Lecture 5: Basic Dynamical Systems

CS 344R: Robotics
Benjamin Kuipers
Dynamical Systems

- A *dynamical system* changes continuously (almost always) according to
  \[ \dot{x} = F(x) \quad \text{where} \quad x \in \mathbb{R}^n \]
- A *controller* is defined to change the coupled robot and environment into a desired dynamical system.
  \[ \dot{x} = F(x,u) \]
  \[ y = G(x) \quad \dot{x} = F(x,H_i(G(x))) \]
  \[ u = H_i(y) \]
In One Dimension

- Simple linear system
  \[ \dot{x} = kx \]
- Fixed point
  \[ x = 0 \quad \Rightarrow \quad \dot{x} = 0 \]
- Solution
  \[ x(t) = x_0 e^{kt} \]
  - Stable if \( k < 0 \)
  - Unstable if \( k > 0 \)
In Two Dimensions

• Often, position and velocity:

\[ \mathbf{x} = (x, v)^T \quad \text{where} \quad v = \dot{x} \]

• If actions are forces, causing acceleration:

\[ \dot{\mathbf{x}} = \begin{pmatrix} \dot{x} \\ \dot{v} \end{pmatrix} = \begin{pmatrix} \dot{x} \\ \ddot{x} \end{pmatrix} = \begin{pmatrix} v \\ \text{forces} \end{pmatrix} \]
The Damped Spring

- The spring is defined by Hooke’s Law:
  \[ F = ma = m \ddot{x} = -k_1 x \]
- Include damping friction
  \[ m \ddot{x} = -k_1 x - k_2 \dot{x} \]
- Rearrange and redefine constants
  \[ \ddot{x} + b \dot{x} + cx = 0 \]

\[
\begin{bmatrix}
\dot{x} \\
\dot{v}
\end{bmatrix} = 
\begin{bmatrix}
\dot{x} \\
\ddot{x}
\end{bmatrix} = 
\begin{bmatrix}
v \\
-b \dot{x} - cx
\end{bmatrix}
\]
The Linear Spring Model

\[ \ddot{x} + b\dot{x} + cx = 0 \quad c \neq 0 \]

- Solutions are:
  \[ x(t) = Ae^{r_1 t} + Be^{r_2 t} \]

- Where \( r_1, r_2 \) are roots of the characteristic equation
  \[ \lambda^2 + b\lambda + c = 0 \]

  \[ r_1, r_2 = \frac{-b \pm \sqrt{b^2 - 4c}}{2} \]
Qualitative Behaviors

\[ r_1, r_2 = \frac{-b \pm \sqrt{b^2 - 4c}}{2} \]

- \( \text{Re}(r_1), \text{Re}(r_2) < 0 \) means stable.
- \( \text{Re}(r_1), \text{Re}(r_2) > 0 \) means unstable.
- \( b^2 - 4c < 0 \) means complex roots, means oscillations.
Generalize to Higher Dimensions

- The characteristic equation for \( \dot{x} = Ax \) generalizes to \( \det(A - \lambda I) = 0 \)
  - This means that there is a vector \( \mathbf{v} \) such that \( A\mathbf{v} = \lambda \mathbf{v} \)
- The solutions \( \lambda \) are called *eigenvalues*.
- The related vectors \( \mathbf{v} \) are *eigenvectors*. 
Qualitative Behavior, Again

• For a dynamical system to be stable:
  – The real parts of all eigenvalues must be negative.
  – All eigenvalues lie in the left half complex plane.

• Terminology:
  – Underdamped = spiral (some complex eigenvalue)
  – Overdamped = nodal (all eigenvalues real)
  – Critically damped = the boundary between.
Node Behavior

FIG. C. Node: $B = \begin{bmatrix} \lambda & 0 \\ 0 & \mu \end{bmatrix}$, $\lambda < \mu < 0$. 
Focus Behavior

FIG. B. Focus: $B = \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix}$, $\lambda < 0$. 
Saddle Behavior

FIG. A. Saddle: $B = \begin{bmatrix} \lambda & 0 \\ 0 & \mu \end{bmatrix}$, $\lambda < 0 < \mu$. 
Spiral Behavior

(stable attractor)

FIG. E. Spiral sink: \( B = \begin{bmatrix} a & -b \\ b & a \end{bmatrix}, \ b > 0 > a. \)
Center Behavior
(undamped oscillator)

\[ B = \begin{bmatrix} 0 & -b \\ b & 0 \end{bmatrix}, \quad b > 0. \]
The Wall Follower

$(x, y)$

$\theta$
The Wall Follower

- Our robot model:

\[
\begin{pmatrix}
\dot{x} \\
\dot{y} \\
\dot{\theta}
\end{pmatrix} = F(x, u) = \begin{pmatrix}
v \cos \theta \\
v \sin \theta \\
o
\end{pmatrix}
\]

\[
u = (v \ \omega)^T \quad y = (y \ \theta)^T \quad \theta \approx 0.
\]

- We set the control law \( u = (v \ \omega)^T = H_i(y) \)

\[
e = y - y_{set} \quad \text{so} \quad \dot{e} = \dot{y} \quad \text{and} \quad \ddot{e} = \ddot{y}
\]
The Wall Follower

- Assume constant forward velocity $v = v_0$
  - approximately parallel to the wall: $\theta \approx 0$.
- Desired distance from wall defines error:
  
  $$
e = y - y_{set}$$

  so

  $$\dot{e} = \dot{y}$$

  and

  $$\ddot{e} = \ddot{y}$$

- We set the control law $u = (v \omega)^T = H_i(y)$
  - We want $e$ to act like a “damped spring”

  $$\ddot{e} + k_1 \dot{e} + k_2 e = 0$$
The Wall Follower

- We want \( \ddot{e} + k_1 \dot{e} + k_2 e = 0 \)
- For small values of \( \theta \)
  \[
  \begin{align*}
  \dot{e} &= \dot{y} = v \sin \theta \approx v \theta \\
  \ddot{e} &= \ddot{y} = v \cos \theta \dot{\theta} \approx v \omega
  \end{align*}
  \]
- Assume \( v = v_0 \) is constant. Solve for \( \omega \)
  \[
  u = \begin{pmatrix} v \\ \omega \end{pmatrix} = \begin{pmatrix} v_0 \\
  -k_1 \theta - \frac{k_2}{v_0} e \end{pmatrix} = H_i(e, \theta)
  \]
- This makes the wall-follower a PD controller.
Tuning the Wall Follower

• The system is $\ddot{e} + k_1 \dot{e} + k_2 e = 0$
• Critically damped is $k_1^2 - 4k_2 = 0$
  \[ k_1 = \sqrt{4k_2} \]
• Slightly underdamped performs better.
  – Set $k_2$ by experience.
  – Set $k_1$ a bit less than $\sqrt{4k_2}$
An Observer for Distance to Wall

- Short sonar returns are reliable.
  - They are likely to be perpendicular reflections.
Experiment with Alternatives

• The wall follower is a PD control law.
• A target seeker should probably be a PI control law, to adapt to motion.

• Try different tuning values for parameters.
  – This is a simple model.
  – Unmodeled effects might be significant.
Ziegler-Nichols Tuning

• Open-loop response to a step increase.
Ziegler-Nichols Parameters

- $K$ is the process gain.
- $T$ is the process time constant.
- $d$ is the deadtime.
Tuning the PID Controller

• We have described it as:

\[ u(t) = -k_P e(t) - k_I \int_0^t e \, dt - k_D \dot{e}(t) \]

• Another standard form is:

\[ u(t) = -P \left[ e(t) + T_I \int_0^t e \, dt + T_D \dot{e}(t) \right] \]

• Ziegler-Nichols says:

\[ P = \frac{1.5 \cdot T}{K \cdot d} \quad T_I = 2.5 \cdot d \quad T_D = 0.4 \cdot d \]
Ziegler-Nichols Closed Loop

1. Disable D and I action (pure P control).
2. Make a step change to the setpoint.
3. Repeat, adjusting controller gain until achieving a stable oscillation.

• This gain is the “ultimate gain” $K_u$.
• The period is the “ultimate period” $P_u$.
Closed-Loop Z-N PID Tuning

- A standard form of PID is:
  \[ u(t) = -P \left[ e(t) + T_I \int_0^t e \, dt + T_D \dot{e}(t) \right] \]

- For a PI controller:
  \[ P = 0.45 \cdot K_u \quad T_I = \frac{P_u}{1.2} \]

- For a PID controller:
  \[ P = 0.6 \cdot K_u \quad T_I = \frac{P_u}{2} \quad T_D = \frac{P_u}{8} \]