcOutputY(self, context, output):
        x = context.get_continuous_state_vector().CopyToVector()
        y = "your output function"
        output.SetFromVector(y)
```

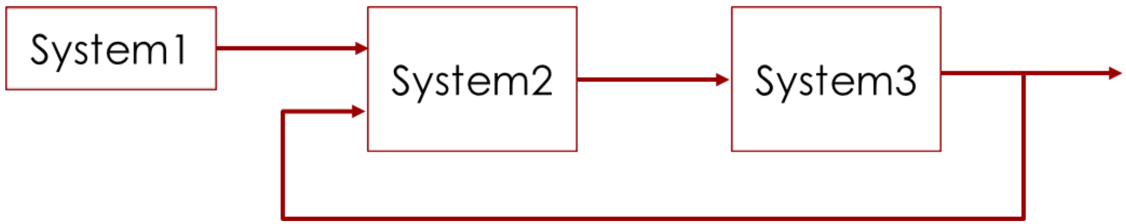
▪ Drake: How to Define a Discrete-Time Dynamic System?

```python
class DTLinearSys(LeafSystem):
    def __init__(self):
        LeafSystem.__init__(self)
        self.DeclareDiscreteState(num_state)
        self.DeclareVectorInputPort("u", BasicVector(num_input))
        self.DeclareVectorOutputPort("y", BasicVector(num_output), self.CalcOutputY)
        self.DeclarePeriodicDiscreteUpdate(0.5)  # dt

    def DoCalcDiscreteVariableUpdates(self, context, events, discrete_state):
        x = context.get_discrete_state_vector().CopyToVector()
        u = self.get_input_port(0).Eval(context)
        xnext = 0.8*x + np.sin(u)
        discrete_state.get_mutable_vector().SetFromVector(xnext)

    def CalcOutputY(self, context, output):
        x = context.get_discrete_state_vector().CopyToVector()
        u = self.get_input_port(0).Eval(context)
        y = x + 2*u
        output.SetFromVector(y)
```

▪ Drake: Block Diagram Construction

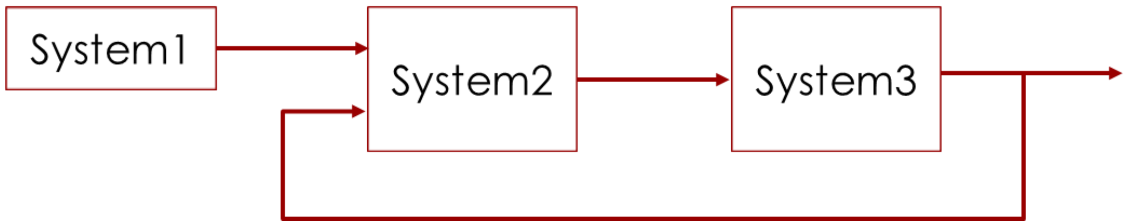



```
builder = DiagramBuilder()
Sys1 = builder.AddSystem(Sys1)
Sys2 = builder.AddSystem(Sys2)
Sys3 = builder.AddSystem(Sys3)

builder.Connect(Sys1.get_output_port(0), Sys2.get_input_port(0))
builder.Connect(Sys2.get_output_port(0), Sys3.get_input_port(0))
builder.Connect(Sys3.get_output_port(0), Sys2.get_input_port(1))

logger_output = LogOutput(Sys3.get_output_port(0), builder)
diagram = builder.Build()
```

Drake: Simulate a Block Diagram




```
simulator = Simulator(diagram)
simulator.set_target_realtime_rate(1)
context = simulator.get_mutable_context()
context.SetContinuousState(CT_x0)
context.SetDiscreteState(DT_x0)
simulator.AdvanceTo(sim_time)
```

Regarding **context** class:

- get_continuous_state_vector()
- get_discrete_state_vector()
- get_mutable_continuous_state_vector()
- get_mutable_discrete_state_vector()
- SetContinuousState(..)
- SetDiscreteState(..)

- **Drake: Simple Simulation Examples**

Outline

- Short introduction to Drake
- **Example 1: Observer and Controller Design**
- Example 2: Cart-Pole Balancing
- From regulation to tracking control
- Example 3: DC Motor speed tracking control

Example 1: design output feedback controller for plant

$$A_c = \begin{bmatrix} 33 & -60 \\ 20 & -33.2 \end{bmatrix}, \qquad B_c = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \qquad C_c = [2 \quad 1]$$

- **Step 1:** Discretization: e.g. with sampling time T = 0.01

$$A_d = (I + A_c T) = \begin{bmatrix} 1.33 & -0.6 \\ 0.2 & 0.668 \end{bmatrix}, \qquad B_d = B_c T = \begin{bmatrix} 0.01 \\ 0.01 \end{bmatrix}, \qquad C_d = C_c = [2 \quad 1]$$

- **Step 2:** Select desired closed-loop eigs (suppose we want the continuous time poles: $s_{1,2}$ ={-2+j, -2-j}

- **Step 3:** Design feedback gain K

- **Step 4:** Observer eigs: suppose we want: observer_$s_{1,2}$ ={-8+j, -8+j}

- **Step 5:** Observer gain L

  Observer dynamical system:

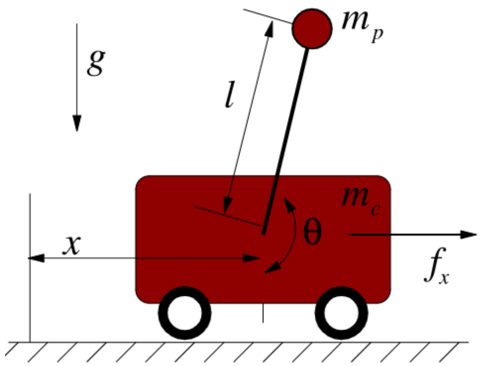
$$\hat{x}_{k+1} = A\hat{x}_k + Bu_k + L(y_k - C\hat{x}_k) = (A - LC)\hat{x}_k + [B \;\; L]\begin{bmatrix} u_k \\ y_k \end{bmatrix}$$

- **Simulation Testing**

Outline

- Short introduction to Drake
- Example 1: Observer and Controller Design
- **Example 2: Cart-Pole Balancing**
- From regulation to tracking control
- Example 3: DC Motor speed tracking control

- Cart-Pole Model



$$\ddot{x} = \frac{1}{m_c + m_p \sin^2\theta}\left[f_x + m_p \sin\theta(l\dot{\theta}^2 + g\cos\theta)\right]$$

$$\ddot{\theta} = \frac{1}{l(m_c + m_p\sin^2\theta)}\left[-f_x\cos\theta - m_p l\dot{\theta}^2\cos\theta\sin\theta - (m_c + m_p)g\sin\theta\right]$$

Outline

- Short introduction to Drake
- Example 1: Observer and Controller Design
- Example 2: Cart-Pole Balancing
- **From regulation to tracking control**
- Example 3: DC Motor speed tracking control

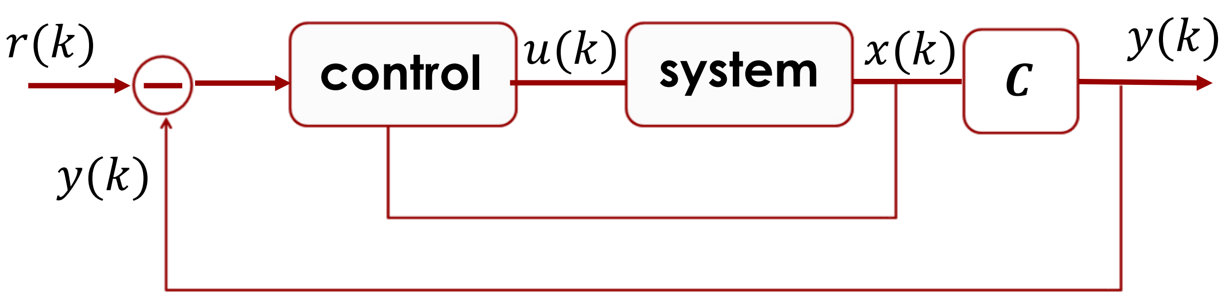
- **Robust tracking problem:**

$$x(k+1) = Ax(k) + Bu(k) + B_d d(k)$$
$$y(k) = Cx(k)$$

  - $d(k)$: disturbance entering the systems

- **Goal:** design u to make output $y(k)$ track a reference $r(k)$

- To simplify discussion, we assume:
  - $r(k)$ and $y(k)$: scalar
  - $u(k)$: full state feedback (add observer if full state is not available)



- **Illustrating example**: consider a simple scalar system

$$x(k+1) = a\,x(k) + u(k), \qquad y(k) = x(k)$$

Suppose tracking reference: $r(k) = r \neq 0$

  - Linear feedback doesn't work: $u(k) = -Kx(k) \Rightarrow \; x(k+1) = (a-K)x(k)$

  - Can we compute the correct input? E.g. assume $|a| < 1$, then $x(k) = a^k x_0 + \sum_{j=0}^{k-1} a^{k-j-1} u(j)$

- Challenges for more general tracking problems:

$$x(k+1) = Ax(k) + Bu(k) + B_d d(k)$$
$$y(k) = Cx(k)$$

  - $x(k) \in R^n$ can be multi-dimensional ($a \rightarrow A$), so can input $u(k)$
  - System may be unstable
  - Uncertainties:
    - reference $r(k)$ may not be know a priori
    - we have nontrivial **unknown** disturbance $d(k)$
  - How to improve transient performance (track promptly)

- Introduce an ``integral state'': $z(k+1) = z(k) + r(k) - y(k)$

- Extended state space: $\tilde{x}(k) = \begin{bmatrix} x(k) \\ z(k) \end{bmatrix}$

- Feedback control: $u(k) = [K_x \quad K_z] \begin{bmatrix} x(k) \\ z(k) \end{bmatrix}$

- CL dynamics: $\tilde{x}(k+1) = \begin{bmatrix} A & 0 \\ -C & 1 \end{bmatrix} \tilde{x}(k) + \begin{bmatrix} B \\ 0 \end{bmatrix} u(k) + \begin{bmatrix} B_d d(k) \\ r(k) \end{bmatrix}$

- Under mild conditions: $(\tilde{A}, \tilde{B})$ is controllable so we can design $\widetilde{K}$ such that $\tilde{A} + \tilde{B}\widetilde{K}$ has desired eigenvalues

- Closed-loop dynamics: $\tilde{x}(k+1) = \left(\tilde{A} + \tilde{B}\widetilde{K}\right)\tilde{x}(k) + \begin{bmatrix} B_d d(k) \\ r(k) \end{bmatrix}$

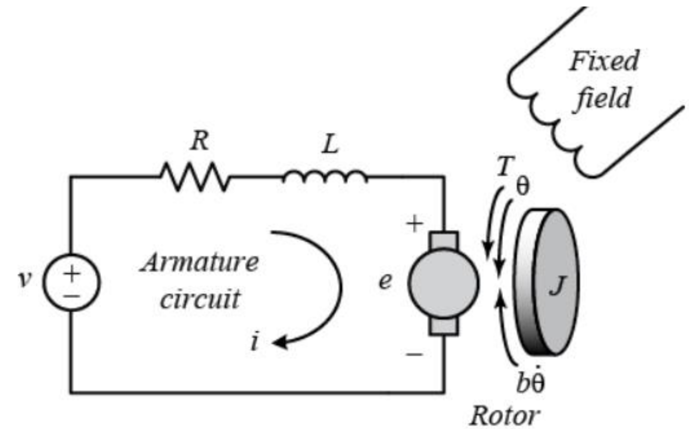
- For constant or slowly changing $d(k) \approx d, r(k) \approx r$,
  - the extended state $\tilde{x}(k)$ converges to a finite value.
  - Thus, $z(k+1) = z(k) \Rightarrow$ error becomes zero

- Proof and Discussions

Outline

- Short introduction to Drake
- Example 1: Observer and Controller Design
- Example 2: Cart-Pole Balancing
- From regulation to tracking control
- **Example 3: DC Motor speed tracking control**

- DC Motor Speed Tracking Control Example



- Summary