Subsystem Safety Filtering: A Unified Framework for Safe Shared Control in Coupled Robotic Systems

Federico Pizarro Bejarano[1], Lukas Brunke[1,2], and Angela P. Schoellig[1,2]

***Abstract*—We introduce subsystem safety filters (SBSFs), a framework for safe shared control in coupled robotic systems. We consider mutli- and single-agent robots that can be divided into separately controllable subsystems, such as a drone swarm or a mobile manipulator divided into a mobile base and a manipulator arm. Our SBSF guarantees safety for the externally-controlled subsystem while simultaneously controlling the other subsystem. Our results on a drone swarm show that our subsystem filtering significantly improves safety and performance over decentralized alternatives.**

I. Introduction

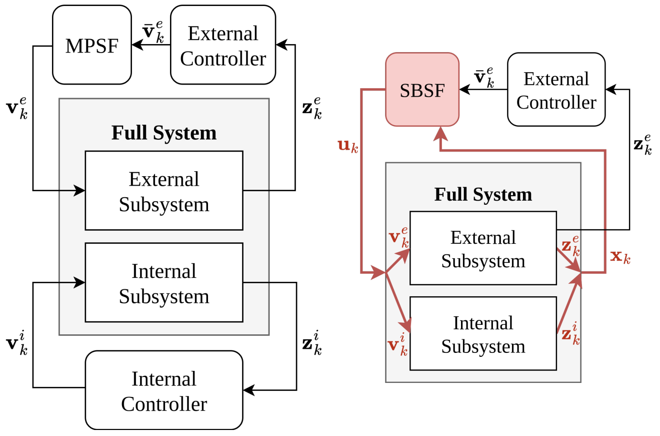
Robotic systems often consist of multiple tightly coupled subsystems, such as the arm and base of a mobile manipulator. In many applications, controlling only a subset of the system is necessary to reduce complexity and enable teleoperation. However, when the subsystems are coupled, the uncontrolled subsystems must be simultaneously controlled to ensure safety.

We propose subsystem safety filtering (SBSF), a control paradigm in which a finite-time optimal control optimization problem ensures safety across the entire system, while minimally correcting the control inputs applied to the externally-controlled subsystem. For example, consider a mobile manipulator carrying objects on a tray while moving through a cluttered environment. The base may be teleoperated to move through the environment, safety filtered by the SBSF, while the SBSF adjusts the arm to keep the tray it is carrying level.

II. Related Work

A. Safety Filters

Safety filters guarantee the safety of a system controlled by an arbitrary policy (e.g., an RL agent or teleoperator) by minimally modifying the commands to enforce state and input constraints. Control barrier function (CBF) safety filters enforce forward-invariance of safe sets [1], but these can induce chattering near constraint boundaries [2]. Model predictive safety filters (MPSFs) instead use model predictive control (MPC) theory to guarantee safe backup trajectories to a terminal set [3]. Recent work has proposed multi-step model predictive safety filters, which optimize over a short horizon to smooth interventions and reduce chattering [4].



(a) Naive safety filtering approach (b) Proposed subsystem filter (SBSF)

Fig. 1: The naive safety filtering approach compared to our proposed approach, with changes in red.

These safety filters assume the controller commands the entire robotic system. As a result, they are not tailored to scenarios in which only a subset of the robot's actuators (or degrees of freedom) are directly commanded. This occurs when control is shared between an internal safety mechanism and the external controller. Our work extends model predictive safety filters to this task, restricting external control to a portion of the system while controlling the remaining subsystems, and maintaining safety guarantees throughout.

B. Shared Control

Shared control arises in human–robot interaction (HRI) when a human operator and an autonomous agent jointly generate commands. Unlike the safety-oriented filter literature, shared-control research often emphasizes task performance, ergonomics, or intent inference over formal guarantees. Two main paradigms have developed:

- **Blended Control:** The inputs from the human and the autonomous agent are fused — for example, via weighted averaging or by solving a receding-horizon MPC that stays close to the human's suggestion while respecting dynamics and environment constraints [5]. However, the blended commands may not satisfy the goals of either the human or autonomous agents, and results in less performant policies.
- **Authority Switching:** Frameworks such as EMICS [6] or HierEMICS [7] dynamically allocate control authority between a human and a robot based on the estimated operator state or the environment. While they can im-

[1] The authors are with the Learning Systems and Robotics Lab (www.learnsyslab.org), University of Toronto, Canada, and affiliated with the University of Toronto Robotics Institute and the Vector Institute for Artificial Intelligence in Toronto. {federico.pizarrobejarano, lukas.brunke, angela.schoellig}@robotics.utias.utoronto.ca

[2] Lukas Brunke and Angela P. Schoellig are also with the Technical University of Munich and the Munich Institute for Robotics and Machine Intelligence (MIRMI).

prove usability and reduce conflict, they rely on heuristic switching logic.

In contrast, our approach decouples the robot system into two groups of subsystems — an externally controlled group that is safety filtered and an internally controlled group — while enforcing safety jointly via a unified optimization problem, the subsystem safety filter (SBSF).

III. Problem Setup

We consider a discrete, time-invariant system given by:

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k), \tag{1}$$

where $\mathbf{x}_k \in \mathbb{X} \subset \mathbb{R}^n$ represents the system state at time step k, $\mathbf{u}_k \in \mathbb{U} \subset \mathbb{R}^m$ denotes the control input, and $\mathbf{f}$ encapsulates the system dynamics. The system is subject to known state and input constraints $\mathbf{x} \in \mathbb{X}_\mathrm{c}$, $\mathbf{u} \in \mathbb{U}_\mathrm{c}$.

We decompose the system into two subsystems, which we denote as an external subsystem and an internal subsystem. Let us define projection functions:

$$\begin{bmatrix} \mathbf{z}^\mathrm{i}_k \\ \mathbf{z}^\mathrm{e}_k \end{bmatrix} = \begin{bmatrix} \mathbb{T}^\mathrm{i}_x(\mathbf{x}_k) \\ \mathbb{T}^\mathrm{e}_x(\mathbf{x}_k) \end{bmatrix} = \mathbb{T}_x(\mathbf{x}_k), \tag{2}$$

$$\mathbf{u}_k = \mathbb{T}_v(\mathbf{v}^\mathrm{i}_k, \mathbf{v}^\mathrm{e}_k), \tag{3}$$

where $\mathbf{z}^\mathrm{i}_k \in \mathbb{R}^{n^\mathrm{i}}$ and $\mathbf{z}^\mathrm{e}_k \in \mathbb{R}^{n^\mathrm{e}}$ are the internal and external subsystem states respectively, $\mathbf{v}^\mathrm{i}_k \in \mathbb{R}^{m^\mathrm{i}}$ and $\mathbf{v}^\mathrm{e}_k \in \mathbb{R}^{m^\mathrm{e}}$ are the inputs to those subsystems, $\mathbb{T}_x$ is a transformation function which projects the full system state $\mathbf{x}_k$ to the two subsystems, and $\mathbb{T}_v$ transforms the subsystem inputs to the full system input $\mathbf{u}_k$.

Constraints are imposed separately on each subsystem via

$$\mathbf{z}^\mathrm{i}_k \in \mathbb{Z}^\mathrm{i}_c, \;\; \mathbf{z}^\mathrm{e}_k \in \mathbb{Z}^\mathrm{e}_c, \quad \text{and} \quad \mathbf{v}^\mathrm{i}_k \in \mathbb{V}^\mathrm{i}_c, \;\; \mathbf{v}^\mathrm{e}_k \in \mathbb{V}^\mathrm{e}_c. \tag{4}$$

These constraints are equivalent to the full system constraints.

Thus, we can write the subsystem dynamics as:

$$\mathbf{z}^\mathrm{i}_{k+1} = \mathbb{T}^\mathrm{i}_x(\mathbf{f}(\mathbf{x}_k, \mathbb{T}_v(\mathbf{v}^\mathrm{i}_k, \mathbf{v}^\mathrm{e}_k))) \tag{5}$$

$$\mathbf{z}^\mathrm{e}_{k+1} = \mathbb{T}^\mathrm{e}_x(\mathbf{f}(\mathbf{x}_k, \mathbb{T}_v(\mathbf{v}^\mathrm{i}_k, \mathbf{v}^\mathrm{e}_k))). \tag{6}$$

In this paper, we focus on subsystems which are *coupled*, thus each subsystem's dynamics depend on the state and/or input from the other subsystem.

$$\begin{bmatrix} \mathbf{z}^\mathrm{i}_{k+1} \\ \mathbf{z}^\mathrm{e}_{k+1} \end{bmatrix} = \begin{bmatrix} \mathbf{f}^\mathrm{i}(\mathbf{z}^\mathrm{i}_k, \mathbf{z}^\mathrm{e}_k, \mathbf{v}^\mathrm{i}_k, \mathbf{v}^\mathrm{e}_k) \\ \mathbf{f}^\mathrm{e}(\mathbf{z}^\mathrm{i}_k, \mathbf{z}^\mathrm{e}_k, \mathbf{v}^\mathrm{i}_k, \mathbf{v}^\mathrm{e}_k) \end{bmatrix} \tag{7}$$

$$\neq \begin{bmatrix} \mathbf{f}^\mathrm{i}(\mathbf{z}^\mathrm{i}_k, \mathbf{v}^\mathrm{i}_k) \\ \mathbf{f}^\mathrm{e}(\mathbf{z}^\mathrm{e}_k, \mathbf{v}^\mathrm{e}_k) \end{bmatrix} \tag{8}$$

A simplification of the projections above is to consider linearly separable subsystems:

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{z}^\mathrm{i}_k \\ \mathbf{z}^\mathrm{e}_k \end{bmatrix}, \; \mathbf{u}_k = \begin{bmatrix} \mathbf{v}^\mathrm{i}_k \\ \mathbf{v}^\mathrm{e}_k \end{bmatrix}, \tag{9}$$

$$\mathbb{X}_c = \mathbb{Z}^\mathrm{i}_c \times \mathbb{Z}^\mathrm{e}_c, \quad \mathbb{U}_c = \mathbb{V}^\mathrm{i}_c \times \mathbb{V}^\mathrm{e}_c. \tag{10}$$

This simplification is sufficient for most cases, such as a mobile manipulator where the mobile base and the manipulator arm are the two subsystems.

Our goal is to design a subsystem safety filter that:

1) Accepts the command for the external subsystem $\bar{\mathbf{v}}^\mathrm{e}_k$.
2) Computes $(\mathbf{v}^\mathrm{i}_k, \mathbf{v}^\mathrm{e}_k)$, ensuring safety.
3) Minimizes $\|\bar{\mathbf{v}}^\mathrm{e}_k - \mathbf{v}^\mathrm{e}_k\|$ in an appropriate norm.
4) Commands the internal subsystem to complete a task, such as stabilization or trajectory tracking.

IV. Subsystem Model Predictive Shared Control

Let us consider a standard MPC optimization:

$$\min_{\mathbf{u}_{\cdot|k}} \; J_{\text{MPC}}(\bar{\mathbf{x}}_{\cdot|k}, \bar{\mathbf{u}}_{\cdot|k}, \mathbf{Q}_\cdot, \mathbf{R}_\cdot) \tag{11a}$$

$$\text{s.t. constraints satisfied,} \tag{11b}$$

where $\mathbf{u}_{j|k}$ are the inputs at the $(k+j)$-th time step computed at time step k, $\bar{\mathbf{x}}_{j|k}$ and $\bar{\mathbf{u}}_{j|k}$ are the desired states and inputs at the $(k+j)$-th time step, and $J_{\text{MPC}}(\bar{\mathbf{x}}_{\cdot|k}, \bar{\mathbf{u}}_{\cdot|k}, \mathbf{Q}_\cdot, \mathbf{R}_\cdot) = \sum_{j=0}^{H-1} \|\bar{\mathbf{x}}_{j|k} - \mathbf{x}_{j|k}\|^2_{\mathbf{Q}_j} + \|\bar{\mathbf{u}}_{j|k} - \mathbf{u}_{j|k}\|^2_{\mathbf{R}_j}$, where $\mathbf{Q}_j \in \mathbb{R}^{n\times n}$ and $\mathbf{R}_j \in \mathbb{R}^{m\times m}$ are the weight matrices for the states and inputs, and $H \in \mathbb{Z}_{>0}$ is the MPC horizon.

MPSFs are standard MPCs with no state tracking ($\mathbf{Q}_j = \mathbf{0}_n \;\; \forall j \geq 0$), and with the desired input trajectory $\bar{\mathbf{u}}_{\cdot|k}$ set to the future commands of the external controller. We denote the MPSF objective as $J_{\text{MPSF}}(\bar{\mathbf{u}}_{\cdot|k}, \mathbf{R}_\cdot) = \sum_{j=0}^{H-1} \|\bar{\mathbf{u}}_{j|k} - \mathbf{u}_{j|k}\|^2_{\mathbf{R}_j}$. In the standard one-step MPSF [3] $\mathbf{R}_0 = \mathbf{1}_m$ and $\mathbf{R}_j = \mathbf{0}_m \;\; \forall j > 0$. In multi-step MPSFs, the future inputs of the external controller $\bar{\mathbf{u}}_{j|k}, \;\; \forall j > 0$ are estimated using [4].

For our approach, we apply the MPC objective function to the internally controlled subsystem and the MPSF objective function to the externally controlled subsystem:

$$\begin{aligned} &J_{\text{SBSF}}(\bar{\mathbf{x}}_{\cdot|k}, \bar{\mathbf{u}}_{\cdot|k}, \mathbf{Q}^\mathrm{i}_\cdot, \mathbf{R}^\mathrm{e}_\cdot, \mathbf{R}^\mathrm{i}_\cdot) = \\ &J_{\text{MPSF}}(\bar{\mathbf{v}}^\mathrm{e}_{\cdot|k}, \mathbf{R}^\mathrm{e}_\cdot) + J_{\text{MPC}}(\bar{\mathbf{z}}^\mathrm{i}_{\cdot|k}, \bar{\mathbf{v}}^\mathrm{i}_{\cdot|k}, \mathbf{Q}^\mathrm{i}_\cdot, \mathbf{R}^\mathrm{i}_\cdot). \end{aligned} \tag{12}$$

By changing the weight matrices $\mathbf{R}^\mathrm{e}_j$, $\mathbf{Q}^\mathrm{i}_j$, and $\mathbf{R}^\mathrm{i}_j$, the system will prioritize different behaviours.

V. Experiments on Drone Swarm

We ran experiments on a swarm of four Crazyflie 2.0 drones to test our subsystem filtering approach. The experiments were conducted in simulation using the massively parallelizable multi-drone simulator Crazyflow [8]. Two of these drones were considered the external subsystem and were controlled by individual linear quadratic regulators (LQRs) representing teleoperators. The other two drones represented the internal subsystem. This experiment represents human teleoperators flying drones within a larger swarm, demonstrating that our approach can ensure safety and maintain performance for all drones.

The desired trajectories for the drones are aggressive 3D trajectories that overlap, leading to collisions if the trajectories are followed precisely. The drones were constrained to be within a box that tightly wraps the desired trajectory, leading to constraint violations if the drones overshoot the trajectory.

A. Approaches

We compare our approach to naive and decentralized approaches to illustrate the importance of jointly filtering the external subsystem and controlling the internal subsystem.

1) No Filtering: Firstly, we ran the externally-controlled drones with no safety filtering, and the internally-controlled drones with a separate centralized MPC controller. The LQRs and the MPC only account for their respective subsystem, and are unaware of the other subsystem. As LQRs do not guarantee safety, the externally-controlled drones violate position constraints and collide with one another. The internally-controlled drones respect the constraints and do not collide with one another. However, due to a lack of communication between the two subsystems, the internally-controlled drones collide with the externally-controlled drones.

2) Naive Filtering: We added an MPSF to the externally-controlled drones, eliminating constraint violations and collisions with one another. However, the two subsystems still do not know of one another and collide.

3) Safe-External: We extend the MPSF on the external subsystem to be aware of the internal subsystem's current state and predicted trajectory. The internal subsystem remains unaware of the external subsystem. Thus, the external subsystem can now try to avoid collisions with the internally-controlled drones. However, since the MPSF cannot control the internal subsystem directly, this approach leads to high-magnitude corrections.

4) Safe-Internal: Similar to *Safe-Exteration*, we extend *Naive Filtering* by making the internal MPC aware of the current state and predicted trajectory of the external subsystem. The external subsystem remains unaware of the internal subsystem. This approach degrades the performance of the internally-controlled drones as they must now dodge the externally-controlled drones.

5) Subsystem Filtering: Our approach combines the internal subsystems MPC and the external subsystem MPSF into one optimization problem that jointly considers and commands all the drones at once.

B. Metrics

1) Performance: We measure the RMSE of both subsystems separately to track how well each approach tracks the desired trajectories of each subsystem.

2) Corrections: The safety filter attempts to minimize corrections to the external controller's desired actions. Thus, we measure the mean correction applied to the external commands [4]. Additionally, safety filters may cause chattering, which we aim to minimize. Thus, we also measure the rate of change of the inputs [4] for all approaches to determine the additional chattering caused by the approaches.

3) Safety: Finally, our approach is mainly concerned with safety. Thus, we measure the number of position constraint violations and collisions, defined as when the drones are within 15cm of each other.

C. Results

Our experiments are summarized in Table I. In all metrics except safety, the approaches with no filtering (*None*) and no communication or knowledge between the subsystems (*Naive*) outperform the safer approaches. This is to be expected, as enforcing safety necessarily requires deviations from the desired trajectory and greater corrections to the external subsystem. While both approaches cause many collisions, the *Naive* approach minimizes constraint violations.

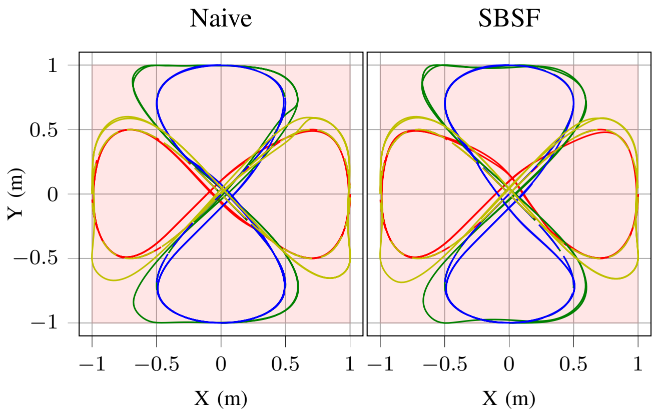


Fig. 2: The trajectories of a four-drone swarm. The yellow and green lines represent the externally-controlled drones, while the red and blue lines represent the internally-controlled drones. The pink square represents the position constraints. The *Naive* approach leads to a high number of collisions, while our SBSF approach coordinates between the two subsystems and minimizes collisions (see Table I).

Safe-Ext and *Safe-Int* have different trade-offs. *Safe-Ext* raises RMSE-Ext (the RMSE of the external drones) and Mean Corrections, while *Safe-Int* worsens RMSE-Int (the RMSE of the internal drones). In those three metrics, our approach (*SBSF*) is always between the two other approaches, although it is significantly closer to the better end of each metric. In the Rate of Change (RoC) of the Inputs and the Constraint Violations, our approach also sits between the other approaches. Our approach results in the minimum number of collisions.

While the *Safe-Ext* and *Safe-Int* approaches can theoretically enforce safety, their inability to affect the other subsystem reduces their effectiveness and results in significantly worsened performance in their own subsystem, as well as a reduced ability to avoid collisions. Our proposed subsystem filtering approach *SBSF* allows for simultaneously controlling both subsystems, balancing the RMSE in both subsystems as well as the corrections. The priority between the two subsystems can be chosen by changing the ratio between the safety filter cost J_{MPSF} and the MPC cost J_{MPC} in the SBSF objective function (Eq. 12). This flexibility also results in far more manoeuvrability, reducing collisions and improving safety.

TABLE I: Results of experiments comparing various approaches in controlling and filtering a swarm of four drones.

	None	Naive	Safe-Ext	Safe-Int	SBSF
RMSE-Ext (m)	0.119	0.162	0.277	0.162	0.177
RMSE-Int (m)	0.042	0.042	0.042	0.088	0.065
Mean Corrections	-	0.658	1.84	0.658	0.859
RoC of the Inputs	18.0	46.1	44.6	65.3	55.5
Constraint Viols	154	6	13	6	11
Collisions	568	332	36	24	4

References

[1] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *European Control Conference*, 2019.

[2] L. Brunke, S. Zhou, and A. P. Schoellig, "Preventing inactive cbf safety filters caused by invalid relative degree assumptions," *IEEE Transactions on Automatic Control*, 2025.

[3] K. P. Wabersich and M. N. Zeilinger, "Linear model predictive safety certification for learning-based control," in *IEEE Conference on Decision and Control*, 2018.

[4] F. Pizarro Bejarano, L. Brunke, and A. P. Schoellig, "Multi-step model predictive safety filters: Reducing chattering by increasing the prediction horizon," in *IEEE Conference on Decision and Control*, 2023.

[5] E. Jabbour, M. Vulliez, C. Preault, and V. Padois, "A model predictive control approach to blending in shared control," 2024.

[6] M. Chiou, N. Hawes, and R. Stolkin, "Mixed-initiative variable autonomy for remotely operated mobile robots," *ACM Transactions on Human-Robot Interaction*, 2021.

[7] D. Panagopoulos, G. Petousakis, A. Ramesh, T. Ruan, G. Nikolaou, R. Stolkin, and M. Chiou, "A hierarchical variable autonomy mixed-initiative framework for human-robot teaming in mobile robotics," in *2022 IEEE 3rd International Conference on Human-Machine Systems (ICHMS)*, 2022.

[8] M. Schuck and M. Rath. (2025) Crazyflow. [Online]. Available: https://github.com/utiasDSL/crazyflow