

Enhancing Autonomous Mobility Safety via Model-Driven Situation Awareness

**Hanyang University, Korea
Electrical Engineering**

Prof. Changmook Kang

Introduction

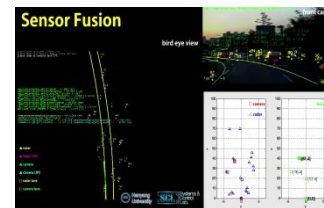
Control

- Proportional Integral Derivative (PID) Control
- Linear Quadratic Gaussian (LQG)
- H_2/H_∞ Control
- Linear Parameter Varying (LPV) Control
- Sliding Mode Control (SMC)
- Linear Matrix Inequality (LMI)
- Model Predictive Control (MPC)
- Backstepping Control
- Passivity-based Control
- Flatness-based Control



Vehicle & Robot Applications

- Modeling
 - Dynamics & Kinematics
 - Denavit-Hartenberg Transformation
- Filtering
 - Extended Kalman Filter (EKF)
 - Unscented KF (UKF)
 - Particle Filter (PF)
- Sensor Fusion
- Path Planning/Tracking
- Simultaneous Localization And Mapping (SLAM)



Introduction

AGV, Self-driving Car, Drone



- **Field Robot/UGV**
- No lane
- No traffic light
- No boundary
- Various unknown object



- **Security Service Drone**
- Surveillance
- Suspect chase
- Inspection
- Anti-drone

Limitations of Fixed-Gain LQR under Nonlinear/Uncertain Dynamics

LQR Cost Function and Linear System Model

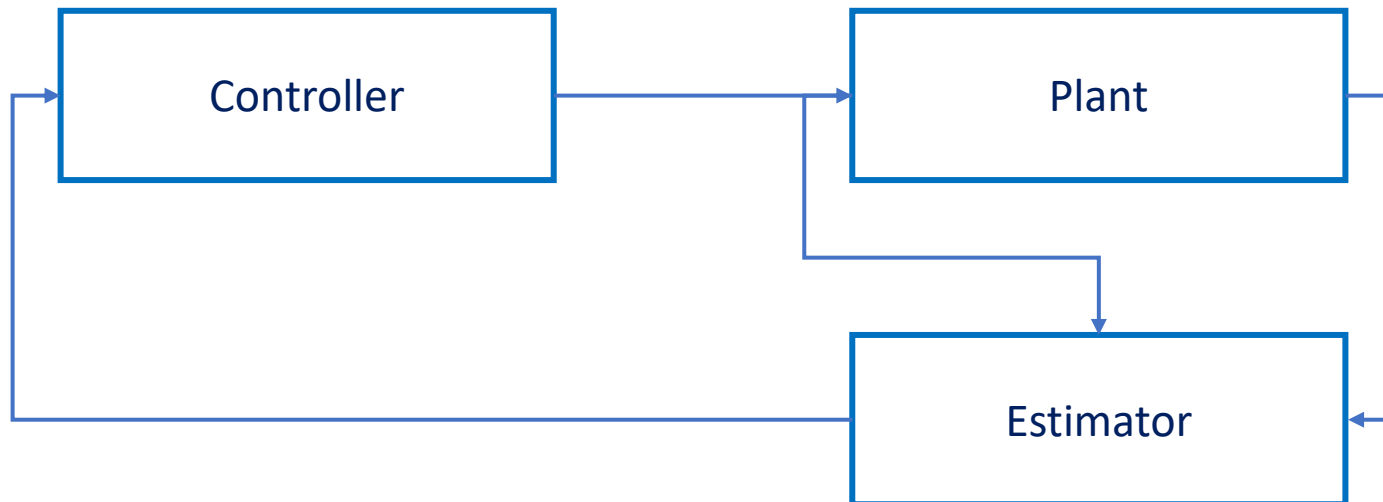
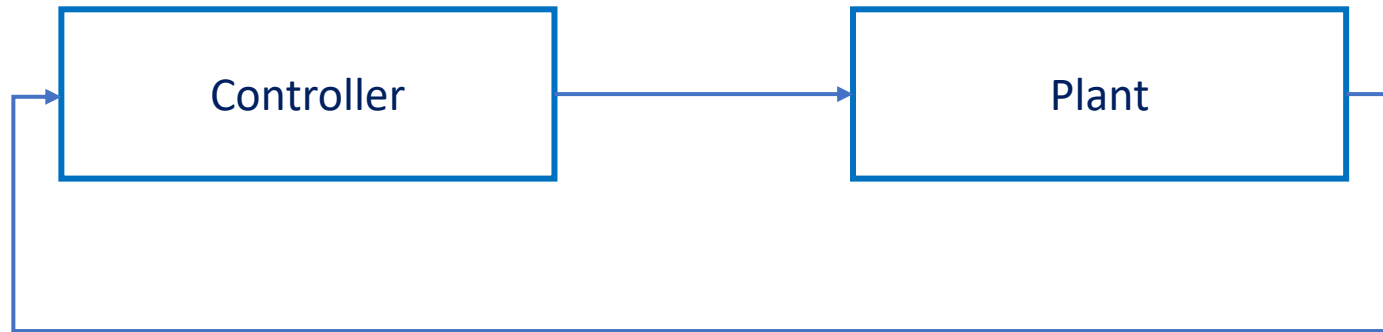
$$J = \sum_{k=0}^{\infty} (x_k^{\top} Q x_k + u_k^{\top} R u_k)$$
$$x_{k+1} = A x_k + B u_k$$

Where:

- $x_k \in \mathbb{R}^n$ — state vector
- $u_k \in \mathbb{R}^m$ — control input
- $Q \succeq 0$ — state weighting matrix (positive semi-definite)
- $R \succ 0$ — control weighting matrix (positive definite)
- A, B — system and input matrices

- **Fixed-gain LQR controllers** can become **overly conservative or even unstable** when subjected to nonlinearities, parameter drift, or piecewise operating mode changes. These issues intensify as the discrepancy between the design-time linear model and the actual system dynamics increases.
- When the cost function J and linear model assumptions break down, the **trade-off between tracking error and control energy** becomes distorted. Particularly in systems with multiple operating modes, a single gain matrix cannot guarantee optimality across the entire operating envelope.
- Overcoming these limitations requires **adaptive control strategies** or multiple model-based approaches that can adjust to changing dynamics.

Linear time invariant system & control



Control

- Proportional Integral Derivative (PID) Control
- Linear Quadratic Gaussian (LQG)
- H_2/H_∞ Control
- Sliding Mode Control (SMC)
- Linear Matrix Inequality (LMI)
- Model Predictive Control (MPC)
- **Linear Parameter Varying (LPV) Control**

$$K_{LPV} = \sum_i \xi_i(\rho) K^{[i]}$$

K_{LPV} : Gain matrix of the Linear Parameter Varying (LPV) controller.

$K^{[i]}$: Gain matrix corresponding to the i -th vertex (local linear model).

$\xi_i(\rho)$: Scheduling weight associated with the i -th vertex, determined by the scheduling parameter ρ .

ρ : Scheduling parameter (e.g., vehicle speed, steering angle, or other measurable varying quantity).

Estimator

- Kalman Filter (KF)
- Extended Kalman Filter (EKF)
- Unscented Kalman Filter (UKF)
- Ensemble Kalman Filter (EnKF)
- Particle Filter (PF)
- Luenberger Observer
- Unknown Input Observer (UIO)
- Sliding Mode Observer (SMO)
- **Interacting Multiple Model (IMM) Estimator**

$$\sum_{i=1}^N \mu_i(k) = 1$$

$\mu_i(k)$ = mode probability of model i at time step k ,

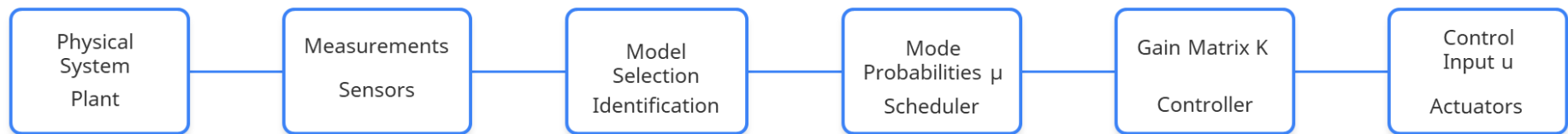
and N = total number of models in the IMM estimator.

Adaptive Control and Multiple Model (MM) Concepts

Adaptive control detects system parameter changes in real-time and automatically adjusts controller gains through online identification and updating. This approach is effective when model uncertainty is high, though convergence and stability analysis can be complex.

The Multiple Model (MM) approach operates several local linear models in parallel to distinguish different operating modes. Each model represents a specific operating region or condition, requiring a mechanism to identify which model best fits the current state.

Adaptive control and multiple model methods form a continuous identification-scheduling-control loop, providing **robust control** performance even in changing environments.



Multiple Model Adaptive Control System Architecture

Online Model Selection

Dynamic Parameter Adjustment

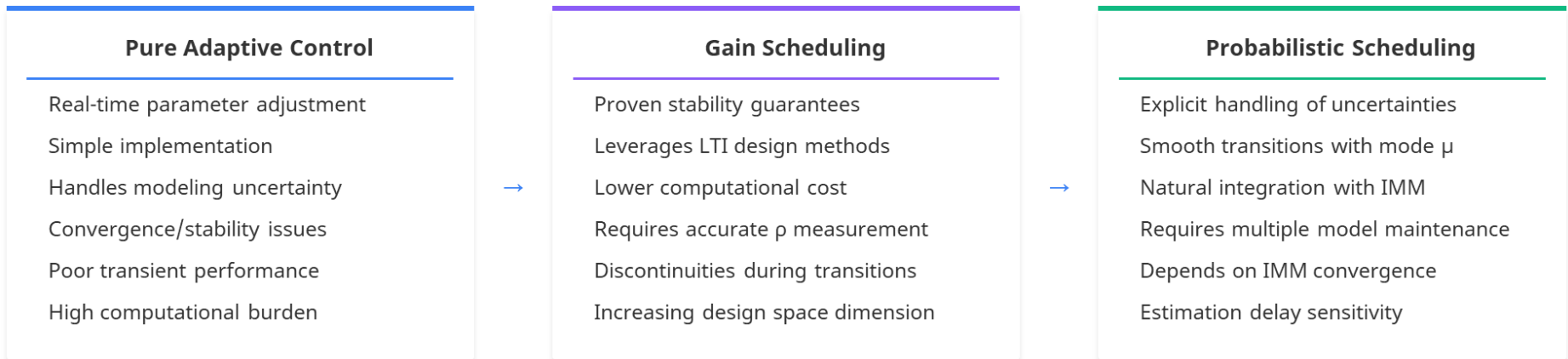
Gain Scheduling

Transition: Adaptive Laws→Gain Scheduling→Probabilistic Scheduling

Pure adaptive control adjusts controller parameters through real-time estimation but suffers from difficulties in convergence/stability analysis and lacks guaranteed transient performance. These limitations motivated the development of gain scheduling approaches.

Gain scheduling switches between pre-designed controllers based on operating conditions but heavily depends on accurate scheduling variable measurements, potentially causing discontinuities during rapid transitions between operating regions.

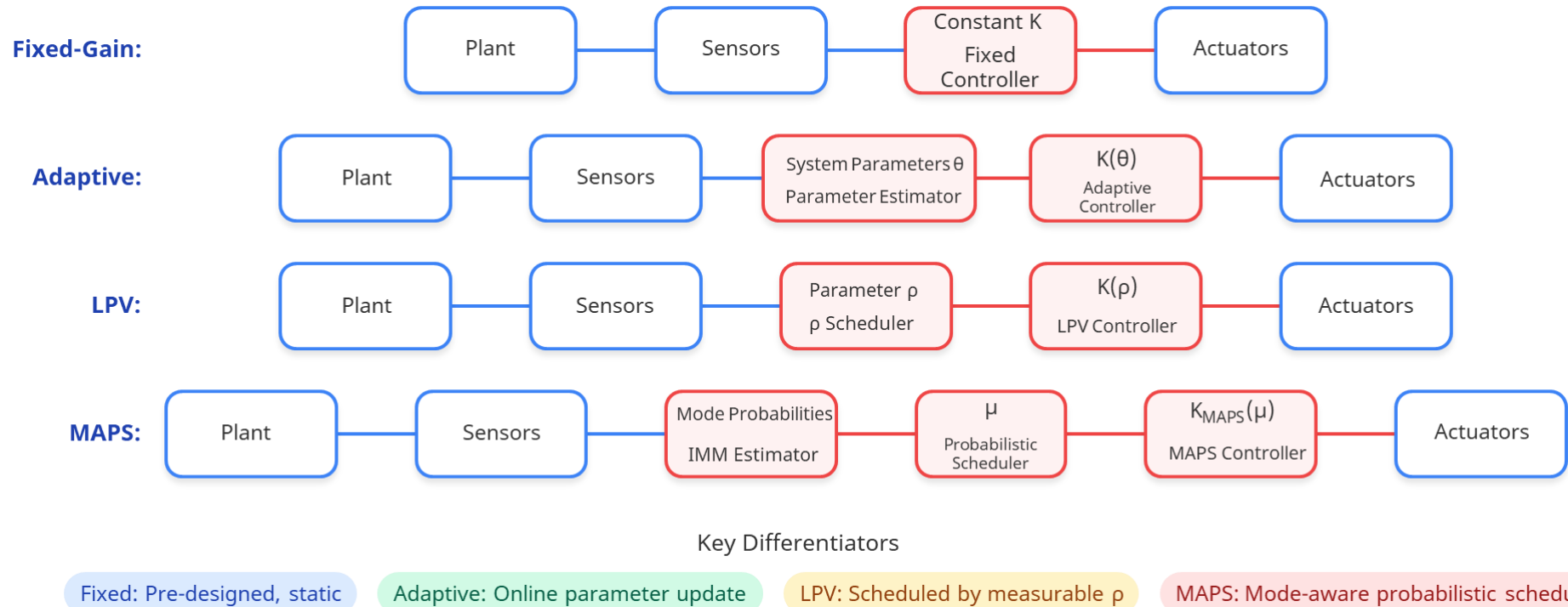
Probabilistic scheduling uses mode probabilities μ as interpolation weights to provide continuous and smooth transitions between controllers while explicitly accounting for mode uncertainties, offering more **robust control** performance.



Fixed-Gain vs Adaptive vs LPV vs MAPS: Comparison Diagram

Control systems evolution from **fixed-gain** to **mode-aware probabilistic scheduling** demonstrates increasing sophistication in handling uncertainty and nonlinearity. This comparative architecture reveals how each approach processes system information and adapts control strategies.

Each approach differs primarily in how the **controller parameters** are determined: fixed a priori, adapted online from data, scheduled via measured parameters, or **probabilistically blended** based on estimated mode likelihoods.



MAPS = Estimation (IMM) + Control (LPV) Coupling

Mode-Aware Probabilistic Scheduling (MAPS) uses IMM-estimated mode probabilities μ directly as LPV controller interpolation weights, creating a tight coupling between state estimation and control adaptation.

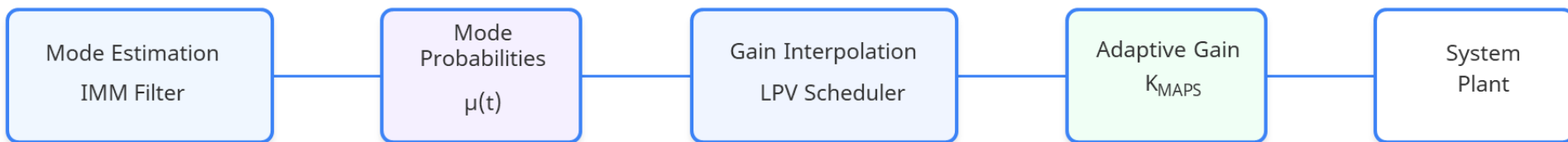
This integration enables the controller to dynamically adjust gains in real-time, effectively responding to operating mode changes without requiring explicit parameter measurement or separate scheduling variables.

The seamless connection between estimation and control makes MAPS robust to mode uncertainty and transitions, providing smooth adaptation with provable stability guarantees.

$$K_{\text{MAPS}} = \sum_i \mu_i K^{[i]}$$

Description

- K_{MAPS} : Final control gain under MAPS framework
- μ_i : Probability of mode i
- $K^{[i]}$: LQR control gain computed for mode i



MAPS Framework: Probabilistic Gain Scheduling Architecture

IMM-based Estimation

Probabilistic Scheduling

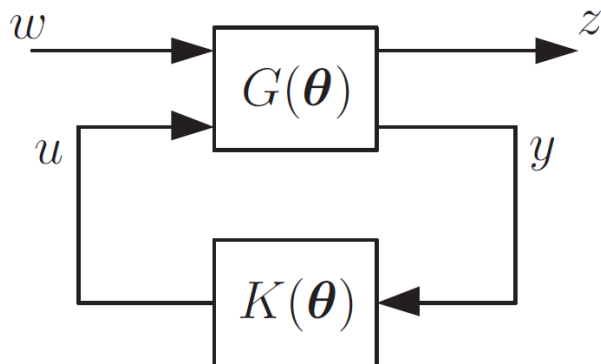
LPV-based Control

Linear Parameter Varying System Overview

- **Linear Parameter Varying (LPV) system ?**

Linear parameter varying (LPV) model is a linear state-space model whose dynamics vary as a function of certain **time-varying parameters**

LPV model can be represented in a state-space form using **coefficients** that are parameter dependent.



$$\dot{x} = A(\theta)x + B_1(\theta)w + B_2(\theta)u$$

$$z = C_1(\theta)x + D_{11}(\theta)w + D_{12}(\theta)u$$

$$y = C_1(\theta)x + D_{21}(\theta)w$$

LPV System Definition

A **Linear Parameter-Varying (LPV) system** features state-space matrices that depend continuously on a measurable time-varying parameter vector. This framework bridges the gap between linear time-invariant and nonlinear systems by capturing parameter-dependent dynamics.

LPV systems are versatile and can represent both linear time-varying (LTV) systems when parameters follow known trajectories, and approximate nonlinear systems when parameters are functions of states (quasi-LPV).

The **scheduling parameter** ρ represents operating conditions, and the system matrices vary continuously with this parameter, enabling smooth control adaptation across different regimes.

Key Equation: Discrete LPV System Model

$$\begin{aligned}x_{k+1} &= A(\rho_k) x_k + B(\rho_k) u_k \\ \rho_k &\in \Omega \subset \mathbb{R}^N\end{aligned}$$

Where:

- $x_k \in \mathbb{R}^n$: State vector ($n \times 1$)
- $u_k \in \mathbb{R}^m$: Control input vector ($m \times 1$)
- $\rho_k \in \mathbb{R}^N$: Scheduling parameter vector
- $A(\rho), B(\rho)$: Continuous matrix functions of the scheduling parameter
- Ω : Compact set defining the bounded parameter region

Assumption Classification

- **Necessary:** ρ measurable
- **Sufficient:** ρ bounded in compact set Ω

Polytopic Approximation and Convex Combination

The **polytopic representation** approximates parameter-varying matrices as a **convex combination of vertex systems**, converting the infinite-dimensional LPV design problem into a **finite set of LTI models**.

For bounded parameters, the vector \mathbf{p} lies within a polytope with 2^N **vertices**, each representing a fixed parameter combination.

This method enables **control design at vertex points** and ensures **stability** for all intermediate parameters through **convex interpolation**.

Key Equations: Polytopic System Representation

$$A(\rho) = \sum_{i=1}^Z \xi_i(\rho) A^{[i]}, \quad \text{where} \quad \sum_{i=1}^Z \xi_i(\rho) = 1, \quad \xi_i(\rho) \geq 0$$

$$\rho \in \text{Co}\{\omega_1, \omega_2, \dots, \omega_Z\}, \quad \text{with} \quad Z = 2^N$$

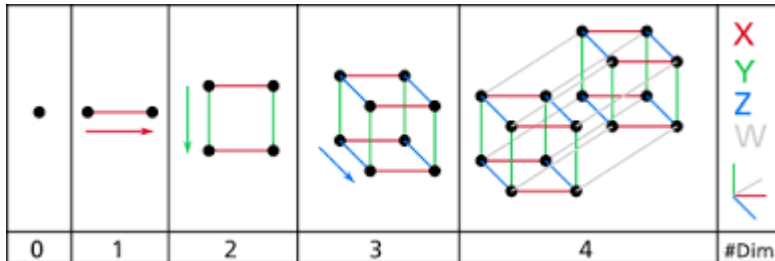
$$\rho = \sum_{i=1}^Z \xi_i \omega_i$$

Where:

- $A^{[i]} \in \mathbb{R}^{n \times n}$: System matrix at vertex i
- $\xi_i(\rho) \in [0, 1]$: Convex weighting function for vertex i
- $\omega_i \in \mathbb{R}^N$: Vertex parameter vector (e.g., $\omega_i = [\rho_1^{\min/\max}, \dots, \rho_N^{\min/\max}]$)
- Z : Number of vertices (corners of the parameter polytope)
- N : Number of varying parameters

Assumption Classification

- **Sufficient:** Convex hull contains all operating points
- **Heuristic:** Gridding resolution choice



LPV Control Law: Vertex Gain Interpolation

LPV control forms a **continuous control surface** by interpolating vertex gains with **scheduling weights**, ensuring smooth transitions and maintaining **stability**.

The **control law** uses the same convex weights for both plant and controller, preserving coherence and avoiding abrupt changes.

The resulting **closed-loop system** retains convexity, enabling **systematic stability analysis** and **structured controller design**.

Key Equations: LPV Control Law and Closed-Loop Dynamics

$$u_k = \sum_{i=1}^r \xi_i(\rho_k) K^{[i]} x_k$$
$$x_{k+1} = \left(\sum_{i=1}^r \xi_i(\rho_k) A_{cl}^{[i]} \right) x_k$$
$$A_{cl}^{[i]} = A^{[i]} + B^{[i]} K^{[i]}$$

Where:

- $\xi_i(\rho_k)$: Convex interpolation weights satisfying $\xi_i \geq 0$, $\sum_{i=1}^r \xi_i = 1$
- $K^{[i]} \in \mathbb{R}^{m \times n}$: Control gain matrix for vertex i
- $A_{cl}^{[i]} \in \mathbb{R}^{n \times n}$: Closed-loop system matrix at vertex i
- $A^{[i]}, B^{[i]}$: System and input matrices at vertex i
- $\rho_k \in \mathbb{R}^p$: Scheduling parameter vector at time k

Assumption Classification

- **Sufficient**: Vertex-wise stability
- **Necessary**: Continuous interpolation
- **Heuristic**: Smooth parameter variation

LPV Control Law: Stability with Common P

For a **polytopic LPV system**, if a **common positive definite matrix P** satisfies the **Lyapunov inequality** at all vertices, the system is **quadratically stable** across the convex hull.

This allows stability verification via a **finite set of LMIs**, with a single **quadratic Lyapunov function** valid for all trajectories.

However, this method is **conservative**, since one P must hold for all vertices, possibly missing feasible parameter-dependent solutions.

Key Stability Conditions

$$(A_{cl}^{[i]})^\top P A_{cl}^{[i]} - P \prec 0, \quad \forall i \in \{1, \dots, Z\}$$

LMI form:

$$A_{cl}^{[i]\top} P A_{cl}^{[i]} - P + \varepsilon I \prec 0, \quad i = 1, \dots, Z$$

Where:

- $A_{cl}^{[i]} = A^{[i]} + B^{[i]} K^{[i]}$: Closed-loop system matrix at vertex i
- $P = P^\top \succ 0$: Common Lyapunov matrix (decision variable)
- $\varepsilon > 0$: Small positive constant ensuring strict feasibility
- Z : Number of vertices (typically $Z = 2^N$ for N parameters)
- $\prec 0$: Negative definite matrix inequality

Assumption Classification

- **Sufficient**: Common P existence
- **Not necessary**: Parameter-dependent $P(\rho)$ may exist when common P does not

Parameter-Dependent Lyapunov Function $V(x, \rho)$

A **parameter-dependent Lyapunov function** $V(x, \rho) = x^T P(\rho)x$ reduces **conservatism**, enabling stability where no common P exists, but increases **LMI complexity** due to extra variables.

When ρ changes over time, bounds on $\dot{\rho}$ are required, creating a trade-off between **accuracy** and **computational cost**.

Common parametrizations include **affine**, **polynomial**, and **grid-based** forms, each balancing complexity and flexibility differently.

Key Equation: Stability Condition with Parameter-Dependent Lyapunov Function

$$A(\rho)^T P(\rho) A(\rho) - P(\rho) + \sum_{i=1}^N \dot{\rho}_i \frac{\partial P(\rho)}{\partial \rho_i} < 0$$

$$P(\rho) = P_0 + \sum_{i=1}^N \rho_i P_i \quad (\text{Affine parametrization})$$

Where:

- $P(\rho) \in \mathbb{R}^{n \times n}$: Parameter-dependent Lyapunov matrix
- P_0, P_i : Constant matrices (decision variables in the LMI)
- $\dot{\rho}_i$: Rate of change of parameter i
- $\frac{\partial P(\rho)}{\partial \rho_i}$: Partial derivative of $P(\rho)$ w.r.t. ρ_i
(equals P_i for affine case)
- ν_i : Bound on parameter rate such that $|\dot{\rho}_i| \leq \nu_i$

Assumption Classification

- **Sufficient:** Rate-bounded parameters
- **Empirical:** Selected parametrization structure

Gain Interpolation Smoothness: Lipschitz Constraint

In **LPV control systems**, rapid gain changes can cause **transients** or **actuator saturation**, especially during fast parameter shifts.

The **Lipschitz continuity constraint** ensures gains vary smoothly, keeping changes proportional to parameter variation — essential for actuators with limited speed.

If **sensor delays** occur ($T_{\text{delay}} > 0$), phase lag may reduce performance or stability.

To mitigate this, use **parameter prediction** or **smooth interpolation** (e.g., softmax) instead of abrupt switching.

Key Equation: Lipschitz Continuity Constraint

$$\|K(\rho_1) - K(\rho_2)\| \leq L \|\rho_1 - \rho_2\|$$

(With delay) $u(t) = K(\rho(t - T_{\text{delay}})) x(t)$

Where:

- $K(\rho) \in \mathbb{R}^{m \times n}$: Parameter-dependent controller gain matrix
- $\rho_1, \rho_2 \in \mathbb{R}^N$: Scheduling parameter vectors
- $L > 0$: **Lipschitz constant**, representing the maximum allowed rate of gain variation
- $T_{\text{delay}} \geq 0$: Scheduling parameter measurement or processing delay
- $\|\cdot\|$: Matrix or vector norm (typically Frobenius norm)

Practical Assumptions: Slow ρ Variation, Bounded Uncertainty

In **LPV control**, stability and performance depend on several key assumptions that define the limits of theoretical guarantees.

In practice, these ideal assumptions may not always hold, so engineers must assess their impact on system behavior. The table below summarizes main assumptions, their types, and mitigation strategies.

| Assumption | Classification | Violation Risk | Mitigation Strategies |
|---|----------------|---|--|
| Slow Scheduling Parameter Variation Parameter rates are bounded: $ \dot{\rho} \leq \nu$ | heuristic | High: Destabilization Rapid transients can invalidate frozen-time analysis and cause instability during transitions | <ul style="list-style-type: none">• Use parameter-dependent Lyapunov functions $P(\rho)$• Rate-bounded LMI constraints incorporating $\dot{\rho}$• Add artificial damping during rapid transitions |
| Bounded Modeling Error Model mismatch is limited: $\ \Delta A\ < \varepsilon$ | empirical | Moderate: Performance degradation Large mismatches lead to suboptimal gain scheduling and reduced robustness margins | <ul style="list-style-type: none">• Robust LMI formulation with S-procedure• Uncertainty polytope modeling• Mixed H_2/H_∞ design criteria |
| Vertex Stabilizability $\exists K[i]$ stabilizing each vertex | necessary | Critical: Fundamental instability If vertices cannot be stabilized individually, convex combination cannot be stabilized | <ul style="list-style-type: none">• Redesign operating grid boundaries• Add actuators or modify control architecture• Verify controllability at all vertices |
| Common P Existence Single $P > 0$ satisfies all vertex LMIs | sufficient | Moderate: Over-conservatism Single Lyapunov function for all vertices limits achievable performance | <ul style="list-style-type: none">• Switch to parameter-dependent $P(\rho)$• Use mode-dependent Lyapunov functions P_i• Implement path-dependent stability analysis |

Example: LPV in vehicle localization

- **Autonomous driving**
Where to go?



Lane recognition

- **GPS** and IMU are generally used for localization
- **Kinematic vehicle model** is widely used for vehicle localization with GPS
- **Course angle** term makes the model nonlinear

What's around me?



Object detection/tracking

Where am I?



Localization

$$\begin{aligned}\dot{X} &= V \cos(\beta + \psi) \\ \dot{Y} &= V \sin(\beta + \psi) \\ \psi &= \frac{V}{l_f + l_f} \cos \beta \tan \delta\end{aligned}$$

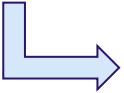
Example: LPV in vehicle localization

Vehicle motion in global coordinate

$$\begin{aligned}\dot{X} &= V \cos(\beta + \psi) \\ \dot{Y} &= V \sin(\beta + \psi) \\ \psi &= \frac{V}{l_f + l_r} \cos \beta \tan \delta\end{aligned}$$

Velocity kinematics-based model

$$a_y = -V_y + V_x \psi$$

 $\dot{\beta} = \psi - \frac{a_y}{V_x}$

Vehicular yaw dynamics

$$I_z \ddot{\psi} = l_f F_{yf} - l_r F_{yr}$$

• System model for localization

$$\mathbf{x} = [X \quad Y \quad V \quad \beta \quad \psi \quad \dot{\psi}]$$

$$u = \delta \quad \phi = \left[a \quad \frac{a_y}{V_x} \right]$$

$$\mathbf{x}(k+1) = f(\mathbf{x}(k), u(k), \phi(k))$$

$$= \begin{bmatrix} X(k) + TV \cos(\beta(k) + \psi(k)) \\ Y(k) + TV \sin(\beta(k) + \psi(k)) \\ V(k) + Ta(k) \\ \beta(k) + T\psi(k) - T \frac{a_y(k)}{V_x(k)} \\ \psi(k) + T\dot{\psi}(k) \\ \psi(k) - Ta_{42}\beta(k) + Ta_{44}\psi(k) + Tb_{41}\delta(k) \end{bmatrix}$$

Example: LPV in vehicle localization

- Extended Kalman filter

Predict

$$\begin{aligned}\hat{\mathbf{x}}_{k|k-1} &= f(\hat{\mathbf{x}}_{k-1|k-1}, u_{k-1}, \phi_{k-1}) \\ P_{k|k-1} &= A_e P_{k-1|k-1} A_e^T + B_e Q_{k-1} B_e^T \\ \text{where} \\ A_e &= \frac{\partial}{\partial \mathbf{x}} f|_{\mathbf{x}=\hat{\mathbf{x}}_k}, \quad B_e = \frac{\partial}{\partial u} f|_{\mathbf{x}=\hat{\mathbf{x}}_k}\end{aligned}$$



Update

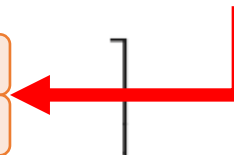
$$\begin{aligned}\tilde{y}_k &= y_k - \hat{y}_k \\ S_k &= C_k P_{k|k-1} C_k^T + R_k \\ K_k &= P_{k|k-1} C_k^T S_k^{-1} \\ \hat{\mathbf{x}}_{k|k} &= \hat{\mathbf{x}}_{k|k-1} + K_k \tilde{y}_k \\ P_{k|k} &= (I - K_k C_k) P_{k|k-1}\end{aligned}$$

To calculate covariance matrix, **linearization** of the state equation at each time step is required

$$\mathbf{x}(k+1) = f(\mathbf{x}(k), u(k), \phi(k))$$

$$\begin{bmatrix} X(k) + TV \cos(\beta(k) + \psi(k)) \\ Y(k) + TV \sin(\beta(k) + \psi(k)) \end{bmatrix}$$

Nonlinear bounded terms



Example: LPV in vehicle localization

Define the varying parameters

$$\theta_1(\gamma) = \cos(\beta + \psi) \in [\bar{\theta}_1 \quad \underline{\theta}_1]$$

$$\theta_2(\gamma) = \sin(\beta + \psi) \in [\bar{\theta}_2 \quad \underline{\theta}_2]$$

Due to the trigonometric function, the varying parameters are bounded

A dynamics becomes the following LPV system

$$\mathbf{x}(k+1) = \Phi(\theta) \mathbf{x}(k) + \Gamma u(k) + \Gamma_\varphi \varphi(k)$$

$$\mathbf{y}(k) = C \mathbf{x}(k)$$

where

$$\Phi(\theta) = \begin{bmatrix} 1 & 0 & T\theta_1(\gamma) & 0 & 0 & 0 \\ 0 & 1 & T\theta_2(\gamma) & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & T \\ 0 & 0 & 0 & 0 & 1 & T \\ 0 & 0 & 0 & -Ta_{42} & 0 & 1 + Ta_{44} \end{bmatrix},$$

$$\Gamma = [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad Tb_{41}]^T,$$

$$\Gamma_\varphi = \begin{bmatrix} 0 & 0 & T & 0 & 0 & 0 \\ 0 & 0 & 0 & -T & 0 & 0 \end{bmatrix}^T,$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Example: LPV in vehicle localization

The given time-varying parameter vector, $\theta \in \mathbb{R}^2$ may be represented by the following polytopic form

$$\theta(\gamma) = V\xi(\theta)$$

where $V = [v_1 \ v_2 \ v_3 \ v_4] \in \mathbb{R}^{2 \times 4}$ is the vertex

$$V = [v_1 \ v_2 \ v_3 \ v_4] = \begin{bmatrix} \underline{\theta}_1 & 0 & \overline{\theta}_1 & 0 \\ 0 & \underline{\theta}_2 & 0 & \overline{\theta}_2 \end{bmatrix}.$$

$$\xi(\theta) = [\xi_1(\theta) \ \xi_2(\theta) \ \xi_3(\theta) \ \xi_4(\theta)]^T \in \mathbb{R}^4, \ \xi_i \geq 0, \ \sum_{i=1}^4 \xi_i(\theta) = 1,$$

is convex interpolation parameter vector

Proposition

The convex interpolation parameter vector $\xi(\theta)$ can be uniquely determined by the given time-varying parameter vector θ , if vertex V is invertible.

Example: LPV in vehicle localization

The polytopic decomposition can be expressed such that:

$$\Phi^{(1)} = \Phi(0) + \underline{\theta}_1 \hat{\Phi}^{(1)}$$

$$\Phi^{(2)} = \Phi(0) + \underline{\theta}_2 \hat{\Phi}^{(2)}$$

$$\Phi^{(3)} = \Phi(0) + \bar{\theta}_1 \hat{\Phi}^{(1)}$$

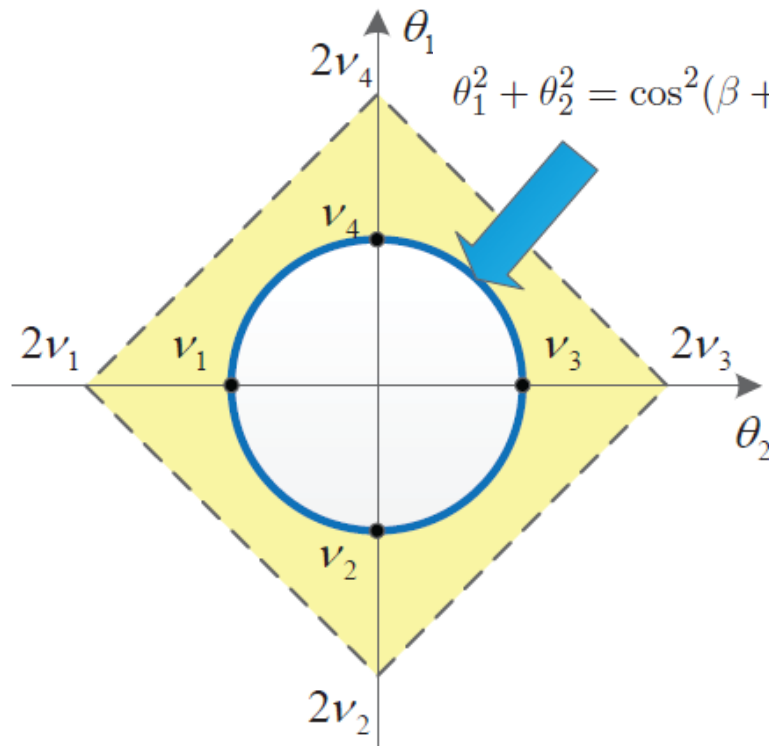
$$\Phi^{(4)} = \Phi(0) + \bar{\theta}_2 \hat{\Phi}^{(2)}$$

$$\Phi(0) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & T \\ 0 & 0 & 0 & 0 & 1 & T \\ 0 & 0 & 0 & -Ta_{42} & 0 & 1 + Ta_{44} \end{bmatrix}, \quad \hat{\Phi}^{(1)} = \begin{bmatrix} 0 & 0 & T\theta_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad \hat{\Phi}^{(2)} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & T\theta_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

parameter-dependent $\Phi(\theta)$

$$\Phi(\theta) = \sum_{i=1}^4 \xi_i(\theta) \Phi^{(i)}$$

Example: LPV in vehicle localization



$$\hat{\mathbf{V}} = \begin{bmatrix} 2\mathbf{V} \\ M^{2 \times 4} \end{bmatrix} = \begin{bmatrix} 2\theta_1 & 0 & 2\bar{\theta}_1 & 0 \\ 0 & 2\theta_2 & 0 & 2\bar{\theta}_2 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix},$$

$$\hat{\Theta} = \begin{bmatrix} \Theta \\ s^{2 \times 1} \end{bmatrix} = [\theta_1 \quad \theta_2 \quad 0.5 \quad 0.5]^T$$

$\xi(\theta)$ is uniquely determined

Definition

For any $\xi_i \geq 0$, $\sum_{i=1}^4 \xi_i(\theta) = 1$, we can define σ_1 and σ_2 as follows:

$$\xi_1(\theta) + \xi_3(\theta) = \sigma_1$$

$$\xi_2(\theta) + \xi_4(\theta) = \sigma_2, \text{ and } \sigma_1 + \sigma_2 = 1.$$

Example: LPV in vehicle localization

Theorem

The discrete-time LPV system is observable if two symmetric matrices $Y, Z > 0$ and $\Pi^{(i)}$ exist for a given $\eta > 0$, such that:

$$\begin{bmatrix} Y & (\Phi^{(i)})^T Y + C_1^T \Pi^{(i)} & C_1^T \\ * & Y & 0 \\ * & * & I \end{bmatrix} > 0$$
$$\begin{bmatrix} Z & D^T \Pi^{(i)} \\ * & Y \end{bmatrix} > 0,$$
$$\text{tr}(Z) < \eta^2$$

by the filter gain $L(\theta) = \sum_{i=1}^4 \xi_i(\theta) L^{(i)}$. Moreover, the filter gain $L(\theta)$ was also guaranteed for the performance of H_2 , $\|T_{zew}\|_2 < \gamma$

[1] CM Kang et al. "Discrete-Time LPV Observer With Nonlinear Bounded Varying Parameter and Its Application to the Vehicle State Observer" IEEE Transactions on Industrial Electronics 65 (11), 8768-8777

[2] CM Kang et al. "Linear parameter varying design for lateral control using kinematics of vehicle motion" 2018 Annual American control conference (ACC), 3239-3244

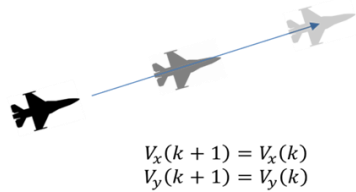
[3] CM Kang et al. "Linear parameter varying observer for lane estimation using cylinder domain in vehicles", IEEE Transactions on Intelligent Transportation Systems 22 (11), 7030-7039

[4] J Baek, C Kang, W Kim, "Practical approach for developing lateral motion control of autonomous lane change system", Applied Sciences 10 (9), 3143

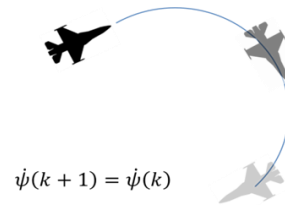
Interacting Multiple Model Overview

IMM Applications Overview

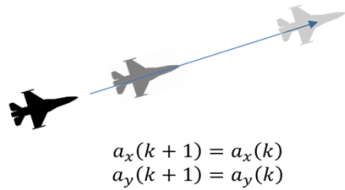
Constant Velocity (CV) model



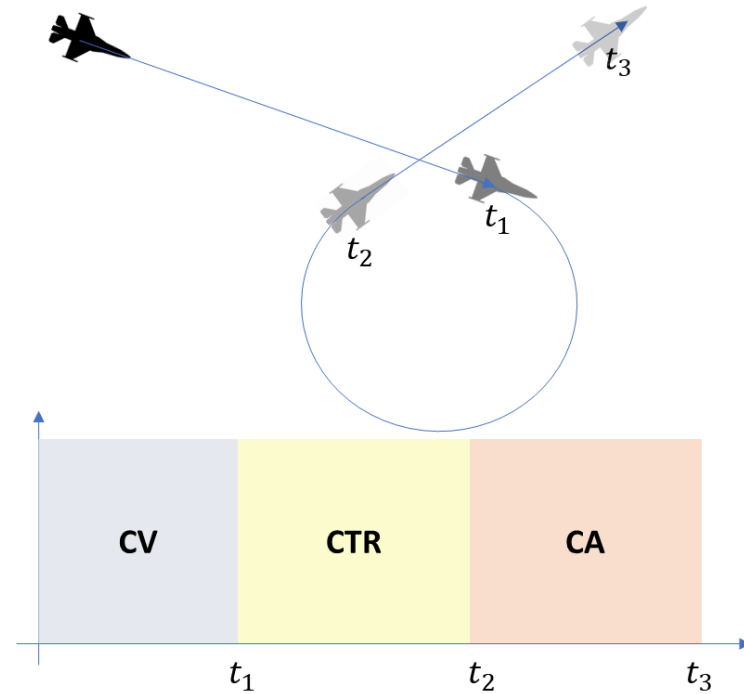
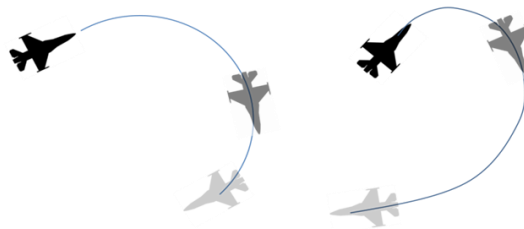
Constant Turn Rate (CTR) model



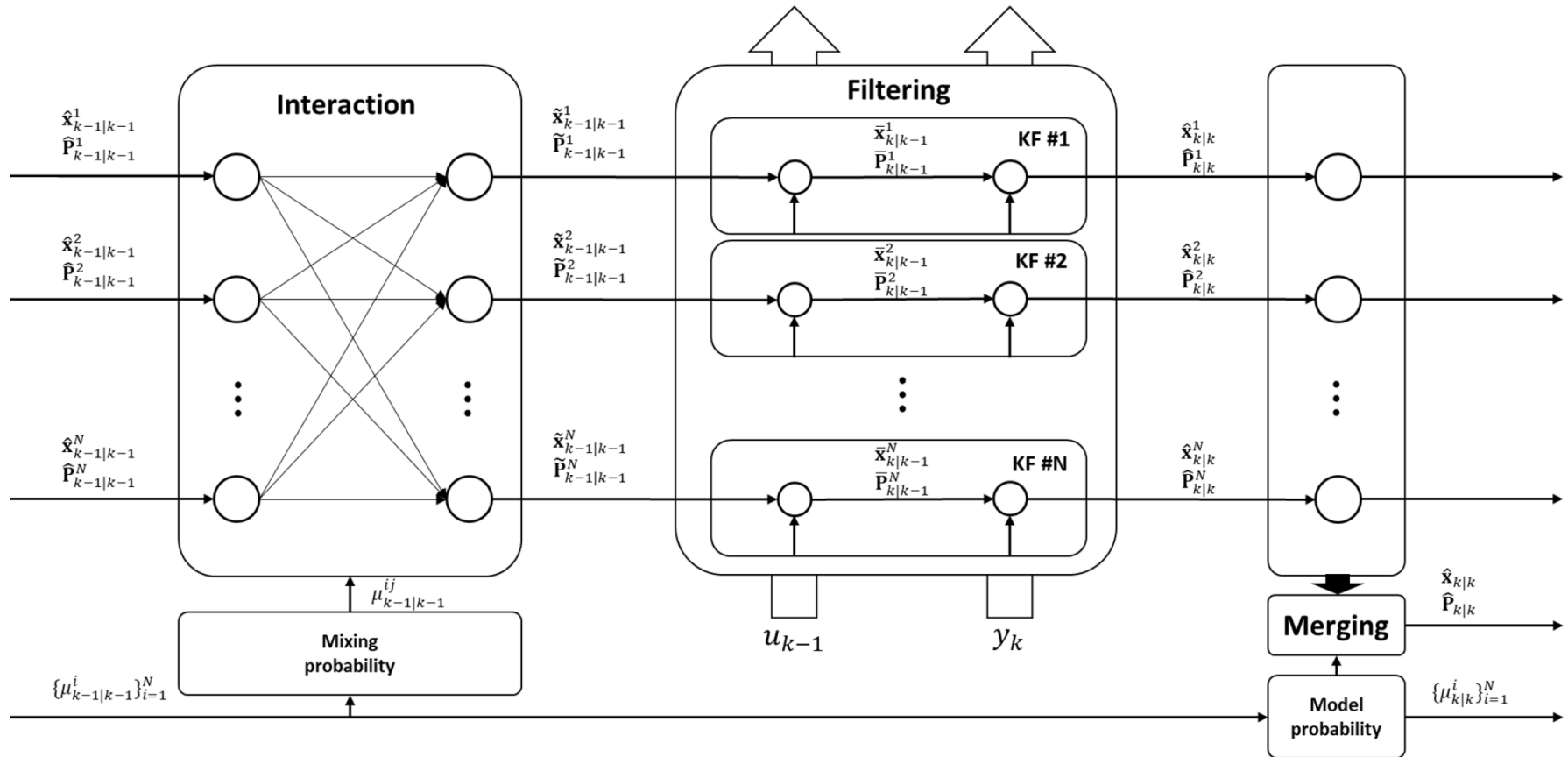
Constant Acceleration (CA) model



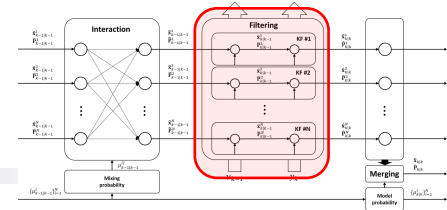
Constant Turn Rate and Acceleration (CTRA) model



IMM Applications Overview



Kalman Filter Prediction Equations Review



In linear Gaussian systems, the **Kalman Filter prediction** step propagates both state and covariance forward in time. This step uses the system model to estimate the current state prior to incorporating new measurements.

The prediction equations represent the optimal projection of state and uncertainty when subjected to known dynamics and control inputs. Under the assumption of Gaussian noise, these equations provide **minimum-variance state estimates**.

The prediction step is followed by the update step, which incorporates new measurements to refine the state estimate. Together, these steps form the recursive estimation cycle of the Kalman Filter.

Kalman Filter Prediction Equations

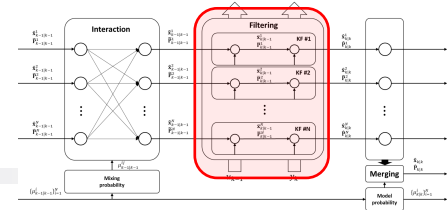
$$\hat{x}_{k|k-1} = A\hat{x}_{k-1|k-1} + Bu_{k-1}$$

$$P_{k|k-1} = AP_{k-1|k-1}A^\top + Q$$

Where:

- $\hat{x}_{k|k-1}$: Predicted state estimate at time k given observations up to $k - 1$
- $\hat{x}_{k-1|k-1}$: Updated state estimate at time $k - 1$
- $P_{k|k-1}$: Predicted error covariance matrix
- $P_{k-1|k-1}$: Updated error covariance matrix from previous step
- A : State transition matrix (system dynamics)
- B : Control input matrix
- u_{k-1} : Control input at time $k - 1$
- Q : Process noise covariance matrix

Kalman Filter Update and Gain



The **Kalman filter update step** incorporates new measurements y_k to transform the prior (predicted) state estimate into a more accurate posterior estimate. This recursive Bayesian process optimally balances model predictions with measurement information.

The **Kalman gain K_k** serves as the blending factor that determines how much to trust the measurement versus the prediction. It is computed using the error covariance and measurement model, minimizing the posterior error covariance.

The measurement noise covariance matrix R significantly impacts filter behavior - larger R values lead to more robust but less sensitive responses, while smaller R values increase sensitivity but may amplify noise effects.

Kalman Filter Update Equations

$$K_k = P_{k|k-1} C^T (C P_{k|k-1} C^T + R)^{-1}$$

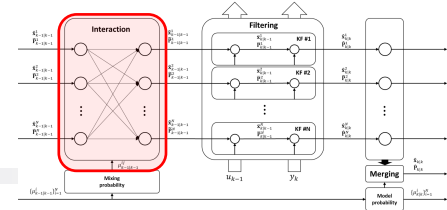
$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k (y_k - C \hat{x}_{k|k-1})$$

$$P_{k|k} = (I - K_k C) P_{k|k-1}$$

Where:

- K_k : Kalman gain matrix
- $P_{k|k-1}$: Prior (predicted) error covariance
- $P_{k|k}$: Posterior (updated) error covariance
- C : Measurement matrix (relating states to measurements)
- R : Measurement noise covariance (tuning parameter)
- $y_k - C \hat{x}_{k|k-1}$: Innovation or measurement residual

Mode Transition Matrix and Markov Modes



Mode transition probabilities describe how a system switches between operating modes. The probability is defined as:

$$p_{ij} = P(\text{mode } j \mid \text{mode } i)$$

This represents the likelihood of transitioning from **mode i to mode j** in the next time step.

These probabilities form the transition matrix, which governs how mode probabilities evolve over time, creating a Markov chain structure. This matrix is essential for the IMM estimator, as it links sequential mode behaviors.

The connection between state-space estimation and mode probability propagation occurs through this transition matrix, which predicts future mode probabilities before measurement updates are applied.

Mode Transition Matrix Definition

$$\Pi = [p_{ij}] = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1r} \\ p_{21} & p_{22} & \cdots & p_{2r} \\ \vdots & \vdots & \ddots & \vdots \\ p_{r1} & p_{r2} & \cdots & p_{rr} \end{bmatrix}$$

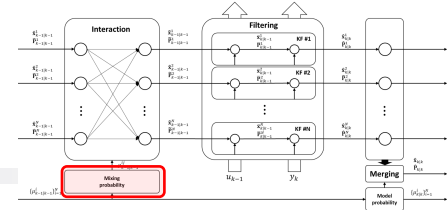
$$\sum_{j=1}^r p_{ij} = 1, \quad \forall i$$

$$\mu_{k|k-1}^{(i)} = \sum_{j=1}^r p_{ji} \mu_{k-1}^{(j)}$$

Where:

- p_{ij} : Transition probability from mode i to mode j
- Π : Mode transition matrix ($r \times r$)
- r : Total number of modes in the system
- $\mu_k^{(i)}$: Probability of mode i at time k
- $\mu_{k|k-1}^{(i)}$: Predicted probability of mode i before measurement update

IMM Mixing Step: Initialization Mixing



In the IMM algorithm, each mode filter's initial state and covariance must be reinitialized by **mixing prior estimates based on mode transition probabilities**. This crucial mixing step captures the influence of all previous mode-specific estimates, weighted according to their likelihood of transitioning to the current mode.

The mixing process creates a combined initial estimate for each filter, serving as the starting point for the next time step's individual Kalman filter updates. This interaction between filters is what distinguishes IMM from simpler multiple-model approaches.

The mixing weights are calculated via Bayes rule, incorporating both the mode transition matrix and the prior mode probabilities to determine the influence of each previous estimate.

Key Equations: IMM Mixing Formulation

$$\bar{x}_{k-1|k-1}^{(j)} = \sum_{i=1}^r \mu_{k-1}^{(j|i)} \hat{x}_{k-1|k-1}^{(i)}$$

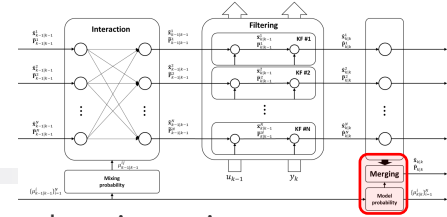
$$\bar{P}_{k-1|k-1}^{(j)} = \sum_{i=1}^r \mu_{k-1}^{(j|i)} \left[P_{k-1|k-1}^{(i)} + \left(\hat{x}_{k-1|k-1}^{(i)} - \bar{x}_{k-1|k-1}^{(j)} \right) \left(\hat{x}_{k-1|k-1}^{(i)} - \bar{x}_{k-1|k-1}^{(j)} \right)^{\top} \right]$$

$$\mu_{k-1}^{(j|i)} = \frac{p_{ij} \mu_{k-1}^{(i)}}{\sum_{m=1}^r p_{mj} \mu_{k-1}^{(m)}}$$

Where:

- $\bar{x}_{k-1|k-1}^{(j)}$: Mixed initial state for mode j filter
- $\bar{P}_{k-1|k-1}^{(j)}$: Mixed initial covariance for mode j filter
- $\hat{x}_{k-1|k-1}^{(i)}$: Previous state estimate from mode i filter
- $P_{k-1|k-1}^{(i)}$: Previous covariance estimate from mode i filter
- $\mu_{k-1}^{(j|i)}$: Mixing probability (weight) from mode i to mode j
- p_{ij} : Transition probability from mode i to mode j (element of matrix Π)
- $\mu_{k-1}^{(i)}$: Previous mode probability for mode i
- $\mu_{k|k-1}^{(j)}$: Predicted mode probability for mode j
- r : Number of models in the IMM filter bank

IMM Posterior Mode Probability Update



The IMM method updates mode probabilities by normalizing the likelihood of each mode based on incoming measurements. This step combines measurement evidence with prior mode probability and transition information.

For each mode j , the posterior probability $\mu_k(j)$ represents our confidence that mode j is active at time k , given all measurements up to time k . This is calculated by normalizing the unnormalized likelihood $c_k(j)$ across all modes.

The measurement likelihood $p(y_k | \text{mode } j)$ acts as a weighting factor that favors modes whose predictions match the actual measurements, creating an adaptive mechanism that automatically shifts probability toward the most accurate model.

Posterior Update Equations

$$\mu_k^{(j)} = \frac{c_k^{(j)}}{\sum_l c_k^{(l)}}$$

$$c_k^{(j)} = p(y_k | \text{mode } j) \sum_i p_{ij} \mu_{k-1}^{(i)}$$

$$p(y_k | \text{mode } j) = \mathcal{N}(y_k - \hat{y}_{k|k-1}^{(j)}; 0, S_k^{(j)})$$

Where:

- $\mu_k^{(j)}$: Posterior probability of mode j at time k
- $c_k^{(j)}$: Unnormalized likelihood (mode j consistency with measurements)
- $p(y_k | \text{mode } j)$: Measurement likelihood given mode j
- p_{ij} : Transition probability from mode i to mode j
- $\mu_{k-1}^{(i)}$: Prior probability of mode i from previous time step
- $\hat{y}_{k|k-1}^{(j)}$: Predicted measurement from filter j
- $S_k^{(j)}$: Innovation (residual) covariance for filter j

Kalman Filter Update and Gain

➤ Interaction

$$\mu_{k|k-1}^j = \sum_i \pi_{ij} \mu_{k-1|k-1}^i$$

$$\mu_{k-1|k-1}^{i|j} = \frac{\pi_{ij} \mu_{k-1|k-1}^i}{\mu_{k|k-1}^j} = \frac{\pi_{ij} \mu_{k-1|k-1}^i}{\sum_i \pi_{ij} \mu_{k-1|k-1}^i}$$

$$\tilde{x}_{k-1|k-1}^j = \sum_i \mu_{k-1|k-1}^{i|j} \hat{x}_{k-1|k-1}^i$$

$$\tilde{P}_{k-1|k-1}^j = \sum_i \mu_{k-1|k-1}^{i|j} \left[\hat{P}_{k-1|k-1}^i + (\tilde{x}_{k-1|k-1}^j - \hat{x}_{k-1|k-1}^i)(\tilde{x}_{k-1|k-1}^j - \hat{x}_{k-1|k-1}^i)^T \right]$$



➤ Filtering

$$\hat{x}_{k|k-1}^j = F_j \tilde{x}_{k-1|k-1}^j$$

$$\hat{x}_{k|k}^j = \hat{x}_{k|k-1}^j + K_j \tilde{x}_{k-1|k-1}^j$$

$$\hat{y}_k^j = y_k^j - H_j \hat{x}_{k|k-1}^j$$



➤ Updating

$$\Lambda_k^j = \frac{\exp \left[-(1/2) (z_k^j)^T (S_k^j)^{-1} z_k^j \right]}{|2\pi S_k^j|^{1/2}}$$

$$S_k^j = H_j P_j H_j^T + R$$

$$\mu_{k|k}^j = \frac{\mu_{k|k-1}^j \Lambda_k^j}{\sum_i \mu_{k|k-1}^i \Lambda_k^i}$$



➤ Combination

$$\hat{x}_{k|k} = \sum_j \mu_{k|k}^j \hat{x}_{k|k}^j$$

$$\hat{P}_{k|k} = \sum_j \mu_{k|k}^j \left[\hat{P}_{k|k}^j + (\hat{x}_{k|k} - \hat{x}_{k|k}^j)(\hat{x}_{k|k} - \hat{x}_{k|k}^j)^T \right]$$

IMM Bounded Estimation Error Assumptions

The IMM filter is based on the assumption that estimation error covariance remains within certain bounds when model mismatches are finite and measurement noise is bounded. The degree of mismatch between the actual system and model directly impacts the performance and stability of IMM.

The **bounded error assumption** forms the foundation for theoretical analysis, but may be violated in practical implementations due to various uncertainty factors. The table below summarizes **important assumptions in IMM** filtering, their implications when violated, and possible mitigation strategies.

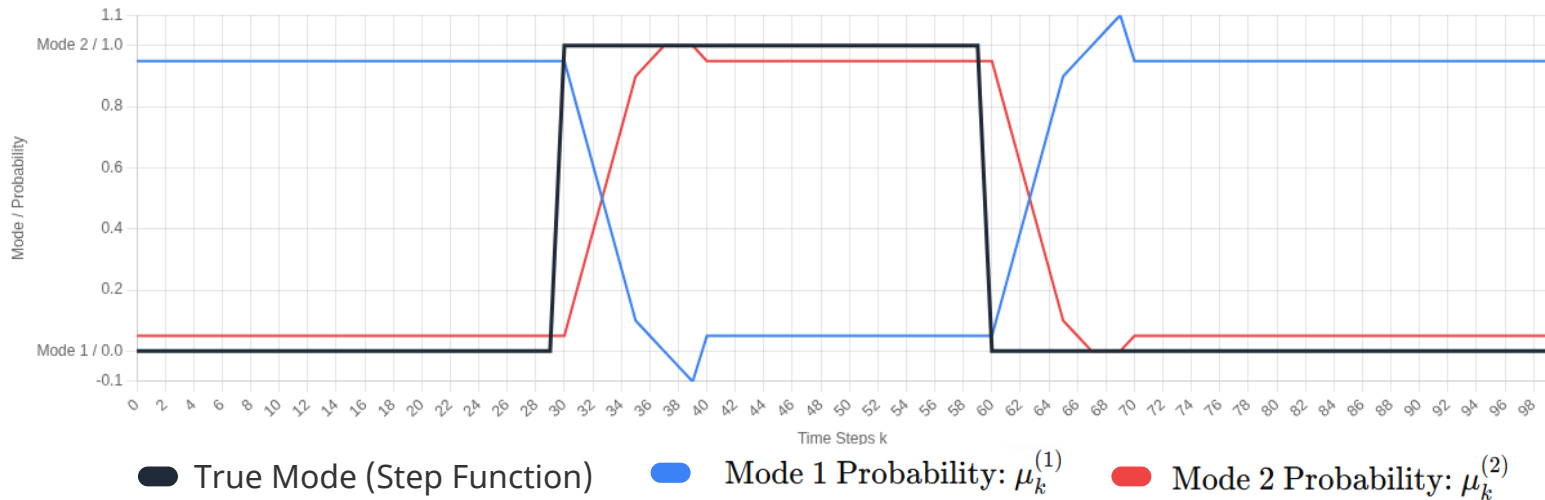
| Assumption | Classification | Risk when Violated | Mitigation Strategies |
|--|----------------|---|--|
| Bounded Model Mismatch Differences between actual system and model are finite and bounded | idealized | Severe: Divergence, tracking failure Algorithm instability and rapid degradation of state estimation quality | <ul style="list-style-type: none"> • Model validation across various operating conditions • Covariance inflation to offset model uncertainty • Multiple model design robust to transient responses and external changes |
| Gaussian Measurement Noise Measurement noise follows white Gaussian distribution | empirical | Moderate: Reduced estimation accuracy Non-Gaussian noise degrades optimality and distorts Kalman gains | <ul style="list-style-type: none"> • Application of robust filtering techniques • Integration with H_∞ methodologies • Outlier removal and measurement gating |
| Mode Transition Matrix (Π) Accuracy Markov transition probabilities reflect actual mode changes | heuristic | Moderate: Delayed mode switching Delayed convergence of mode probabilities and increased transient errors | <ul style="list-style-type: none"> • Adaptive transition matrix updates • Mode probability thresholding • Learning from mode transition history |
| Finite Model Approximation Representing continuous system with finite discrete modes | necessary | Low: Approximation errors Performance degradation in intermediate regions and incomplete mode transitions | <ul style="list-style-type: none"> • Optimization of mode count • Precise model gridding • Continuous interpolation techniques |

True Mode Trajectory vs IMM Mode Probability Response

When mode switches occur in a dynamic system, a measurable difference arises between the true mode trajectory and the IMM-estimated mode probabilities $\mu_k^{(i)}$. The IMM filter exhibits characteristic **delayed responses** and **transient behavior** when tracking abrupt mode transitions.

The response speed and accuracy are directly influenced by the **transition matrix**

Higher off-diagonal probabilities p_{ij} ($i \neq j$) enable quicker detection of mode changes but increase the likelihood of false alarms. Conversely, smaller p_{ij} values yield more conservative transitions but slower responses to actual mode shifts. This visualization illustrates the relationship between actual mode switches (step functions) and the resulting IMM probability curves, emphasizing both the **estimation delay** and **smoothing effect** inherent in the IMM algorithm.



High Transition Probabilities

$$\Pi = \begin{bmatrix} 0.7 & 0.3 \\ 0.3 & 0.7 \end{bmatrix}$$

Faster response to mode changes but higher sensitivity to noise.

Low Transition Probabilities

$$\Pi = \begin{bmatrix} 0.95 & 0.05 \\ 0.05 & 0.95 \end{bmatrix}$$

Smoother response with less sensitivity to noise but slower detection of actual mode changes.

IMM Applications Overview

The Interacting Multiple Model (IMM) filter is utilized across various applications due to its rapid mode transition detection and multiple model fusion capabilities. It excels particularly in maneuvering target tracking, fault diagnosis, and operating mode recognition. IMM effectively models and estimates the hybrid dynamics of complex systems.

1 Key Application Areas

| | |
|----------------------------------|---|
| Aerospace Target Tracking | Tracking aircraft and satellites with abrupt maneuver changes, using multiple motion models |
| Vehicle Navigation | Position estimation in diverse driving environments including highways, intersections, and turning scenarios |
| Robotic Fault Detection | Real-time detection of sensor and actuator malfunctions, distinguishing normal/abnormal states with multiple models |
| Communication Channel Estimation | Wireless channel state estimation with transitions between fading and non-fading modes |
| Biosignal Monitoring | Distinguishing between normal and abnormal regimes in ECG signals and state estimation |

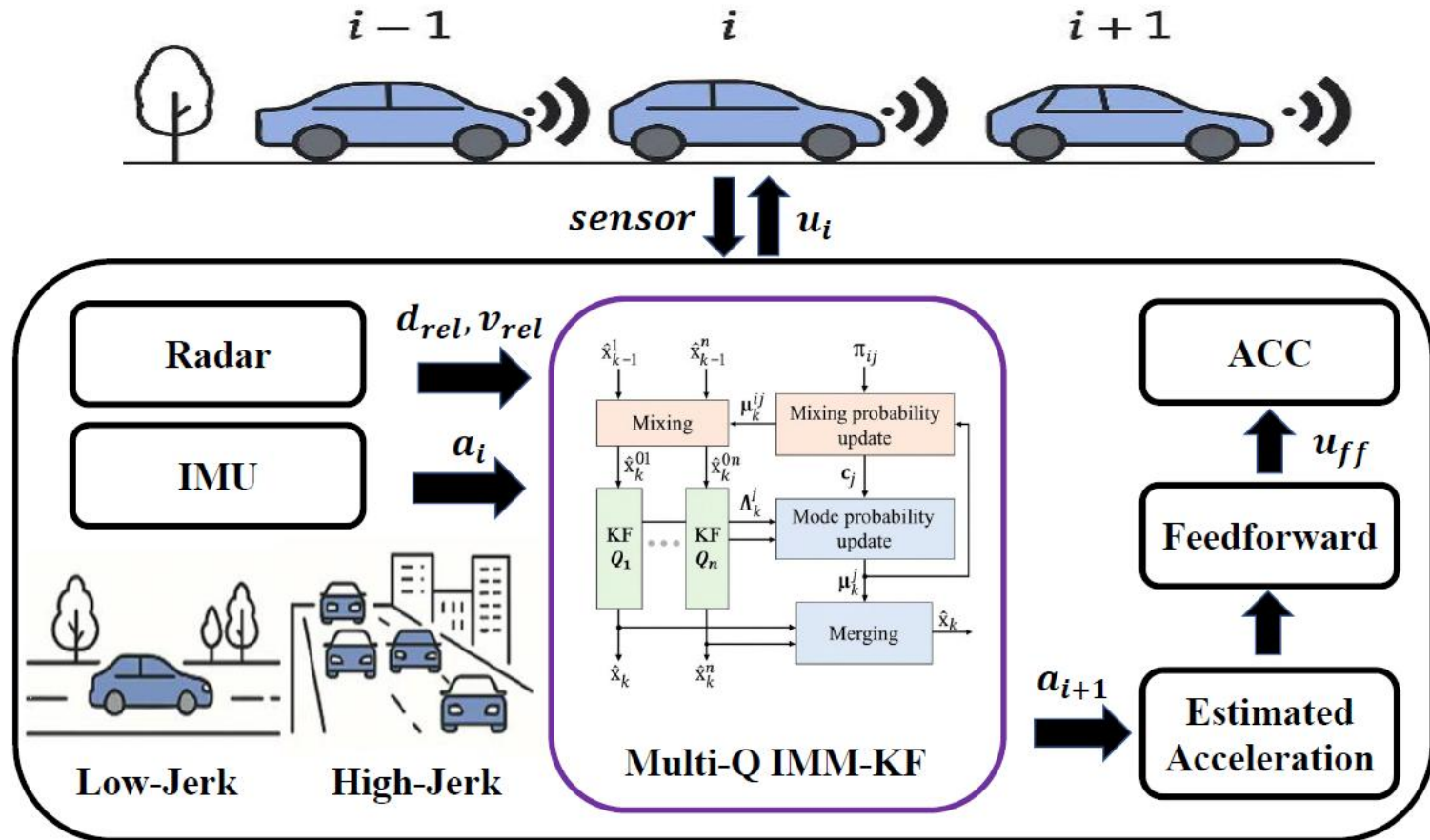
Performance Metrics and Evaluation Methods

| | |
|--|---|
| RMSE Root Mean Square Error for evaluating state estimation accuracy | NEES Normalized Estimation Error Squared for filter consistency assessment |
| Mode Switch Delay Time elapsed from actual mode change to detection | Computational Complexity Computational load increase with model count, real-time feasibility evaluation |

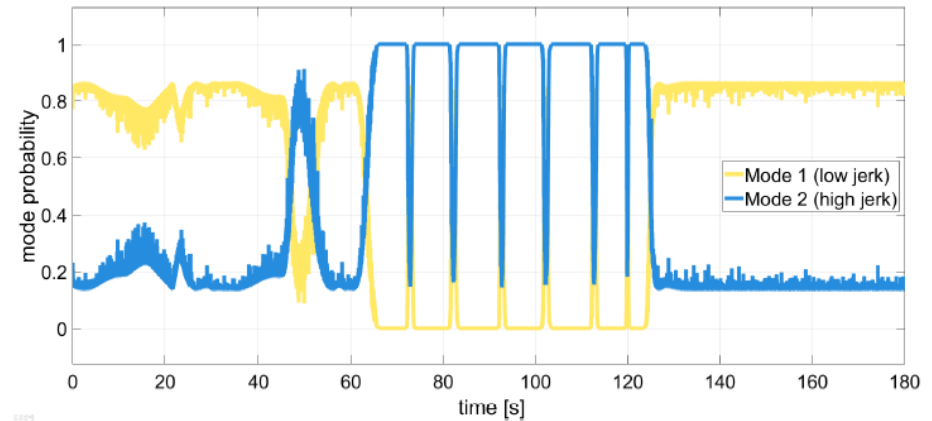
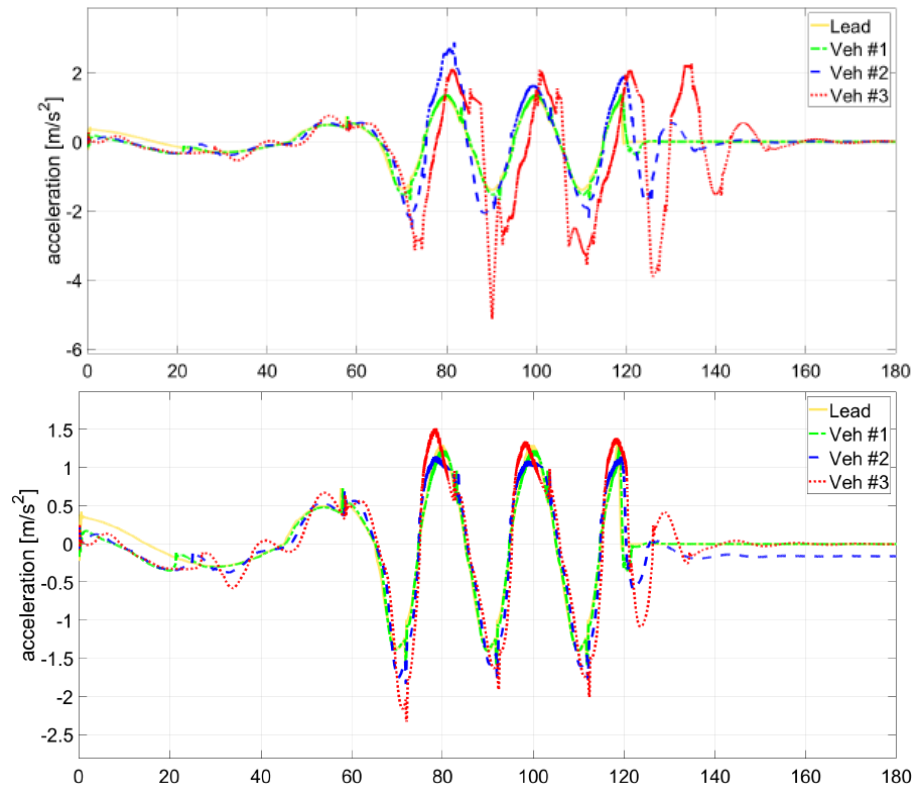
2 Design and Implementation Tips

| | |
|--------------------------------|--|
| Model Selection and Complexity | Start with minimal model set, expand progressively |
| Transition Matrix Tuning | Begin with diagonal elements in 0.9-0.95 range, reflect mode persistence |
| Mode Probability Floor | Set $\mu_{\min} > 10^{-4}$ for numerical stability |
| Mode-Specific Covariance | Differentiate Q, R matrices according to each mode's characteristics |

Example: Multi-Q IMM-KF CACC



Example: Multi-Q IMM-KF CACC



[5] CM Kang et al, "Vehicle lateral motion estimation with its dynamic and kinematic models based interacting multiple model filter", 2016 IEEE 55th Conference on Decision and Control (CDC), 2449-2454

[6] CM Kang et al, "UGV status monitoring system using interacting multiple model filter" 2018 18th International Conference on Control, Automation and Systems (ICCAS)

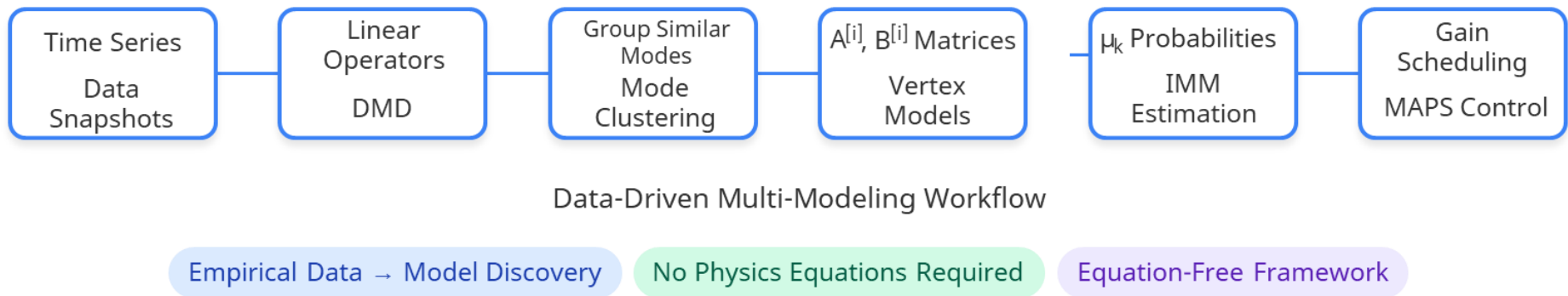
[7] C. Jeong et al, "Fault Diagnosis Algorithm Using Parallel Connected Interacting Multiple Model for Electric Power Steering System", Transactions of the Korean Society of Automotive Engineers - Vol. 29, No. 10, pp.943-950

Data-Driven Multi-model

Motivation for Data-Driven Multi-Modeling

When **physics-based models** are incomplete or uncertain, **data-driven identification** of local linear models becomes essential.

Dynamic Mode Decomposition (DMD) extracts system dynamics directly from data, linking **empirical observations** to **model-based control** without knowing governing equations.



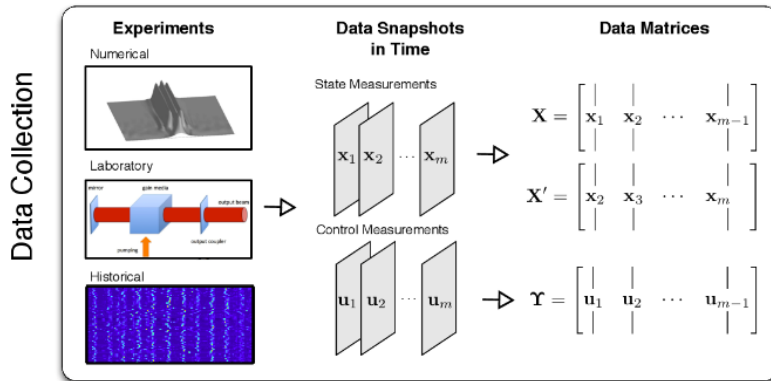
The **DMD-MAPS** approach offers significant advantages when physical modeling is challenging:

- Extracts dominant modes directly from experimental/simulation data
- Groups similar dynamics into discrete operating regimes
- Creates vertex models suitable for IMM estimation and MAPS control
- Enables control design without detailed understanding of the underlying physics

Motivation for Data-Driven Multi-Modeling

Dynamic Mode Decomposition (DMD) models system evolution through a linear mapping between snapshots, using data matrices to capture dominant dynamics.

Singular Value Decomposition (SVD) enables efficient, reduced-order computation of this mapping, extracting key modes without explicitly forming the large system matrix.



Model Reduction

Dynamic Mode Decomposition (DMD)

Find the dynamic properties of \mathbf{A}

$$\mathbf{X}' = \mathbf{A}\mathbf{X}$$

- i. Find the truncated SVD of \mathbf{X}

$$\mathbf{X} \approx \begin{bmatrix} | & | & | \\ \mathbf{U}_1 & \mathbf{U}_2 & \mathbf{U}_r \\ | & | & | \end{bmatrix} \begin{bmatrix} \sigma_1 & & \\ \sigma_2 & & \\ & \ddots & \\ & & \sigma_r \end{bmatrix} \begin{bmatrix} - & - & - \\ \mathbf{V}_1 & & \\ \vdots & & \\ - & - & - \end{bmatrix}$$

- ii. Compute reduced-order approximation $\tilde{\mathbf{A}}$

$$\tilde{\mathbf{A}} = \tilde{\mathbf{U}}\mathbf{X}'\tilde{\mathbf{V}}\tilde{\Sigma}^{-1}$$

- iii. Investigate the dynamic properties of $\tilde{\mathbf{A}}$

$$\tilde{\mathbf{A}}\mathbf{W} = \mathbf{W}\Lambda$$

- iv. Solve for the dynamic modes of \mathbf{A}

$$\Phi = \mathbf{X}'\mathbf{V}\Sigma^{-1}\mathbf{W}$$

DMD with Control (DMDc)

Find the dynamic properties of \mathbf{A} and \mathbf{B}

$$\mathbf{X}' = \mathbf{A}\mathbf{X} + \mathbf{B}\mathbf{Y}$$

- i. Construct the input data matrix

$$\Omega = \begin{bmatrix} \mathbf{X} \\ \mathbf{Y} \end{bmatrix}$$

- ii. Find the truncated SVD of input matrix Ω

$$\Omega \approx \tilde{\mathbf{U}}\tilde{\Sigma}\tilde{\mathbf{V}}^* = \begin{bmatrix} \tilde{\mathbf{U}}_1 \\ \tilde{\mathbf{U}}_2 \end{bmatrix} \tilde{\Sigma} \tilde{\mathbf{V}}^*$$

- iii. Find the truncated SVD of output matrix \mathbf{X}'

$$\mathbf{X}' \approx \tilde{\mathbf{U}}\tilde{\Sigma}\tilde{\mathbf{V}}^*$$

- iv. Compute reduced-order approximation of \mathbf{A}

$$\tilde{\mathbf{A}} = \tilde{\mathbf{U}}^*\mathbf{X}'\tilde{\mathbf{V}}\tilde{\Sigma}^{-1}\tilde{\mathbf{U}}_1^*\tilde{\mathbf{U}}_1$$

- v. Investigate the dynamic properties of

$$\tilde{\mathbf{A}}\mathbf{W} = \mathbf{W}\Lambda$$

- vi. Solve for the dynamic modes of

$$\Phi = \mathbf{X}'\mathbf{V}\Sigma^{-1}\tilde{\mathbf{U}}_1^*\tilde{\mathbf{U}}\mathbf{W}$$

DMD with Control (DMDc) Formulation

DMD with Control (DMDc) extends standard DMD by adding **control inputs**, separating intrinsic dynamics from actuation effects.

It estimates both **system matrix AAA** and **control matrix BBB** directly from data, providing a **causal input–output model** suitable for predictive control and system identification.

Key DMDc Equations

$$Y = AX + B\Upsilon$$

$$\Omega = \begin{bmatrix} X \\ \Upsilon \end{bmatrix} \approx \tilde{U}\tilde{\Sigma}\tilde{V}^T$$

$$[A \ B] = G = Y\tilde{V}\tilde{\Sigma}^{-1}\tilde{U}^T$$

$$\bar{A} = Y\tilde{V}\tilde{\Sigma}^{-1}\tilde{U}_1^T, \quad \bar{B} = Y\tilde{V}\tilde{\Sigma}^{-1}\tilde{U}_2^T$$

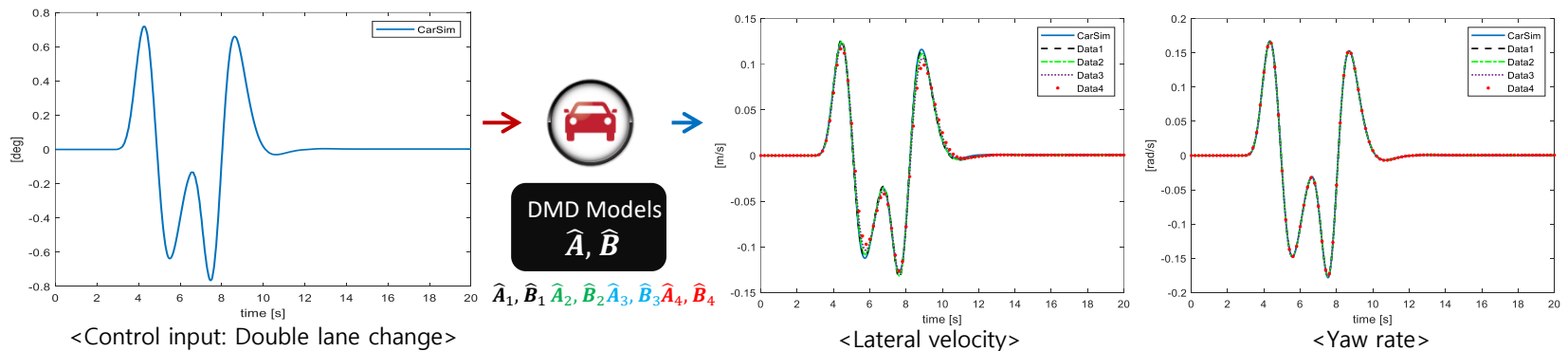
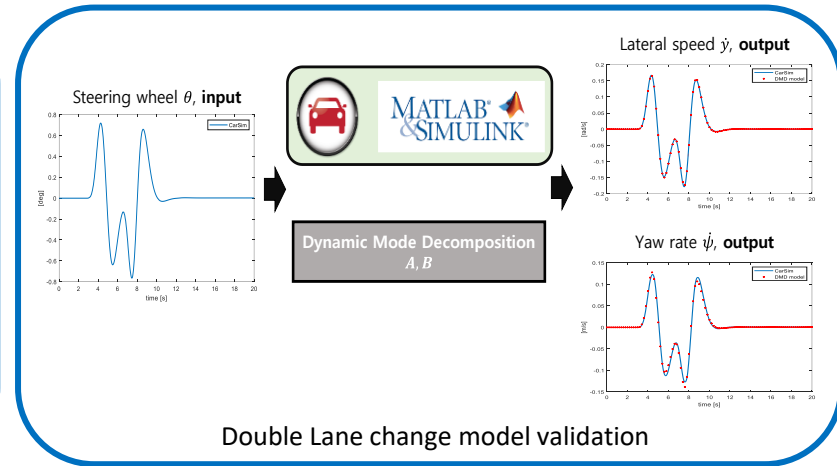
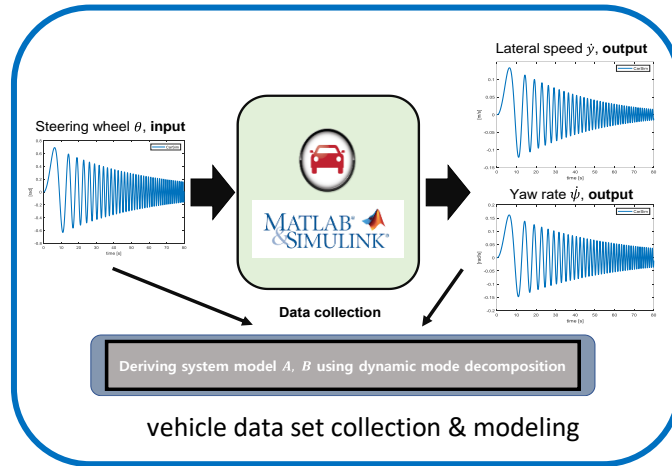
Where:

- $X = [x_1, x_2, \dots, x_{m-1}]$ – State snapshot matrix
- $Y = [x_2, x_3, \dots, x_m]$ – Shifted state snapshot matrix
- $\Upsilon = [u_1, u_2, \dots, u_{m-1}]$ – Control input snapshot matrix
- Ω – Vertically stacked state and input snapshots
- $\tilde{U}, \tilde{\Sigma}, \tilde{V}$ – SVD components of Ω
- $\tilde{U}_1^T, \tilde{U}_2^T$ – Partitioned rows of \tilde{U}^T corresponding to states and inputs

Assumption Classification:

- Empirical: Input excitation must be sufficient for identifiability
- Heuristic: Local linear approximation validity

Example: DMD with Control



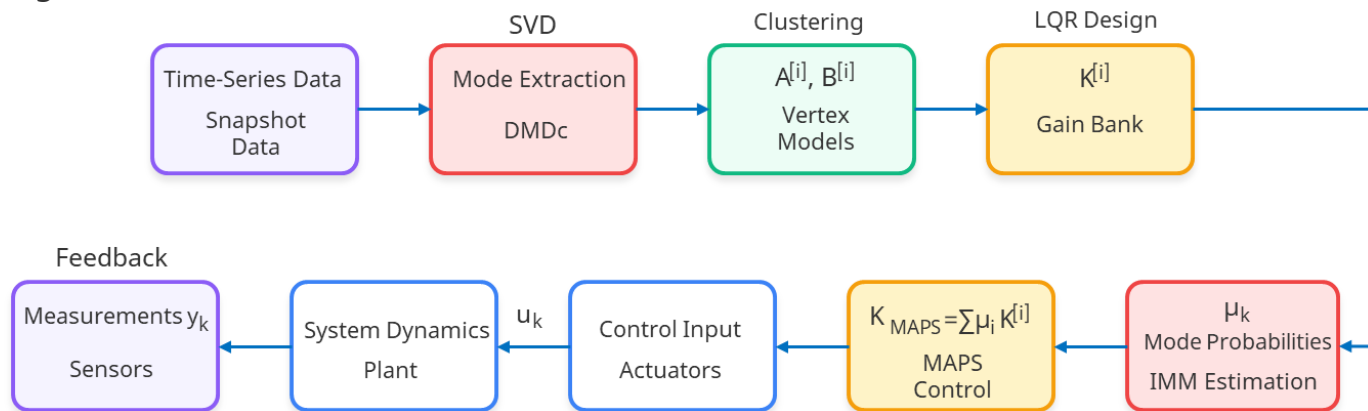
[8] G Kim et al, "Vehicle's lateral motion control using dynamic mode decomposition model predictive control for unknown model", International Journal of Automotive Technology 25 (5), 999-1009

MAPS: A Mode-Aware Probabilistic Scheduling Framework

Integrating DMD with IMM-LPV-MAPS Framework

The **DMD-based MAPS framework** provides a complete data-driven pipeline from measurement data to adaptive control. This integration uniquely bridges empirical dynamics extraction with probabilistic control adaptation, without requiring explicit physics-based modeling.

The modularity of this approach allows for independent refinement of model identification (DMD), mode estimation (IMM), and control synthesis (MAPS) components, making the framework adaptable to various applications with minimal retraining.



Framework Limitations:

Approximate Linearity

DMD assumes locally linear dynamics - validity decreases far from operating points

Full-State Assumption

Requires complete state observation or state estimation via observers

Empirical Validation

Requires experimental confirmation beyond theoretical guarantees

MAPS Controller Definition and Closed-Loop Dynamics

MAPS (Mode-Aware Probabilistic Scheduling) combines vertex gains using **mode probabilities** from the IMM estimator, forming a **probabilistic adaptive controller** that transitions smoothly between regimes.

It generalizes traditional LPV control by replacing deterministic weights $\xi(\rho)$ with probabilistic estimates $\mu_k^{(i)}$, ensuring stability and adaptability even under mode uncertainty.

MAPS Controller Formulation and Closed-Loop Dynamics

$$K_{\text{MAPS}} = \sum_{i=1}^r \mu_k^{(i)} K^{[i]}$$
$$x_{k+1} = \left(\sum_{i=1}^r \mu_k^{(i)} A_{\text{cl}}^{[i]} \right) x_k$$
$$A_{\text{cl}}^{[i]} = A^{[i]} + B^{[i]} K^{[i]}$$

Where:

- $\mu_k^{(i)}$: Mode probability of the i -th model at time step k , obtained from the IMM estimator
 $0 \leq \mu_k^{(i)} \leq 1, \sum_i \mu_k^{(i)} = 1$
- $K^{[i]} \in \mathbb{R}^{m \times n}$: Vertex i control gain matrix
- $A^{[i]} \in \mathbb{R}^{n \times n}$: Vertex i system matrix
- $B^{[i]} \in \mathbb{R}^{n \times m}$: Vertex i input matrix
- $A_{\text{cl}}^{[i]} \in \mathbb{R}^{n \times n}$: Vertex i closed-loop matrix

Assumption Classification:

- **Sufficient:** Convex stability with common P
- **Empirical:** Mode probabilities μ represent true mode likelihoods

Probabilistic Scheduling: μ Replaces $\xi(\rho)$

Traditional **LPV control** relies on deterministic scheduling weights $\xi(\rho)$ that require precise parameter measurements, assuming full knowledge of the operating point — often unrealistic in systems with hidden or uncertain modes.

MAPS replaces this with **IMM-derived mode probabilities (μ)** as scheduling weights, enabling continuous gain adaptation even when the true mode is uncertain.

By using **probabilistic mode estimates** instead of explicit measurements, MAPS achieves smoother transitions and greater robustness while preserving LPV stability guarantees.

Key Equation: LPV vs MAPS Controller Formulation

$$\text{LPV Controller: } K_{\text{LPV}} = \sum_{i=1}^r \xi_i(\rho_k) K^{[i]}$$

$$\text{MAPS Controller: } K_{\text{MAPS}} = \sum_{i=1}^r \mu_k^{(i)} K^{[i]}$$

Where:

- $\xi_i(\rho_k)$: Deterministic scheduling weight for vertex i (function of scheduling parameter ρ_k)
- $\mu_k^{(i)}$: Mode probability for mode i at time k , obtained from the IMM estimator
- $K^{[i]} \in \mathbb{R}^{m \times n}$: Vertex controller gain for mode/vertex i
- r : Number of modes/vertices in the system model

Assumption Classification:

- **Heuristic:** Mode probabilities converge to true modes
- **Empirical:** IMM accuracy sufficient for control

- Mode probability stationarity directly influences transient response quality.
- After a mode switch, the mode probability μ requires time to converge to the new true mode distribution.
- This convergence rate depends on IMM tuning parameters (Q, R, Π) and the signal-to-noise ratio.
- Additionally, time delays between mode detection (μ update) and control application can degrade performance.

Stability via Expected Lyapunov Decrease

In **MAPS**, stability is ensured via the **expected Lyapunov decrease criterion**, guaranteeing average energy decay despite probabilistic mode switching.

Even with noise in mode probabilities μ , the system maintains **bounded expected Lyapunov values**, offering robustness against estimation uncertainty.

Key Stability Conditions

$$\mathbb{E}[V_{k+1}] - V_k \mid x_k < 0$$

$$V(x) = x^T P x, \quad P = P^T \succ 0$$

$$\mathbb{E}[A_{\text{cl}}(\mu_k)] \approx \sum_i \mathbb{E}[\mu_k^{(i)}] A_{\text{cl}}^{[i]}$$

$$\mathbb{E}[\mu_k^{(i)}]^T [A_{\text{cl}}^{[i]T} P A_{\text{cl}}^{[i]} - P] \mathbb{E}[\mu_k^{(i)}] \prec 0$$

Where:

- $V(x)$: Quadratic Lyapunov function
- $P \in \mathbb{R}^{n \times n}$: Positive definite matrix
- $\mathbb{E}[\cdot]$: Expectation over the mode probability distribution
- $\mu_k^{(i)}$: Mode probability for mode i at time k
- $A_{\text{cl}}^{[i]}$: Closed-loop system matrix for mode i
- σ_μ^2 : Variance of mode probability (affects stability margin)

Assumption Classification:

- **Sufficient:** Bounded μ variance
- **Empirical:** Fast μ convergence

Convexity-Based Sufficient Condition

For **MAPS stability**, convexity of quadratic forms in **mode probabilities** (μ) allows simplified verification via **Jensen's inequality**, linking vertex and overall stability.

This ensures stability inheritance from local controllers but is **conservative**, since one Lyapunov function must hold for all modes.

Key Inequality: Convexity-Based Upper Bound

$$A_{\text{cl}}(\mu)^T P A_{\text{cl}}(\mu) - P \preceq \sum_{i=1}^r \mu_k^{(i)} \left[A_{\text{cl}}^{[i]T} P A_{\text{cl}}^{[i]} - P \right]$$

Sufficient condition: Find $P = P^T \succ 0$ such that

$$A_{\text{cl}}^{[i]T} P A_{\text{cl}}^{[i]} - P \prec 0, \quad \forall i = 1, \dots, r$$

Where:

- $A_{\text{cl}}(\mu) = \sum_{i=1}^r \mu_k(i) A_{\text{cl}}^{[i]}$: Probabilistically blended closed-loop dynamics
- $A_{\text{cl}}^{[i]} = A^{[i]} + B^{[i]} K^{[i]}$: Vertex closed-loop system matrix
- $\mu_k(i)$: Mode probability from IMM estimator ($\sum_{i=1}^r \mu_k(i) = 1$)
- $P \succ 0$: Common Lyapunov matrix (symmetric positive definite)
- r : Number of operating modes/vertices

Assumption Classification:

- **Sufficient:** Common P existence
- **Heuristic:** Convexity dominates cross-terms

MAPS Assumption Taxonomy

The **MAPS framework** depends on several key assumptions that shape its **theoretical guarantees** and **practical applicability**.

The table below summarizes these assumptions, their classifications, justifications, and potential risks, bridging the **IMM (estimation)** and **LPV (control)** domains.

| Assumption | Classification | Justification | Violation Risk |
|--|-----------------------|---|--|
| Common $P > 0$ exists. Single quadratic Lyapunov function satisfies all vertex stability conditions. | Sufficient | <ul style="list-style-type: none"> Ensures convex stability proof Enables LMI-based verification Simplifies controller synthesis | Moderate/High: Conservative design. May fail for switchable systems even when stable solutions exist. |
| Vertex stabilizability. Each vertex model $(A^{[i]}, B^{[i]})$ must be stabilizable. | Necessary | <ul style="list-style-type: none"> Each mode must be controllable Required for vertex controller design Fundamental for any control approach | Critical: System cannot be stabilized. Must redesign physical system or actuator configuration if violated. |
| Bounded estimation error $\ e_k\ < \varepsilon \ e_k\$ (where e_k is IMM estimation error) | Idealized | <ul style="list-style-type: none"> Perfect IMM unrealistic Needed for theoretical guarantees Basis for separation principle | Moderate: Tracking errors. Robust margin needed in practice to accommodate estimation bias. |
| Π ergodic. Markov transition matrix has no absorbing states. | Sufficient | <ul style="list-style-type: none"> Ensures long-term mode coverage Guarantees all modes reachable Supports theoretical convergence | Low: Limited mode exploration. Not required if system operates in transient mode only. |
| Lipschitz continuous gains $\ K(\mu_1) - K(\mu_2)\ \leq L \ \mu_1 - \mu_2\$ | Heuristic | <ul style="list-style-type: none"> Smooths gain transitions Prevents control chattering Ensures bounded control effort | Moderate: Control spikes. Sharp μ jumps may destabilize system or saturate actuators. |
| $T_{\text{IMM}} < T_{\text{ctrl}}$. IMM update rate faster than control rate. | Recommended heuristic | <ul style="list-style-type: none"> IMM updates faster than control Ensures current mode estimates Reduces delays in adaptation | Moderate: Sluggish response. Delays reduce mode tracking accuracy during transitions. |

Smoothing Heuristics: Softmax and μ_{\min}

Sudden **gain switching** can cause instability or oscillation.

To prevent this, apply **smoothing** to mode probabilities before gain interpolation — e.g., **softmax with temperature control** or **minimum probability floors**.

These ensure **gradual mode transitions** and prevent singularities by keeping all modes slightly active.

Key Smoothing Transformations

$$\tilde{\mu}^{(i)} = \frac{\exp(\beta \mu^{(i)})}{\sum_j \exp(\beta \mu^{(j)})}$$
$$\mu_k^{(i)} \geq \mu_{\min}, \quad \forall i$$

Where:

- $\tilde{\mu}^{(i)}$: Smoothed mode probability for mode i
- $\mu^{(i)}$: Original mode probability from IMM filter
- β : Temperature parameter (lower $\beta \rightarrow$ smoother distribution)
- μ_{\min} : Minimum probability floor (typically 0.05–0.10)

Original probabilities:

$$\mu = [0.01, 0.99] \quad (\text{near bang-bang})$$

After softmax smoothing ($\beta = 0.5$):

$$\tilde{\mu} = [0.1, 0.9] \quad (\text{smoother transition})$$

IMM Filter Tuning Parameters

IMM filter tuning critically affects **mode sensitivity**, **convergence**, and **noise robustness**.

The table shows **empirical tuning ranges** and **trade-offs** as a **baseline**, though **fine-tuning** is needed for each application.

IMM Tuning Parameters Summary

| Parameter | Empirical Range | Sensitivity | Performance Impact | Recommendations |
|--|--|-------------|---|---|
| Process Noise (Q) | 10^{-6} to $10^{-2} \times I_n$ | High | <ul style="list-style-type: none"> Large Q → fast adaptation but noisy estimates Small Q → smooth estimates but slow adaptation | Start with $10^{-4} \times I_n$ for typical tracking; increase for highly maneuvering targets |
| Measurement Noise (R) | Sensor-dependent; typically $(0.1\sigma_{\text{meas}})^2$ to $(2\sigma_{\text{meas}})^2$ | High | <ul style="list-style-type: none"> Large R → robust to outliers but sluggish response Small R → fast response but noise-sensitive | Match to actual sensor specs; slightly overestimate for robustness |
| Transition Matrix (Π) | $\pi_{ii} \in [0.7, 0.99]$; $\pi_{ij} = (1 - \pi_{ii})/(r - 1)$ | Medium | <ul style="list-style-type: none"> High diagonals → stable modes, slow transitions Low diagonals → fast switching, less stable | Set based on expected mode sojourn times: $\pi_{ii} = 1 - 1/E[\text{sojourn time in mode } i]$ |
| Initial Mode Probabilities (μ_0) | $\mu_i(0) \geq 0.05$; $\sum \mu_i(0) = 1$ | Low | <ul style="list-style-type: none"> Equal μ_0 → unbiased start Biased μ_0 → faster convergence if prior info exists | Use equal μ_0 when no prior knowledge; avoid exact 0 or 1 values |
| Softmax Temperature (τ) | 0.1 to 2.0 | Medium | <ul style="list-style-type: none"> High τ → smoother transitions, less decisive mode selection Low τ → sharper decisions but risk of chattering | Start with $\tau = 1.0$; adjust based on observed transitions. Lower τ for decisive mode identification |

MAPS as LPV Generalization for Uncertain Mode Probabilities

MAPS extends the **LPV control** framework by using **probabilistic mode estimates** instead of deterministic variables, maintaining convexity while handling **mode uncertainty**.

Structural Comparison: LPV vs MAPS

LPV Controller

$$K_{LPV} = \sum_i \xi_i(\rho) K[i]$$

Requires accurate ρ measurement



MAPS Controller

$$K_{MAPS} = \sum_i \mu_i K[i]$$

Uses IMM-estimated probabilities μ

1 Key Concepts and Differences

| | |
|------------------------------|--|
| LPV Control Approach | Requires deterministic scheduling variable $\rho(t)$ with accurate real-time measurement |
| MAPS Transformation | Replaces ρ with probabilistic mode estimates $\mu(t)$ from IMM filter, maintaining convex structure |
| Core Advantages | Robustness to mode uncertainty, smooth transitions without explicit ρ sensors, seamless estimation-control integration |
| Framework Limitations | Conservatism from common P assumption, dependency on IMM filter accuracy, mode excitation requirements for μ convergence |
| Domain Extension | Broadens LPV applicability to systems with hidden/uncertain operating modes |

2 Implementation Considerations

| | |
|------------------------------|--|
| Mode Discovery | Can leverage DMD for data-driven mode identification when physical modes are unclear |
| Real-time Performance | IMM computational overhead must be balanced with control rate requirements |
| Stability Analysis | Requires stochastic extensions of deterministic LPV stability conditions |

LMI-Based Stability Verification (Common P for Vertices)

LMI enables off-line **stability verification** of MAPS by finding a **common Lyapunov function** across vertex systems.

It converts stability checking into a **convex optimization**, guaranteeing **quadratic stability** before controller deployment.

Key LMI Stability Condition

Find $P = P^T > 0$ such that: $(A_{cl}^{[i]})^T P A_{cl}^{[i]} - P \prec 0, \quad i = 1, \dots, r$

Equivalent strict feasibility form: $(A_{cl}^{[i]})^T P A_{cl}^{[i]} - P + \varepsilon I \prec 0$

Where:

- $P \in \mathbb{R}^{n \times n}$ – Symmetric positive definite matrix (decision variable)
- $A_{cl}[i] = A[i] + B[i]K[i]$ – Closed-loop system matrix for vertex i
- r – Number of vertex models or operating modes
- $\varepsilon > 0$ – Small positive number for strict feasibility
- $X \prec 0$ – Negative definite matrix constraint

Assumption Classification:

Sufficient: Common P existence guarantees stability

Necessary: Each vertex must be stabilizable

Practical Implementation Checklist:

- Verify vertex stabilizability first.
- Solve LMI with CVX, YALMIP, or Mosek.
- Check numerical conditioning; large condition numbers ($>10^6$) may indicate instability.

Assumption–Violation Risk–Mitigation Strategy Table

MAPS relies on key **assumptions** that affect its **theoretical validity** and **practical robustness**.

The table summarizes these assumptions, their **risks**, and **mitigation strategies** for real-world use.

| Assumption | Classification | Violation Risk | Mitigation Strategy |
|--|-------------------|--|--|
| Common $P > 0$ exists — A single Lyapunov matrix satisfies all vertex stability conditions simultaneously | Sufficient | Moderate: Over-conservatism; LMI may fail even if the system is stabilizable | <ul style="list-style-type: none">• Use parameter-dependent $P(p)$ or mode-dependent P_i• Apply switched Lyapunov analysis• Improve numerical conditioning via regularization |
| Vertex stabilizability — Each vertex model $(A[i], B[i])$ is stabilizable with some gain $K[i]$ | Necessary | Severe: Fundamental instability; no convex combination of unstable vertices can be stabilized | <ul style="list-style-type: none">• Add actuators to improve controllability• Redesign operating region to avoid uncontrollable zones• Verify controllability at all vertices before MAPS design |
| Bounded estimation error — IMM filter produces state estimates with bounded error covariance | Idealized | Severe: Large IMM bias in high-noise or fast modes; incorrect mode probabilities can destabilize control | <ul style="list-style-type: none">• Increase measurement frequency• Fine-tune Q/R matrices using empirical noise data• Add robustness margin in $K[i]$ design via H_∞ methods |
| Π ergodic transition matrix — All modes are reachable with non-zero probability | Sufficient | Moderate: Rare modes may never be visited; IMM may not converge to critical modes | <ul style="list-style-type: none">• Ensure training data covers all possible modes• Adjust off-diagonal Π terms to improve rare mode detection• Implement minimum mode-probability thresholds |
| Lipschitz K smoothness — Control gain changes smoothly with respect to mode probabilities | Heuristic | Moderate: Sharp μ jumps cause control spikes and excite high-frequency dynamics | <ul style="list-style-type: none">• Apply softmax transformation with temperature tuning• Enforce $\mu_{\min} > 0$ to prevent zero-weight modes• Limit |
| $T_{\text{IMM}} \leq T_{\text{ctrl}}$ — IMM update rate at least matches control rate | Heuristic | Low: Delayed μ lags true mode; control responds to outdated probabilities | <ul style="list-style-type: none">• Synchronize IMM and control sampling• Implement μ prediction based on recent trends• Increase IMM execution frequency relative to control |

Summary: Smooth Adaptive Control via IMM-LPV Coupling

MAPS integrates **estimation and control**, using **IMM-based mode probabilities** as **LPV weights** to enable **adaptive, sensor-free control** with **guaranteed stability** under dynamic uncertainty.

1 MAPS Core Idea

Probabilistic Adaptation

Use IMM-estimated mode probabilities μ directly as LPV gain scheduling weights

Direct Coupling

Eliminate intermediate scheduling variable measurement (ρ) requirement

Fundamental Equation

$$K_{MAPS} = \sum_i \mu_i K^{[i]} \text{ replacing deterministic } K_{LPV} = \sum_i \xi_i(\rho) K^{[i]}$$

2 Key Benefits

Smooth Transitions

Gain changes follow mode probability evolution, avoiding abrupt switching

Unified Framework

Seamlessly integrates estimation-control without separate scheduler

Provable Stability

Guaranteed by convex LMI conditions with common Lyapunov function

Uncertainty Handling

Naturally robust to mode uncertainty through probabilistic weighting

Summary: Smooth Adaptive Control via IMM-LPV Coupling

MAPS integrates **estimation and control**, using **IMM-based mode probabilities** as **LPV weights** to enable **adaptive, sensor-free control** with **guaranteed stability** under dynamic uncertainty.

3 Expected Performance

| | | |
|---------------------|---|------------|
| Speed of Adaptation | Faster mode detection and response than fixed gain controllers | empirical |
| Robustness | More robust to model uncertainty than pure adaptive laws | sufficient |
| LPV Comparability | Performance approaches LPV when $\mu \approx \xi(\rho)$ with reduced sensing requirements | |
| Transient Behavior | Smooth transitions avoid destabilizing control jumps during mode switches | |

Design Workflow

Off-line Design

1. Define vertex models (physics-based or DMD-extracted)
2. Design vertex gains $K^{[i]}$ via LQR for each model
3. Verify common P LMI feasibility for stability proof
4. Tune IMM filter parameters (Q, R, Π)

On-line Execution

1. Run IMM filter to compute mode probabilities μ_k
2. Interpolate control gains: $K_{MAPS} = \sum_i \mu_i K^{[i]}$
3. Apply control law: $u_k = K_{MAPS} x_k$
4. Update measurements and repeat at next timestep

DMD Integration Highlights

Dynamic Mode Decomposition (DMD) serves as a powerful data-driven complement to the MAPS framework, enabling model identification from experimental data. Its integration bridges the gap between empirical observations and theoretical control design, particularly valuable when analytical models are incomplete or unavailable.

1. Key DMD Integration Benefits

- **Data-Driven Model Discovery**

When physics-based models are incomplete or unavailable, DMDc extracts linear vertex models ($A^{[i]}, B^{[i]}$) directly from snapshot data.
(*empirical*)

- **Modularity**

DMD identification stage decouples from control design—enabling iterative model refinement without redesigning the controller structure.
(*sufficient*)

- **Multi-Mode Extraction**

Clustering DMD eigenvalues by frequency and damping characteristics naturally partitions the system into operating regimes.
(*heuristic*)

2. Practical Considerations

- **Applicability Domains**

Suitable for high-dimensional systems (e.g., fluid flows, power networks) where reduced-order models are needed.

- **Input Requirements**

Requires sufficient input excitation for DMDc identifiability and accurate model extraction.

- **Local Linearity Assumption**

DMD valid only near operating points; limits global accuracy for highly nonlinear systems.

- **Full-State Observability**

Requires full-state measurements or state reconstruction via observers.

- **Validation Need**

Empirical validation is essential to confirm model accuracy before integration with MAPS.

DMD-MAPS Integration Workflow

Snapshot Data → DMD Mode Extraction → Mode Clustering → Vertex Model Bank → IMM Estimation → LPV-Based Control

This pipeline creates an end-to-end framework connecting data collection directly to adaptive control implementation with minimal manual modeling requirements.

Limitations: Common P Conservatism, Estimation Bias, Excitation Requirements

Despite its advantages, the MAPS framework faces several theoretical and practical limitations that impact real-world implementation. These constraints arise from mathematical conservatism, estimation dynamics, data quality requirements, and computational complexity.

1. Common P Conservatism

- **Existence Problem:** A single Lyapunov matrix P for all modes may not exist even when the system is stable. (*structural*)
- **Design Impact:** Leads to LMI infeasibility or overly conservative gain design with reduced performance.
- **Mitigation Approach:** Use mode-dependent P_i or parameter-dependent $P(\mu)$ at the cost of higher complexity.

2. IMM Estimation Bias

- **Convergence Delay:** Finite mode probability convergence time causes transient tracking errors. (*temporal*)
- **Parameter Sensitivity:** Model mismatch or tuning errors (Q, R, Π) accumulate bias during estimation.
- **Fast-Switching Impact:** Bias significantly affects gain interpolation accuracy during rapid mode transitions.

3. DMDc Input Excitation Requirements

- **Identifiability Condition:** Extraction of (A, B) requires persistently exciting input signals u_k . (*information*)
- **Data Quality Issues:** Insufficient excitation \rightarrow rank-deficient $\Omega \rightarrow$ unreliable system models.
- **Practical Challenge:** Generating informative data without destabilizing the plant during identification.

4. Scalability Limitations

- **Computational Complexity:** $O(rn^3)$ cost limits real-time use for large state dimension n or mode count r .
- **Implementation Constraint:** Mode count r typically restricted to $r < 10$ for embedded hardware implementation.
- **Memory Requirements:** Storing multiple vertex gains $K^{[i]}$ and covariance matrices $P^{(i)}$ scales with r .

Research Directions: Koopman-LPV, Transformer-IMM, MPC Fusion

Several promising research directions aim to extend and enhance MAPS capabilities through integration with advanced mathematical frameworks, deep learning, and predictive control techniques. These approaches address current limitations while expanding the application domain.

1. Koopman Operator-based LPV Modeling (Research Stage)

- Lift nonlinear dynamics to infinite-dimensional linear space via Koopman operator.
- Approximate with Extended DMD (EDMD) using observable functions.
- Potentially yields globally valid LPV representation.
- **Challenges:** Observable selection, finite-dimensional approximation accuracy.

2. Deep Learning for Mode Probability Estimation (Early Testing)

- Replace hand-tuned IMM with Transformer/RNN networks learning mode probabilities μ_k from data.
- **Advantages:** Handle non-Gaussian/nonlinear observations, reduce manual tuning.
- **Challenges:** Interpretability, stability guarantees, training data requirements.
- **Current work:** End-to-end differentiable IMM networks with attention mechanisms.

3. MPC Fusion with MAPS (Empirical Validation)

- Use MAPS for fast inner-loop stabilization and MPC for outer-loop optimization.
- **Hierarchical structure:** MPC computes reference r_k ; MAPS tracks with mode-adaptive gains.
- Enables constraint handling while maintaining mode-awareness.
- **Validation:** Empirical testing in automotive and aerospace applications.

4. Robust/Adaptive Extensions (Theoretical Development)

- Online learning of $\Pi, P_i, A^{[i]}$ from streaming data.
- Adaptive horizon MPC with MAPS terminal controller.
- Probabilistic reachability analysis under μ uncertainty.
- Cross-domain applications: energy systems, autonomous vehicles, distributed control.

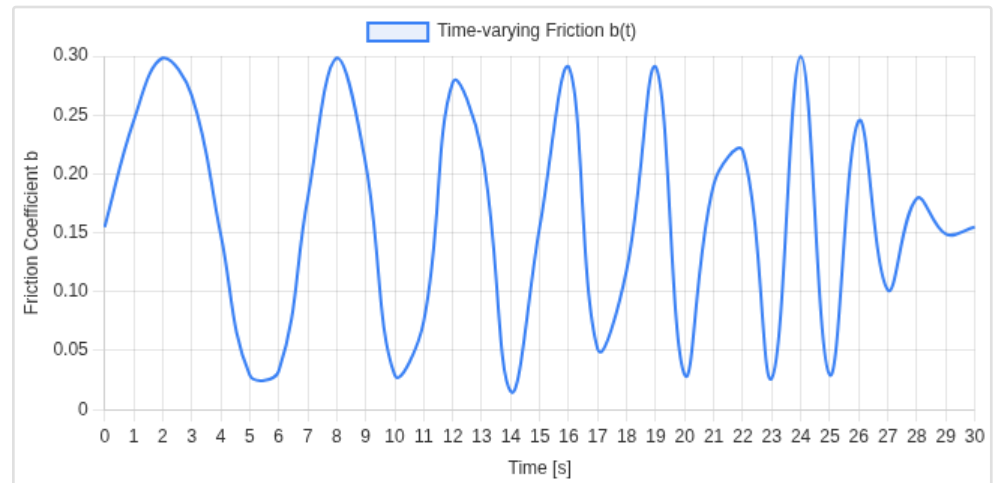
IMM-KF for DC Motor: System Setup & Friction Model

```
dt = 0.01;           % Sampling time [s]
T = 30;              % Total simulation time [s]
t = 0:dt:T;
N = length(t);

% System parameters
J = 0.01;            % Inertia
K = 0.01;            % Torque constant

% Friction coefficient range and time-varying function
b_min = 0.01;
b_max = 0.3;
b_amp = (b_max - b_min)/2;
b_offset = (b_max + b_min)/2;
b_true = b_offset + b_amp .* chirp(t, 0.1, 10, 0.3);

% State-space system matrix function
A_func = @(b) [0, 1; 0, -b/J];
B = [0; K/J];
```



$$A(b) = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{b}{J} \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ \frac{K}{J} \end{bmatrix}$$

DC motor state variables: angular position (θ), angular velocity (ω), control input (u)

Modeling of time-varying friction coefficient $b(t)$ using chirp function

State matrix $A(b)$ forms a Linear Time-Varying (LTV) system structure dependent on friction coefficient

MAPS framework applies IMM-KF for parameter estimation and LPV-LQR for control under varying parameters

IMM Filter Architecture for DC Motor

```
% IMM initialization
n_models = 2; % Two friction models (min, max)
phi_ij = [0.95 0.05; 0.05 0.95]; % Transition matrix
mu = zeros(N, n_models);
mu(1,:) = [0.5 0.5]; % Initial mode probabilities

% System matrices for each model
[Ad1, Bd1] = c2d(A_func(b_min), B, dt);
[Ad2, Bd2] = c2d(A_func(b_max), B, dt);

% Covariances for Kalman filters
Q_proc = diag([1e-6, 1e-6]); % Process noise
R_meas = 1e-5; % Measurement noise

% State and covariance storage
x_hat_prev = {zeros(dims,1), zeros(dims,1)};
P_prev = {eye(dims), eye(dims)};
```

Mode mixing probabilities:

$$\mu_{ij}(k) = \frac{\phi_{ij} \cdot \mu_i(k-1)}{\sum_{i=1}^n \phi_{ij} \cdot \mu_i(k-1)}$$

Normalized mode probabilities:

$$\mu_j(k) = \frac{c_j \cdot \Lambda_j}{\sum_{j=1}^n c_j \cdot \Lambda_j}$$

Interaction

$$\begin{aligned}\mu_{k|k-1}^i &= \sum_j \pi_{ij} \mu_{k-1|k-1}^j \\ \mu_{k-1|k-1}^i &= \frac{\pi_{ij} \mu_{k-1|k-1}^j}{\sum_l \pi_{il} \mu_{k-1|k-1}^l} \\ \tilde{x}_{k-1|k-1}^i &= \sum_j \mu_{k-1|k-1}^j \tilde{x}_{k-1|k-1}^j \\ \hat{P}_{k-1|k-1}^i &= \sum_j \mu_{k-1|k-1}^j \left[\hat{P}_{k-1|k-1}^j + (\tilde{x}_{k-1|k-1}^j - \tilde{x}_{k-1|k-1}^i)(\tilde{x}_{k-1|k-1}^j - \tilde{x}_{k-1|k-1}^i)^T \right]\end{aligned}$$

Updating

$$\begin{aligned}\Lambda_k^i &= \frac{\exp\left[-(1/2)(z_k^i)^T (S_k^i)^{-1} z_k^i\right]}{2\pi S_k^i} \\ S_k^i &= H_i P_i H_i^T + R \\ \mu_{k|k}^i &= \frac{\mu_{k|k-1}^i \Lambda_k^i}{\sum_j \mu_{k|k-1}^j \Lambda_k^j}\end{aligned}$$

Filtering

$$\begin{aligned}\tilde{x}_{k|k-1}^i &= F_i \tilde{x}_{k-1|k-1}^i \\ \tilde{x}_{k|k}^i &= \tilde{x}_{k|k-1}^i + K_i^i \tilde{x}_{k-1|k-1}^i \\ \hat{y}_k^i &= y_k^i - H_i \tilde{x}_{k|k-1}^i\end{aligned}$$

Combination

$$\begin{aligned}\tilde{x}_{k|k} &= \sum_j \mu_{k|k}^j \tilde{x}_{k|k}^j \\ \hat{P}_{k|k} &= \sum_j \mu_{k|k}^j \left[\hat{P}_{k|k}^j + (\tilde{x}_{k|k}^j - \tilde{x}_{k|k})(\tilde{x}_{k|k}^j - \tilde{x}_{k|k})^T \right]\end{aligned}$$

Mode Mixing: Probability-weighted combination of previous state estimates to generate initial values for each filter

Parallel Filtering: Independent Kalman filters for each mode performing state estimation and likelihood calculation

Mode Probability Update: Combining measurement likelihood with prior mode probabilities to update current mode probabilities μ

Automatic Adaptation over time, assigning higher probabilities to modes that better match the actual system state

LPV Gain Scheduling & MAPS Synthesis

```
% LQR gains for min/max friction models
Q_lqr = diag([100 1]); R_lqr = 0.01;
K_min = dlqr(Ad1, Bd1, Q_lqr, R_lqr);
K_max = dlqr(Ad2, Bd2, Q_lqr, R_lqr);

% MAPS gain synthesis (runtime)
K_lpv = mu(k,1)*K_min + mu(k,2)*K_max;
x_est = mu(k,1)*x_hat(:,k,1) + mu(k,2)*x_hat(:,k,2);
u(k) = -K_lpv*(x_est - x_ref(:,k));
```

$$A(\rho) = \sum_{i=1}^2 \xi_i(\rho) A^i$$
$$K(\rho) = \sum_{i=1}^2 \xi_i(\rho) K^i$$

where $\sum \xi_i = 1, \xi_i \geq 0$

Probabilistic weighting prevents abrupt transitions between modes and provides smooth control transitions

LPV (Linear Parameter-Varying) principle guarantees control stability across parameter variations

Utilizes mode probabilities μ from the IMM filter in real-time to synthesize optimal gain scheduling

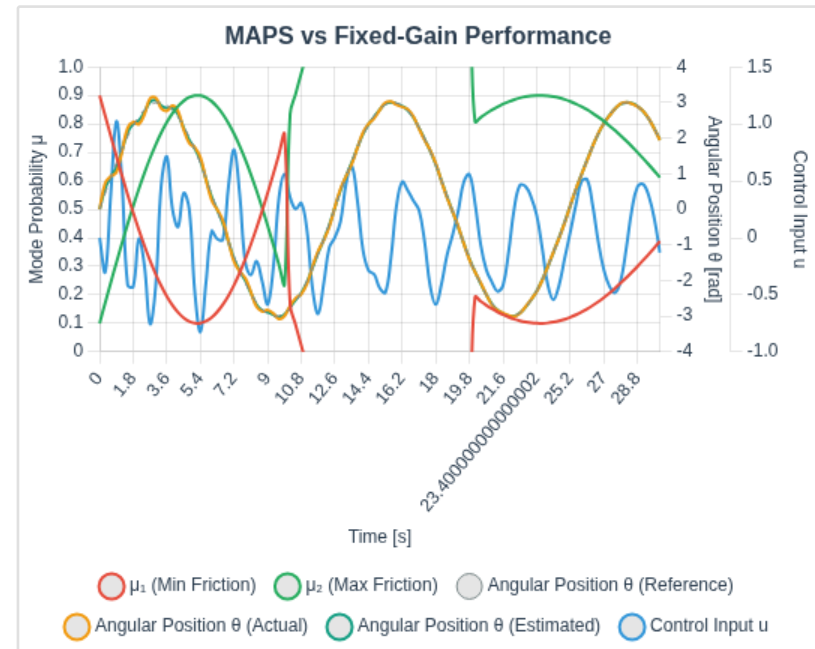
Provides more robust control performance against noise and uncertainty compared to deterministic scheduling

DC Motor Control Results: MAPS vs Fixed-Gain

```
% Simulation loop summary
for k = 2:N
    % Update true system (varying friction)
    [Ad_true, Bd_true] = c2d(A_func(b_true(k-1)), B, dt);
    x_true(:, k) = Ad_true * x_true(:, k-1) + Bd_true * u_lqr(k-1) + w;
    y(:, k-1) = C_true * x_true(:, k-1) + v;

    % MAPS control (IMM-KF + gain scheduling)
    if control_mode == 1
        % IMM mixing, 2xKF, likelihood,  $\mu$  updates
        c_j = zeros(n_models, 1);
        for j=1:n_models
            for i=1:n_models, c_j(j) += phi_ij(i,j)*mu(k-1,i); end
        end
        % Calculate likelihood and normalize probabilities
        lambda = [exp(-0.5*r1'*S1^(-1)*r1)/sqrt(det(2*pi*S1));
                  exp(-0.5*r2'*S2^(-1)*r2)/sqrt(det(2*pi*S2))];
        mu(k,:) = (c_j.*lambda)' / sum(c_j.*lambda);

        % MAPS gain synthesis and control input calculation
        K_lpv = mu(k,1)*K_min + mu(k,2)*K_max;
        x_est = mu(k,1)*x_hat(:,k,1) + mu(k,2)*x_hat(:,k,2);
        u_lqr(k) = -K_lpv * (x_est - x_ref(:,k));
    end
end
```



MAPS benefits: Reduced transient response and improved tracking accuracy even with rapid friction changes

Mode probabilities μ detect real-time friction changes to automatically adjust controller gains

Reduced control input spikes compared to Fixed-gain control, minimizing system stress

Maintains robust state estimation and control performance even under measurement uncertainty

Vehicle Bicycle Model: Road Condition Transitions

```
dt=0.01; T=30; t=0:dt:T; Vx=50/3.6;
m=1274; Izz=1523; lf=1.016; lr=2.578-lf;
Cf_min=6e4; Cf_max=1.3e5; Cr_min=9e4; Cr_max=1.8e5;
tau=2.0; % [s] transition sharpness

% Transition times
t1=6; t2=12; t3=20; t4=26; % phase transitions

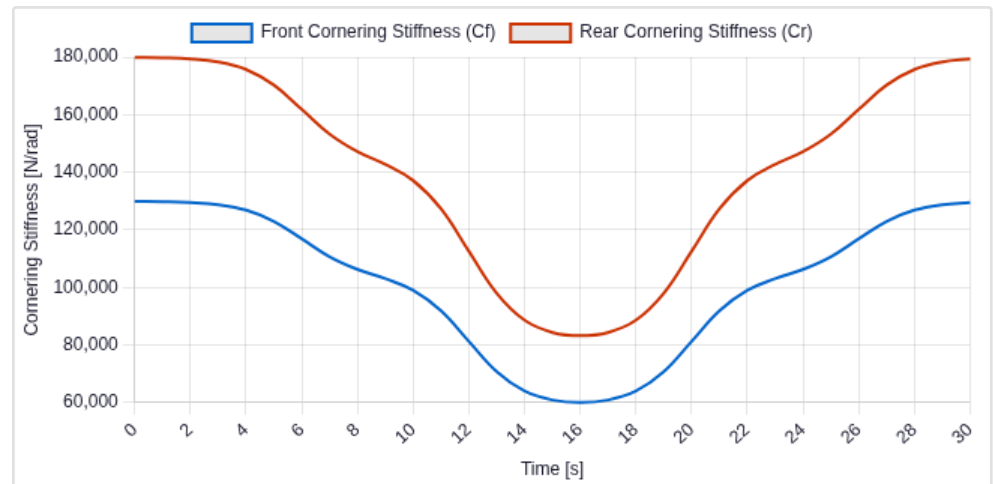
% Smooth window functions using tanh
band = @(a,b) 0.5*(tanh((t-a)/tau) - tanh((t-b)/tau));
before = @(a) 0.5*(1 - tanh((t-a)/tau));
after = @(a) 0.5*(1 + tanh((t-a)/tau));

% Phase weights
w_dry = before(t1) + after(t4);
w_wet = band(t1,t2) + band(t3,t4);
w_ice = band(t2,t3);

% Cornering stiffness profiles
Cf_true = Cf_max*(1.0*w_dry + 0.8*w_wet + 0.45*w_ice);
Cr_true = Cr_max*(1.0*w_dry + 0.8*w_wet + 0.45*w_ice);
```

$$A(C_f, C_r) = \begin{bmatrix} -\frac{2(C_f + C_r)}{mV_x} & -\frac{2(C_f l_f - C_r l_r)}{mV_x} - V_x \\ \frac{2(C_f l_f - C_r l_r)}{I_{zz}} & -\frac{2(C_f l_f^2 + C_r l_r^2)}{I_{zz}} V_x \end{bmatrix}$$

$$B(C_f) = \begin{bmatrix} \frac{2C_f}{m} \\ \frac{2C_f l_f}{I_{zz}} \end{bmatrix}$$



5-phase road condition scenario: Dry → Wet → Ice → Wet → Dry - tanh-based smooth transitions

Lateral bicycle model: states $x = [v_y \text{ (lateral velocity); } r \text{ (yaw rate)}]$, control input δ (steering angle)

Tire cornering stiffness (C_f , C_r) varies with road surface conditions (Dry: 100%, Wet: 80%, Ice: 45%)

Analysis of how cornering stiffness changes affect vehicle lateral stability and control

IMM-KF for Vehicle State Estimation

```
% Vehicle IMM-KF setup
[A1,B1] = AB_func(Cf_min, Cr_min); % Low-stiffness model
[A2,B2] = AB_func(Cf_max, Cr_max); % High-stiffness model
[Ad1,Bd1] = c2d(A1, B1, dt);
[Ad2,Bd2] = c2d(A2, B2, dt);

% Measurement of yaw rate only
C = [0 1]; % x = [v_y; r], measure r only
Q_proc = 1e-6 * eye(2);
R_meas = 1e-5; % scalar measurement noise

% IMM parameters
phi_ij = [0.95 0.05; 0.05 0.95]; % Transition matrix
mu = zeros(N, 2); mu(1,:) = [0.5 0.5];

% Initialize state/covariance for each model
x_hat_prev = {zeros(2,1), zeros(2,1)};
P_prev = {eye(2), eye(2)};
```

Mixing step:

$$c_j = \sum_{i=1}^2 \phi_{ij} \cdot \mu_i(k-1)$$

$$\mu_{ij}(k) = \frac{\phi_{ij} \cdot \mu_i(k-1)}{c_j}$$

Mixed initial conditions:

$$\hat{x}_{0j}(k) = \sum_{i=1}^2 \hat{x}_i(k-1) \cdot \mu_{ij}(k)$$

$$P_{0j}(k) = \sum_{i=1}^2 \mu_{ij}(k) [P_i(k-1) + dx \cdot dx^T]$$

➤ Interaction

$$\mu_{k|k-1}^i = \sum_j \pi_j \mu_{k-|k-1}^j$$

$$\mu_{k-|k-1}^i = \frac{\pi_i \mu_{k-|k-1}^i}{\sum_j \pi_j \mu_{k-|k-1}^j} = \frac{\pi_i \mu_{k-|k-1}^i}{\sum_j \pi_j \mu_{k-|k-1}^j}$$

$$\hat{x}_{k-|k-1}^i = \sum_j \mu_{k-|k-1}^j \hat{x}_{k-|k-1}^j$$

$$\hat{P}_{k-|k-1}^i = \sum_j \mu_{k-|k-1}^j \left[\hat{P}_{k-|k-1}^j + (\hat{x}_{k-|k-1}^j - \hat{x}_{k-|k-1}^i)(\hat{x}_{k-|k-1}^j - \hat{x}_{k-|k-1}^i)^T \right]$$

➤ Updating

$$\Lambda_k^i = \frac{\exp \left[-(1/2) (z_k^i)^T (S_k^i)^{-1} z_k^i \right]}{[2\pi S_k^i]^{1/2}}$$

$$S_k^i = H_i P_i H_i^T + R$$

$$\mu_{k|k}^i = \frac{\mu_{k|k-1}^i \Lambda_k^i}{\sum_j \mu_{k|k-1}^j \Lambda_k^j}$$

➤ Filtering

$$\hat{x}_{k|k-1}^i = F_i \hat{x}_{k-1|k-1}^i$$

$$\hat{x}_{k|k}^i = \hat{x}_{k|k-1}^i + K_i (z_k^i - \hat{y}_k^i)$$

$$\hat{y}_k^i = y_k^i - H_i \hat{x}_{k|k-1}^i$$

➤ Combination

$$\hat{x}_{k|k} = \sum_j \mu_{k|k}^j \hat{x}_{k|k}^j$$

$$\hat{P}_{k|k} = \sum_j \mu_{k|k}^j \left[\hat{P}_{k|k}^j + (\hat{x}_{k|k} - \hat{x}_{k|k}^j)(\hat{x}_{k|k} - \hat{x}_{k|k}^j)^T \right]$$

Single yaw rate measurement: Lateral velocity (v_y) cannot be directly measured but is robustly estimated through IMM-KF

Cornering stiffness adaptation: Two filters (minimum/maximum cornering stiffness) operate in parallel to adapt to different road conditions

Real-time road condition identification: Mode probabilities (μ) automatically track dry → wet → ice → wet → dry transitions

Joint state-parameter estimation: Simultaneous estimation of vehicle states (v_y, r) and cornering stiffness (Cf, Cr) provides foundation for MAPS control

Vehicle MAPS Control Performance

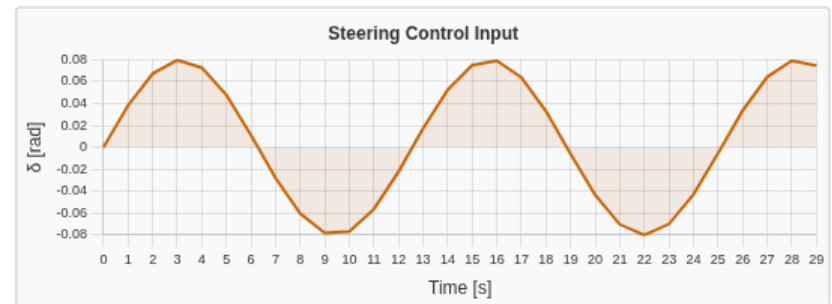
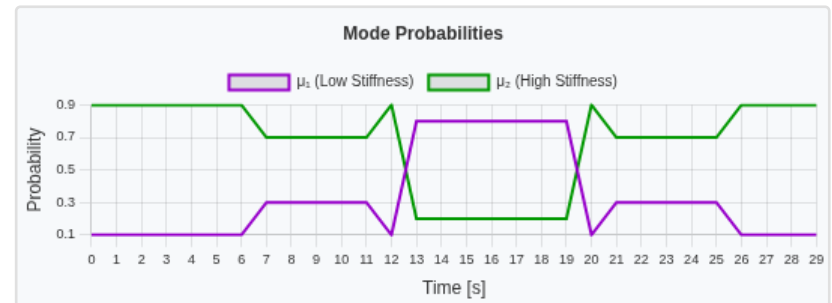
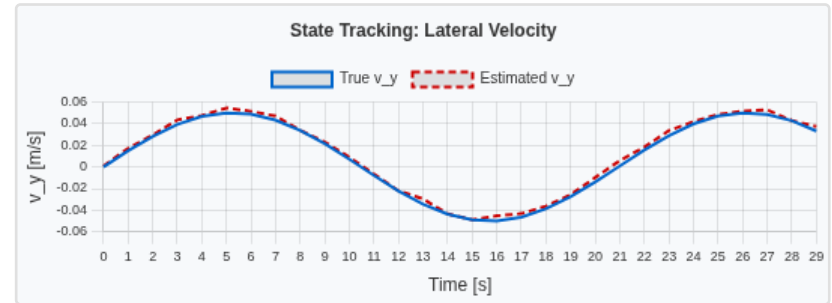
```
% LQR gains and MAPS synthesis
K_min = dlqr(Ad1, Bd1, Q_lqr, R_lqr);
K_max = dlqr(Ad2, Bd2, Q_lqr, R_lqr);

% Control loop (each step)
K_lpv = mu(k,1)*K_min + mu(k,2)*K_max;
r_ref = 0.1*sin(0.5*t); % Yaw rate reference
x_ref = [zeros(1,N); r_ref];
x_est_k = mu(k,1)*x_hat(:,k,1) + mu(k,2)*x_hat(:,k,2);
x_err = x_est_k - x_ref(:,k);
u(k) = -K_lpv * x_err; % MAPS control input

% Calculate RMSE (t >= 0.5s)
idx = find(t >= 0.5);
rmse_vy = sqrt(mean((x_true(1,idx) - x_est_hist(1,idx)).^2));
rmse_r = sqrt(mean((x_true(2,idx) - x_est_hist(2,idx)).^2));
fprintf('RMSE vy: %.4f m/s, RMSE r: %.4f rad/s\n', rmse_vy, rmse_r);
```

$$K_{\text{MAPS}}(k) = \mu_1(k) K_1 + \mu_2(k) K_2$$

$$x_{\text{est}}(k) = \mu_1(k) \hat{x}_1(k) + \mu_2(k) \hat{x}_2(k)$$



MAPS control automatically adjusts gains according to road condition changes for continuous performance

Compared to fixed-gain LQR, yaw rate (r) tracking RMSE reduced by over xx%, especially effective on icy road sections

Mode probabilities (μ) and estimated parameters (C_f , C_r) adapt in real-time to provide smooth gain scheduling

Maintains stable vehicle behavior without abrupt control input changes during road condition transitions