MEEN 432 –Automotive Engineering

Fall 2026

Instructor: Dr. Arnold Muyshondt

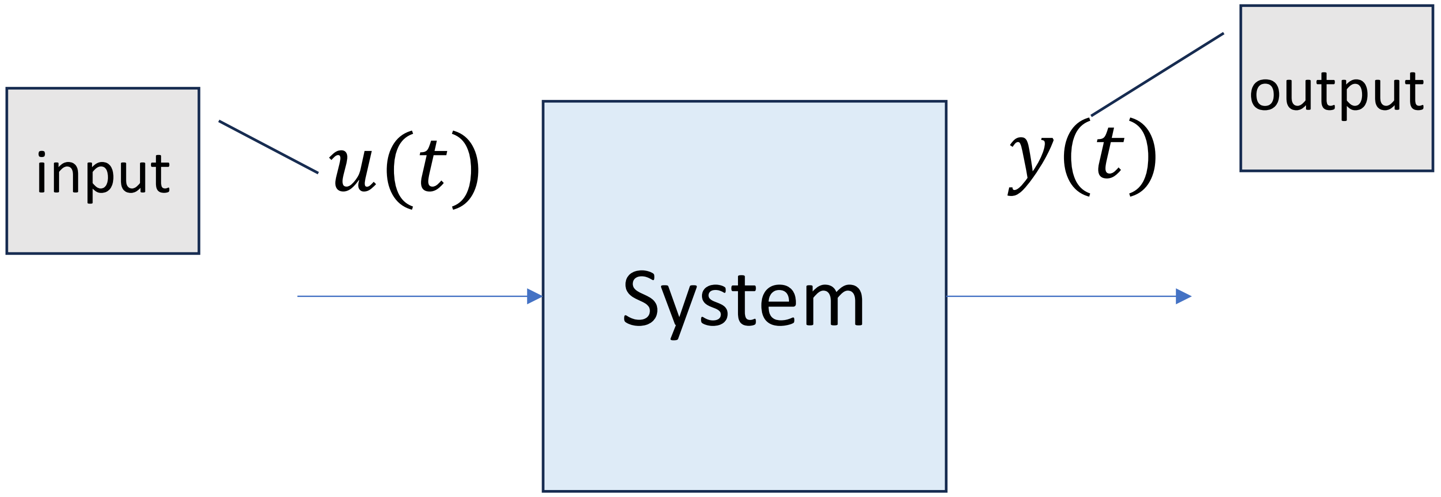
Acknowledgement: Most of the material for this class was developed by Dr. Swami Gopalswamy

Lecture 2: Simulation of Dynamic Systems

- Numerical simulation of a dynamic system
- Project 1 Roll-out

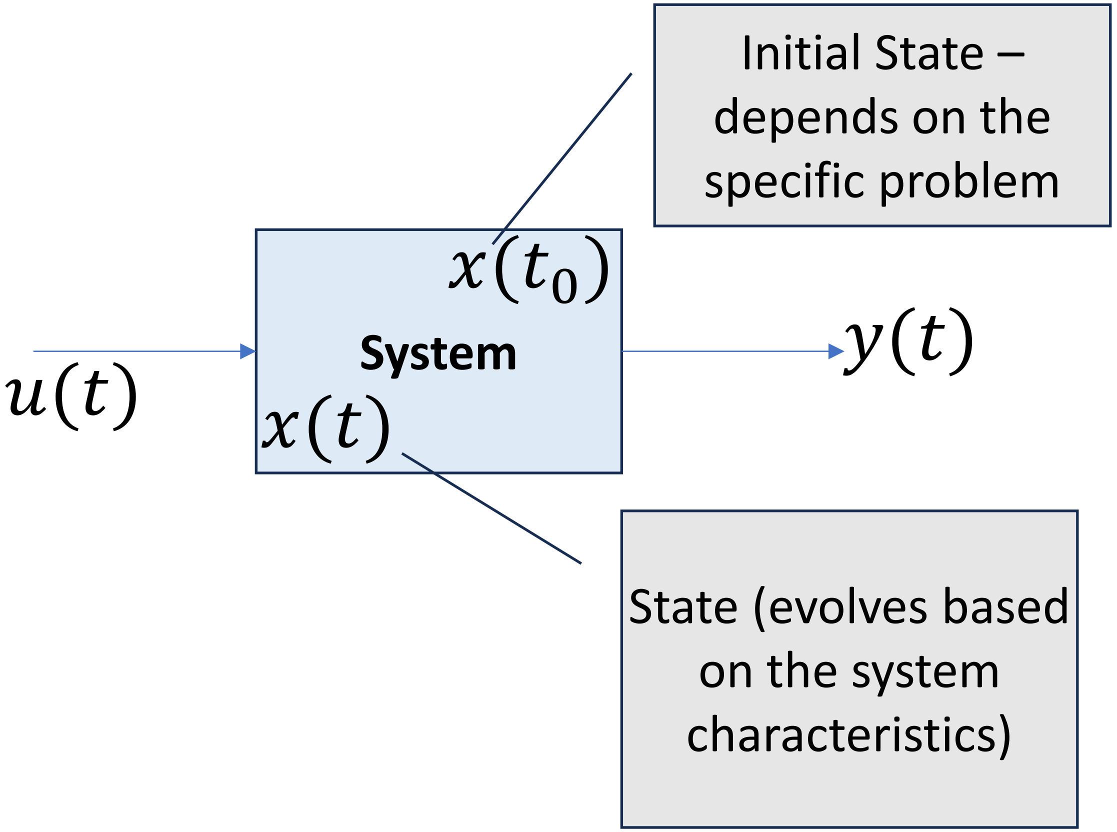
Dynamic System (1/3)

- Any system has inputs $u(t)$ that impact the system, and outputs $y(t)$ that we care to measure or observe
- (Recall MEEN 432)



Dynamic System (2/3)

- A dynamic system is characterized by the evolution in time of a "state"
  - The state is captured by a variable $x(t)$ (which could be a scalar, or a vector in general), that defines the status of a system at the instant t
  - At any time, the output depends on the state of the system, and (potentially) the input
    - $y(t) = h(x(t), u(t))$ Often it depends only on the state (e.g. $y(t) = h(x(t))$)



Dynamic System (3/3)

- An important property of the state: It defines everything you need to know about the system, independent of the history
  - If you know the state $x(t_0)$ at time t_0, then the future states depend ONLY on $x(t_0)$ (the initial condition), and "inputs" $u(t)$ for $t \geq t_0$
  - A state is considered "memory" (sometimes, loosely, memory variable)
- Many a time, a system's output depends only on the output (i.e. no memory)
  - $y = f(u)$
  - Such as system is also called "stateless"

State Evolution

- Continuous Dynamic Systems
  - $\dot{x} = f(x, u)$ → Ordinary Differential Equation (ODE)
  - $g(x, \dot{x}, u) = 0$ → Differential Algebraic Equation (DAE)
  - In this class we will consider only ODEs
    - In general, most DAEs can be converted to ODEs
- Discrete Dynamic Systems
  - $x_{k+1} = x_k + f_D(x_k, u_k)$ → Difference Equation (DE)
  - $g_D(x_{k+1}, x_k, u_k) = 0$ → Implicit difference equation
  - In this class, we will consider DEs
  - k represents time: $t(k+1) = t(k) + \Delta t(k)$, with $\Delta t(k) > 0$ for all k
    - Often, (but not always), $\Delta t(k) = \text{constant}$

Simulation of Discrete Dynamic Systems

- This is straightforward application of the DE in a recursive fashion!
- Given:
  - Initial Condition $x(0) = x_0$
  - Input time profile: u_k for all time instants $k \in [0, k_f]$ Here k_f represents the final time instant
  - System Dynamics: $x_{k+1} = x_k + f_D(x_k, u_k)$
- Find:
  - x_k for all time $k \in [0, k_f]$
- Iterative Solution:
  - $x(0) = x_0; k = 0$
  - While $k < k_f$
    - $x_{k+1} = x_k + f_D(x_k, u_k);$
    - $k = k + 1$

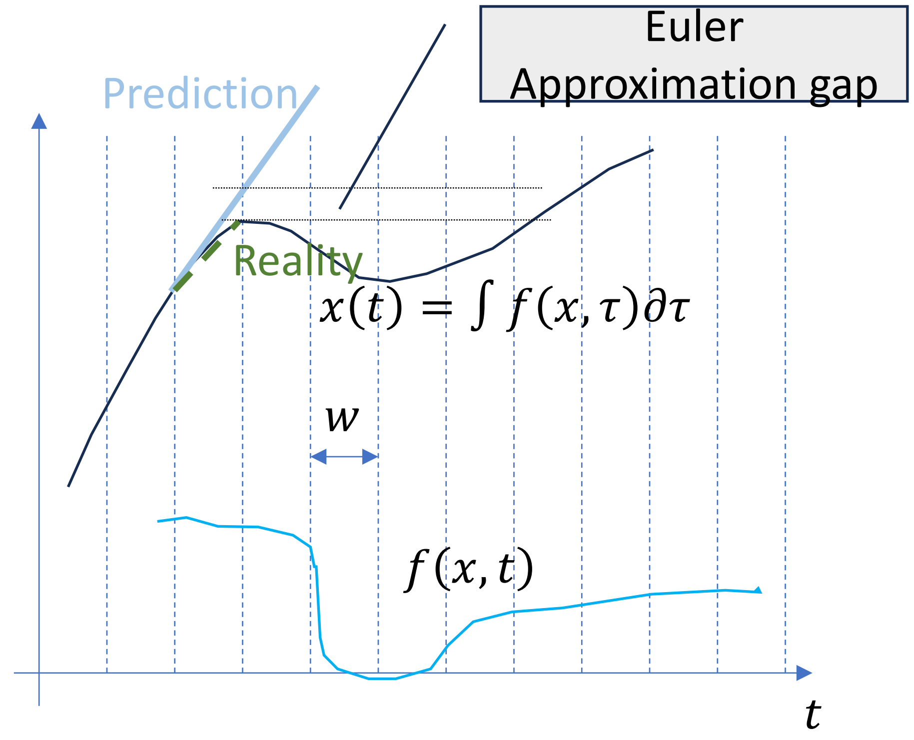
Simulation of Continuous Dynamic Systems

- Continuous Systems:
  - Given:
    - Initial Condition $x(0)$
    - Input time profile: $u(t)$ for all time $t \in [0, t_f]$ Here t_f represents the final time
    - System Dynamics: $\dot{x} = f(x, u)$
  - Find:
    - $x(t)$ for all time $t \in [0, t_f]$

- Solution:
  - $x(t) = x(0) + \int_0^t f(x(\tau), u(\tau))d\tau$
  - Solving the above equation analytically is not straightforward
  - Can we solve it numerically, (but potentially approximately)?
  - Yes. Inspired by the solution to discrete systems, we can capture the solution at discrete time instants if we can approximate the term inside the integral above:
    - Define $t_{k+1} = t_k + \Delta T;\, t_0 = 0;\, x_k = x(t_k)$
    - Find $f_D(x_k, u_k) \approx \int_{t_k}^{t_{k+1}} f(x(\tau), u(\tau))d\tau$

Integration (1/4)

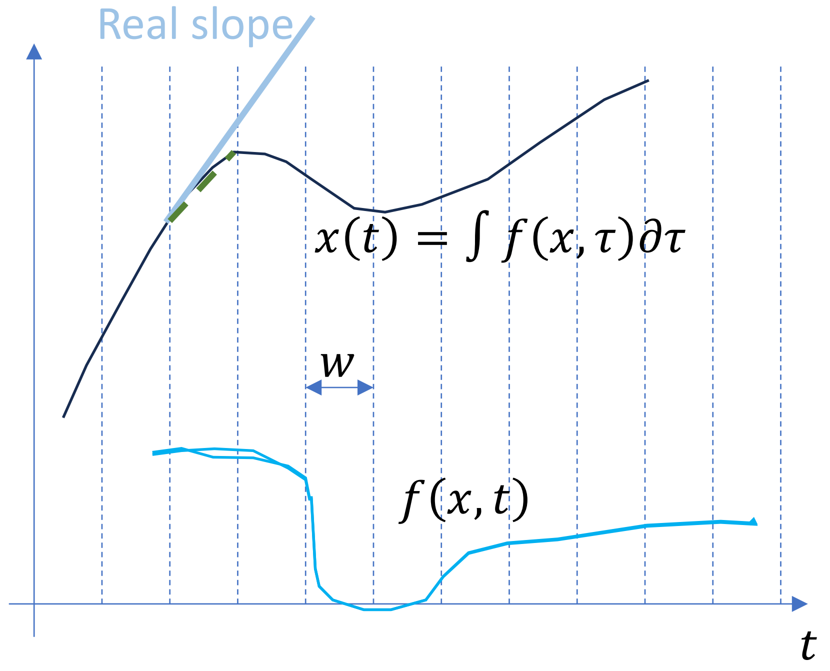
- Consider the time interval k given by:
  - $t_k \leq t \leq t_{k+1}$ with $\Delta T \coloneqq t_{k+1} - t_k$
- IF we can assume that $f\big(x(t), u(t)\big) \approx \bar{f}(x_k, u_k)$ i.e. can be approximated by a constant between sampling time instants,
- Then we can write:
  - $x(t+1) = x(t) + \Delta t\, \bar{f}(x_k, u_k)$
  - We can use the DE algorithm from here
- The questions are:
  - When can we make this assumption?
  - What is the best value of the constant $\bar{f}(x_k, u_k)$



- Small ΔT
- $\bar{f}(x_k, u_k) \coloneqq f(t_k)$? Forward, Euler assumption

Integration (2/4)

- $\bar{f}(x_k, u_k) \coloneqq f(t_k)$? Forward, Euler assumption leads to the Euler Integration technique
- There are many other integration techniques:
  - Euler, Runge-Kutta, …
  - Trade-off Computational Speed vs Accuracy



Integration (3/4)

- In general. The integration time step Δt needs to be small for better approximation (i.e. higher accuracy)
  - How small depends on how fast the system dynamics are. (Smaller step size ➔ More iterations ➔ longer simulation time!)
  - A “stiffer” system usually exhibits faster changes ➔ requires a smaller integration time step
    - System stiffness characterized by the “eigenvalues” of the linearized system dynamics

Integration (4/4)

- Integration methods can be characterized by whether Δt is a constant, or if it can be varied.
  - Variable time step methods automatically determine the “stiffness” of the system, and automatically adjust the integration step size to achieve fast simulation without losing accuracy

Project 0 – Due Jan 18th , 11:59 PM

1. Familiarize yourself with CANVAS for the class.
2. Make sure to get access to MATLAB/Simulink. **Make sure to install the following toolboxes:**
   - MATLAB
   - Simulink
   - Stateflow
   - Automated Driving
3. Work through the MATLAB/Simulink Tutorials
   - MATLAB: https://matlabacademy.mathworks.com/details/matlabonramp/gettingstarted
   - Simulink: https://matlabacademy.mathworks.com/details/simulinkonramp/simulink
   - Upload Onramp certificates to "Project0_Onramp" CANVAS Assignment called Project 0 by **Jan 18th, 11:59 pm**

Project 1 Week 1 – Due Jan 24, 11:59 PM

1. Begin looking at Project 1 documentation now!
2. Do Part 1 of the project, with the following subset for item 2 of Part 1.
    - 2.a (w_0 = 1.0) [rad/s]
    - 2.b (J = 1) [kg m^2]
    - 2.c (b = 1) [N-m/(rad/s)]
    - 2.d.i (A = 1) [N-m]
    - 2.e.i Fixed Time Step: (0.001, 0.1, 1 s)
        - Euler Solver
        - Runge Kutta 4th Order Solver
    - 2.f Total simulation time of 25 seconds
3. Upload to **"Project 1"** Assignment in CANVAS
    - Make sure to add a **README.file** explaining how to run your scripts

Tentative Schedule for Project Deliverables

- Project 0 – due Jan 18 11:59 PM
- Project 1 – due Jan 31 11:59PM
- Project 2 – due Feb 28 11:59 PM
- Project 3 – due Mar 28 11:59 PM
- Project 4 (team) - due April 18 11:59 PM
- Project 4 (individual) – due April 25 11:59 PM