Adaptive Control – Mini-Project 1

Control Lyapunov Functions and Control Barrier Functions for Safety-Critical MIMO Nonlinear Systems

Instructor: Seyyed Ali Emami

Semester: Spring 2026 Due Date: 21 Ordibehesht 1405

Based on Chapters 2 and 3 of "Adaptive and Learning-Based Control of Safety-Critical Systems" by Max Cohen and Calin Belta

Abstract

This mini-project provides hands-on experience with the foundational concepts of safety-critical control. You will work with a validated multi-input multi-output (MIMO) nonlinear system, design Control Lyapunov Functions (CLFs) to guarantee stability, and formulate Control Barrier Functions (CBFs) to enforce safety constraints. Through a unified quadratic programming (QP) framework, you will synthesize controllers that simultaneously achieve stabilization, safety, and respect for input limits. The project emphasizes both theoretical derivation and practical implementation in MATLAB/Simulink.

1 Project Overview

This mini-project is designed to solidify your understanding of Chapters 2 and 3 from the Cohen and Belta text. The core tasks are:

1. Model and validate a MIMO nonlinear system from the control literature.
2. Design a Control Lyapunov Function (CLF) and formulate the corresponding minimum-norm stabilization QP.
3. Design a Control Barrier Function (CBF) and implement a safety filter that minimally modifies a nominal controller.
4. Unify CLF and CBF constraints within a single QP to achieve simultaneous stability and safety.
5. Incorporate input constraints (e.g., saturation limits) into all optimization formulations.
6. Analyze and discuss the trade-offs between stability, safety, and control effort through comprehensive simulation studies.

1.1 Recommended Reference Library

The **CBF-CLF-Helper** MATLAB library[1] provides an excellent foundation for implementing CLF/CBF controllers. You may use this library as a starting point, but you **must** demonstrate understanding of the underlying mathematics by explicitly deriving the CLF/CBF conditions in your report.

2 Project Tasks and Detailed Instructions

This project is structured into six sequential tasks. Each task builds upon the previous one.

2.1 Task 1: System Modeling and Validation

2.1.1 System Selection

Select a MIMO nonlinear system from the control literature that possesses the following characteristics:

- **Control-affine form**: $\dot{x} = f(x) + g(x)u$.
- **Multiple inputs** ($m \geq 2$) and multiple outputs ($p \geq 2$).
- **Established benchmark** with known parameters and validation in the literature.
- **Non-trivial nonlinear dynamics** that make linearization inadequate.

2.1.2 Model Documentation

For your chosen system, provide:

- **Dynamic equations** in control-affine form: $\dot{x} = f(x) + g(x)u$, clearly defining $f(x)$ and $g(x)$.
- **State vector** description and physical interpretation of each component.
- **Input vector** description and physical interpretation.
- **Output equation** $y = h(x)$ (if output tracking is considered).
- **Nominal parameter values** with proper units.
- **Equilibrium points** of interest.

[1] `https://github.com/hybridrobotics/cbf-clf-helper`

2.1.3 Model Validation

Demonstrate that your model is correctly implemented:

- **Open-loop simulation**: Apply a constant or sinusoidal input and verify that the system response matches physical intuition.
- **Simulink implementation**: Create a Simulink block diagram representing the nonlinear dynamics. The block should accept the input vector u and output the state derivative $\dot{x}$.

Deliverable for Task 1: Simulink model file (`System_Dynamics.slx`) and a section in your report documenting the system equations, parameters, and validation results.

2.2 Task 2: Control Lyapunov Function Design and Minimum-Norm Stabilization

2.2.1 CLF Construction

Design a Control Lyapunov Function (CLF) $V : \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ for your system. For a CLF, the following properties must hold:

1. **Positive definiteness**: $V(0) = 0$ and $V(x) > 0$ for all $x \neq 0$.
2. **Radial unboundedness**: $V(x) \to \infty$ as $\|x\| \to \infty$.
3. **Control Lyapunov inequality**: There exists a class $\mathcal{K}$ function α and a control input u such that:
$$\dot{V}(x,u) = \frac{\partial V}{\partial x}\big(f(x) + g(x)u\big) \leq -\alpha(V(x)) \tag{1}$$
for all x in the domain of interest.

Important: Explicitly derive $\frac{\partial V}{\partial x}$ and compute the Lie derivatives $L_f V(x)$ and $L_g V(x)$ in your report.

2.2.2 Minimum-Norm Stabilizing Controller

Formulate the **minimum-norm optimization problem** that finds the control input u closest to zero while satisfying the CLF constraint:

$$\begin{aligned} \min_{u \in \mathbb{R}^m} \quad & \|u\|^2 \\ \text{s.t.} \quad & L_f V(x) + L_g V(x) u \leq -\gamma V(x) \end{aligned} \tag{2}$$

where $\gamma > 0$ is a design parameter that controls the convergence rate (Note that this formulation is similar to those we have used in the class by putting $\alpha(\|x\|) = \gamma V(x)$, assuming that it satisfies the strictly increasing condition).

This problem is a **quadratic program (QP)** and can be written in standard form:

$$\min_{u} \quad u^T H u + f^T u \tag{3}$$

$$\text{s.t.} \quad Au \leq b \tag{4}$$

with $H = I_m$, $f = \mathbf{0}$, $A = L_g V(x)$, and $b = -L_f V(x) - \gamma V(x)$.

2.2.3 Implementation Requirements

- **MATLAB function**: Write a function `u_stab = CLF_Controller(x, gamma)` that computes the stabilizing control input using `quadprog`.
- **Simulink integration**: Embed this controller in a closed-loop Simulink model.
- **Simulation studies**: Demonstrate stabilization from various initial conditions and analyze the effect of varying γ.

2.2.4 Input Constraints (Task 2 Extension)

Incorporate actuator limits of the form $u_{\min} \leq u_i \leq u_{\max}$ into the QP:

$$\begin{aligned} \min_{u} \quad & u^T u \\ \text{s.t.} \quad & L_f V(x) + L_g V(x) u \leq -\gamma V(x) \\ & u_{\min} \leq u \leq u_{\max} \end{aligned} \tag{5}$$

Discuss cases where the QP becomes infeasible and propose potential remedies (e.g., relaxing the CLF constraint with a slack variable).

Deliverable for Task 2: MATLAB function `CLF_Controller.m`, Simulink model `CLF_ClosedLoop.slx`, and a report section with CLF derivation, QP formulation, and simulation results comparing unconstrained vs. input-constrained stabilization.

2.3 Task 3: Control Barrier Function Design and Safety Filter

2.3.1 Safety Specification

Define a safety set $\mathcal{C}$ as the 0-superlevel set of a continuously differentiable function $h : \mathbb{R}^n \to \mathbb{R}$:

$$\mathcal{C} = \{x \in \mathbb{R}^n : h(x) \geq 0\} \tag{6}$$

2.3.2 CBF Construction

A function h is a **Control Barrier Function (CBF)** on $\mathcal{C}$ if there exists an extended class $\mathcal{K}$ function α_B such that:

$$\sup_{u \in \mathbb{R}^m} \left[L_f h(x) + L_g h(x) u + \alpha_B(h(x)) \right] \geq 0 \tag{7}$$

for all $x \in \mathcal{C}$.

For implementation, we enforce the CBF constraint as:

$$L_f h(x) + L_g h(x) u \geq -\gamma_B h(x) \tag{8}$$

where we choose $\alpha_B(h) = \gamma_B h$ with $\gamma_B > 0$ (exponential CBF).

Higher-order considerations: If $L_g h(x) \equiv 0$ (the relative degree is greater than 1), you must use **High-Order CBFs (HOCBFs)**.

2.3.3 Safety Filter Implementation

Implement a **safety filter** that minimally modifies a nominal (potentially unsafe) controller u_{nom}:

$$\begin{aligned} \min_{u} \quad & \|u - u_{\text{nom}}\|^2 \\ \text{s.t.} \quad & L_f h(x) + L_g h(x) u \geq -\gamma_B h(x) \end{aligned} \tag{9}$$

- Use the CLF controller from Task 2 as the nominal controller u_{nom}.
- Simulate scenarios where the nominal controller would violate the safety constraint.
- Demonstrate that the safety filter successfully enforces the constraint while minimizing deviation.

2.3.4 Input Constraints (Task 3 Extension)

Incorporate actuator limits $u_{\min} \leq u \leq u_{\max}$ into the safety filter QP. Analyze situations where both safety and input constraints are active simultaneously.

Deliverable for Task 3: MATLAB functions `CBF_Constraint.m` (computes $L_f h$, $L_g h$), `Safety_Filter.m` (QP solver), Simulink model `CBF_SafetyFilter.slx`, and a report section with CBF derivation, safety filter formulation, and comparative simulation results.

2.4 Task 4: Unified CLF-CBF Quadratic Program

2.4.1 Joint Optimization Formulation

Combine the CLF stability constraint and CBF safety constraints into a single QP that solves for u directly:

$$\boxed{\begin{aligned} \min_{u,\delta} \quad & \|u\|^2 + p\delta^2 \\ \text{s.t.} \quad & L_f V(x) + L_g V(x) u \leq -\gamma V(x) + \delta \\ & L_f h_j(x) + L_g h_j(x) u \geq -\gamma_{B,j} h_j(x), \quad j = 1, \ldots, k \\ & u_{\min} \leq u \leq u_{\max} \\ & \delta \geq 0 \end{aligned}} \tag{10}$$

Explanation of terms:

- **Slack variable** δ: Allows temporary relaxation of the CLF constraint when safety requirements conflict with stabilization. This ensures the QP remains feasible.
- **Penalty parameter** $p > 0$: Weights the violation of the CLF constraint; larger p enforces stricter stabilization.

2.4.2 Implementation

Write a MATLAB function `CLF_CBF_Controller.m` that implements the unified QP. The function should:

1. Compute $V(x)$, $\frac{\partial V}{\partial x}$, $L_f V(x)$, and $L_g V(x)$.
2. Compute $h_j(x)$, $\frac{\partial h_j}{\partial x}$, $L_f h_j(x)$, and $L_g h_j(x)$ for each barrier.
3. Formulate and solve the QP using `quadprog`.
4. Return the optimal control input u^*.

2.4.3 Simulation Studies

Conduct the following experiments and analyze the results:

Table 1: Unified CLF-CBF Simulation Experiments

Experiment	**Description**	**What to Observe**
Nominal stabilization	Start from safe initial condition; no safety threats.	CLF drives state to zero; CBF constraints inactive.
Safety-critical scenario	Nominal trajectory would violate safety constraint.	CBF constraints become active; δ may become positive; safety maintained.
Input saturation	Demanded control exceeds actuator limits.	Input clipping occurs; analyze effect on stability and safety.
Infeasibility analysis	Conflict between CLF and CBF makes QP infeasible.	Propose and test remediation strategies.

Deliverable for Task 4: MATLAB function `CLF_CBF_Controller.m`, Simulink model `CLF_CBF_Unified.slx`, and a comprehensive report section analyzing the trade-offs between stability, safety, and control effort.

2.5 Task 5: Comprehensive Analysis and Discussion

Your report must include a thorough discussion addressing the following points:

Comparative Performance Analysis

- Compare the **unconstrained CLF controller** vs. **CLF with input constraints**.
- Compare the **safety filter** vs. **unified CLF-CBF QP**.
- Quantify performance using metrics: settling time, overshoot, control effort $\int \|u(t)\|^2 dt$, constraint violation magnitude.

Trade-off Discussion

- How does the choice of γ (CLF convergence rate) affect safety?
- How does γ_B (CBF decay rate) influence control effort and system responsiveness?
- What is the role of the slack penalty p? Provide simulation evidence showing the effect of $p \to \infty$ vs. $p \to 0$.

Infeasibility and Practical Considerations

- Identify conditions under which the CLF-CBF QP becomes infeasible.
- Discuss the theoretical guarantees (or lack thereof) when input constraints are active.
- Propose practical strategies for handling infeasibility (e.g., emergency stop, control allocation, constraint prioritization).

Validation and Limitations

- How well does your model capture the true system dynamics? Discuss unmodeled effects (friction, flexibility, sensor noise).
- What assumptions underlie the CLF-CBF framework (continuous-time, perfect state measurement) and how might they be violated in practice?

2.6 Task 6: Code Organization and Deliverables

2.6.1 Code Structure

Your submission must include a `Main.m` script that orchestrates all simulations. The script should:

- Initialize system parameters and simulation settings.
- Call the appropriate controller functions for each simulation scenario.
- Generate all plots required for the report.
- Be well-commented and self-contained.

Required file structure:

```
Main.m                    # Master script to run all simulations
System_Parameters.m       # System constants and model parameters
Dynamics.m                # Computes f(x) and g(x)
CLF_Controller.m          # Task 2: Minimum-norm CLF controller
CBF_Constraint.m          # Task 3: Computes CBF Lie derivatives
Safety_Filter.m           # Task 3: CBF safety filter QP
CLF_CBF_Controller.m      # Task 4: Unified CLF-CBF QP
```

```
Plot_Results.m              # Helper function for generating plots
System_Dynamics.slx         # Simulink model of open-loop system
CLF_ClosedLoop.slx          # Simulink model with CLF controller
CBF_SafetyFilter.slx        # Simulink model with safety filter
CLF_CBF_Unified.slx         # Simulink model with unified QP
README.txt                  # Instructions for running the code
```

2.6.2 Deliverables Summary

Table 2: Project Deliverables

Deliverable	**Format**	**Details**
Handwritten Report	PDF scan	Maximum 10 pages, single-sided, legible handwriting.
Video Presentation	MP4 file	7 minutes (strict), explaining code and key results.
MATLAB Code	`.m` and `.slx` files	All files listed above, zipped with folder structure.

3 Report Formatting Guidelines

Your handwritten report should be scanned to PDF. Use the following structure:

```
1. Introduction
2. System Modeling and Validation
   2.1 System Description
   2.2 Control-Affine Form
   2.3 Model Validation
3. Control Lyapunov Function Design
   3.1 CLF Construction
   3.2 Minimum-Norm QP Formulation
   3.3 Simulation Results (Unconstrained)
   3.4 Input-Constrained CLF Controller
4. Control Barrier Function Design
   4.1 Safety Specification
   4.2 CBF Construction and Relative Degree
   4.3 Safety Filter Implementation
   4.4 Simulation Results
5. Unified CLF-CBF Quadratic Program
   5.1 Joint Optimization Formulation
   5.2 Simulation Studies and Analysis
6. Discussion and Conclusions
```

```
   6.1 Comparative Analysis
   6.2 Trade-offs and Limitations
7. References
8. Appendix (optional)
```

4 References

1. Cohen, M., & Belta, C. (2023). *Adaptive and Learning-Based Control of Safety-Critical Systems*. Springer.

2. Ames, A. D., Xu, X., Grizzle, J. W., & Tabuada, P. (2017). Control barrier function based quadratic programs for safety critical systems. *IEEE Transactions on Automatic Control*, 62(8), 3861–3876.

3. Ames, A. D., Coogan, S., Egerstedt, M., Notomista, G., Sreenath, K., & Tabuada, P. (2019). Control barrier functions: Theory and applications. *2019 18th European Control Conference (ECC)*, 3420–3431.

4. MathWorks Documentation: Control Barrier Function (CBF) and Control Lyapunov Function (CLF) Based Control Methods.

5. CBF-CLF-Helper: MATLAB Interface for Control Barrier Function and Control Lyapunov Function Based Control Methods. GitHub Repository. `https://github.com/hybridrobotics/cbf-clf-helper`