

Risk-Aware Safety Filters with Poisson Safety Functions and Laplace Guidance Fields

Gilbert Bahati, Ryan M. Bena, Meg Wilkinson, Pol Mestres, Ryan K. Cosner, and Aaron D. Ames

Abstract—Robotic systems navigating in real-world settings require a semantic understanding of their environment to properly determine safe actions. This work aims to develop the mathematical underpinnings of such a representation—specifically, the goal is to develop safety filters that are risk-aware. To this end, we take a two step approach: encoding an understanding of the environment via Poisson’s equation, and associated risk via Laplace guidance fields. That is, we first solve a Dirichlet problem for Poisson’s equation to generate a safety function that encodes system safety as its 0-superlevel set. We then separately solve a Dirichlet problem for Laplace’s equation to synthesize a safe *guidance field* that encodes variable levels of caution around obstacles—by enforcing a tunable flux boundary condition. The safety function and guidance fields are then combined to define a safety constraint and used to synthesize a risk-aware safety filter which, given a semantic understanding of an environment with associated risk levels of environmental features, guarantees safety while prioritizing avoidance of higher risk obstacles. We demonstrate this method in simulation and discuss how *a priori* understandings of obstacle risk can be directly incorporated into the safety filter to generate safe behaviors that are risk-aware.

I. INTRODUCTION

As modern robots increasingly venture into the real world, they encounter obstacles with variable degrees of relevance to system safety. Different obstacle are often associated with different levels of risk, motivating different degrees of conservatism during navigation. For example, while avoiding collisions in the environment, it is often important for the robot to behave more cautiously around high-risk obstacles like humans or expensive equipment. A *risk-aware* safety approach ensures collision avoidance while incorporating additional conservatism near high-risk obstacles.

One common approach for encoding system safety is through control barrier functions (CBFs)[1], [2]. CBFs divide the system’s operable region into safe states and unsafe states and can be used to synthesize safe controllers by enforcing the forward invariance of the safe region. However, this binary representation of safety, especially in the context of collision avoidance, often treats all possible collisions equally by imposing uniform gradients on the boundary of the safe region [3], [4]. This uniformity limits the ability to tailor the system behavior to specific obstacles or spatially-dependent risk factors.

This research is supported by BP.

G. Bahati, R. M. Bena, P. Mestres and A. Ames are with the Department of Mechanical and Civil Engineering, Caltech, Pasadena, CA; M. Wilkinson is with the Department of Computing and Mathematical Sciences, Caltech, Pasadena, CA. Emails: {gbahati, ryanbena, mwilkinsmestres, ames}@caltech.edu.

R. K. Cosner is with the Department of Mechanical Engineering, Tufts University, Medford, MA, ryan.cosner@tufts.edu.

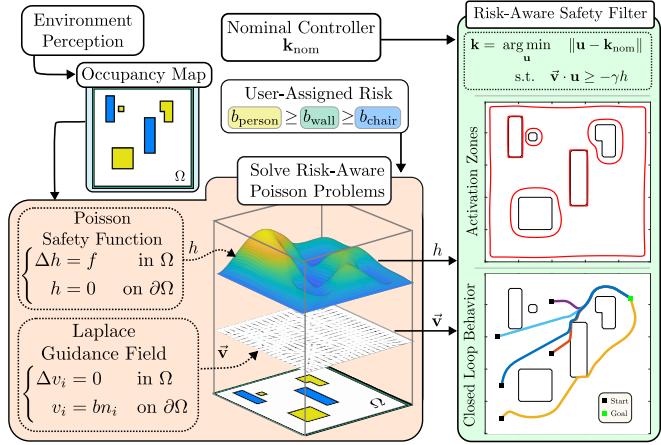


Fig. 1: The proposed risk-aware control synthesis method. By incorporating user-assigned risk values in the guidance field, our approach generates controllers and closed-loop behaviors exhibiting conservatism aligned with the prescribed risk levels.

While CBFs have been shown to be powerful theoretical tools for defining and enforcing safety, they are often difficult to generate, even for systems with simple dynamics. For complex environments, CBFs are often constructed *ad hoc* with limited generalizability [5]–[7]. While some reachability and learning approaches have shown promise as generalizable methods for automatic CBF synthesis [3], [8]–[10], they are computationally complex, and typically require that the CBF be computed offline, preventing actively incorporation of dynamic environment data. Alternatively, Poisson safety functions (PSFs) [11] enable real-time generation of safe sets from perception data by solving a Dirichlet problem for Poisson’s equation online. This approach leverages a *guidance field* that encodes gradient information required for safety [12]. The guidance field provides additional flexibility in defining safety by allowing boundary conditions to specify desired gradients (i.e., boundary flux) on obstacle surfaces.

Inspired by the ability to assign the desired boundary flux for PSFs, this work presents a method for generating safe behavior with variable, spatially-dependent, user-assigned conservatism by directly assigning boundary flux values through *Laplace* guidance fields and decoupling the safety gradient from the safety value in the standard CBF safety filter [13]. Specifically, the safety gradient, i.e., the vector field yielding system safety, is now directly defined by the guidance field and is not necessarily the exact gradient of the PSF. While other methods such as tunable input-to-state safe CBFs [14] and state-dependent CBF decay conditions [15] also enable spatially variable degrees of conservatism, neither are capable of rapidly generating CBFs

with variable conservatism online. Alternatively, our method is constructive, capable of online synthesis, and results in behaviors which display different levels of caution depending around different obstacles depending on the assigned risk—remaining agnostic to what the risk represents and how it is quantified [16], thus is broadly applicable to different scenarios and objectives.

The main contributions of this work are threefold. First, we develop a novel method for automatically synthesizing PSF-based safety filters with Laplace guidance fields to incorporate *a priori* risk-awareness, achieving spatially variable conservatism. Next, we provide analysis on how adjustments of the guidance field affect the *activation zones* of the safety filter, directly displaying how modifications to the flux impact the conservatism of the filter. Finally, we provide several examples demonstrating how our method can be used to achieve risk-aware safety in a variety of contexts such as environments with probabilistic occupancies, dynamic obstacles or semantic labels with risk magnitudes.

II. BACKGROUND

Consider the nonlinear control affine system of the form:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u}, \quad (1)$$

where $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{u} \in \mathbb{R}^m$ are the state and input, and $\mathbf{f} : \mathbb{R}^n \rightarrow \mathbb{R}^n$, $\mathbf{g} : \mathbb{R}^n \rightarrow \mathbb{R}^{n \times m}$ are assumed to be locally Lipschitz continuous functions. Given a locally Lipschitz continuous controller $\mathbf{k} : \mathbb{R}^n \rightarrow \mathbb{R}^m$, the closed-loop system $\dot{\mathbf{x}} = \mathbf{f}_{\text{cl}}(\mathbf{x}) = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{k}(\mathbf{x})$ has a unique solution for any initial condition $\mathbf{x}_0 \in \mathbb{R}^n$. For all closed-loop systems considered in this work, we assume that such solutions exist for all $t \geq 0$ for ease of exposition.

A. Safety and Control Barrier Functions

We formalize the definition of safety as the forward invariance of a *safe set*, where the system is considered safe when all trajectories of the closed-loop system remain in the desired safe set for all $t \geq 0$. In particular, we consider safe sets defined as the 0-superlevel set of a continuously differentiable function $h_{\mathcal{S}} : \mathbb{R}^n \rightarrow \mathbb{R}$ as:

$$\mathcal{S} = \{\mathbf{x} \in \mathbb{R}^n \mid h_{\mathcal{S}}(\mathbf{x}) \geq 0\}.$$

Control Barrier Functions (CBFs) are a constructive tool that can be used to design controllers for (1) that enforce the forward invariance of the set \mathcal{S} .

Definition 1. (Control Barrier Functions [1]) We call a function $h_{\mathcal{S}} : \mathbb{R}^n \rightarrow \mathbb{R}$ a Control Barrier Function (CBF) for (1) if there exists¹ $\gamma \in \mathcal{K}_{\infty}^e$ such that for all $\mathbf{x} \in \mathbb{R}^n$, the following condition holds:

$$\sup_{\mathbf{u} \in \mathbb{R}^m} \left\{ \underbrace{Dh_{\mathcal{S}}(\mathbf{x}) \cdot \mathbf{f}(\mathbf{x})}_{L_{\mathbf{f}}h_{\mathcal{S}}(\mathbf{x})} + \underbrace{Dh_{\mathcal{S}}(\mathbf{x}) \cdot \mathbf{g}(\mathbf{x})}_{L_{\mathbf{g}}h_{\mathcal{S}}(\mathbf{x})} \mathbf{u} \right\} > -\gamma(h_{\mathcal{S}}(\mathbf{x})), \quad (3)$$

where $Dh_{\mathcal{S}}$ denotes the gradient of $h_{\mathcal{S}}$.

¹A continuous function $\gamma : \mathbb{R} \rightarrow \mathbb{R}$ is an *extended class K*, denoted by $\gamma \in \mathcal{K}_{\infty}^e$, if γ is monotonically increasing, $\gamma(0) = 0$, $\lim_{s \rightarrow \infty} \gamma(s) = \infty$, and $\lim_{s \rightarrow -\infty} \gamma(s) = -\infty$.

Given a nominal controller $\mathbf{k}_{\text{nom}} : \mathbb{R}^n \rightarrow \mathbb{R}^m$, and a CBF $h_{\mathcal{S}}$, a typical way of synthesizing safe controllers is through quadratic programming-based safety filters, which adjust \mathbf{k}_{nom} to the nearest safe action:

$$\begin{aligned} \mathbf{k}(\mathbf{x}) = \arg \min_{\mathbf{u} \in \mathbb{R}^m} \|\mathbf{u} - \mathbf{k}_{\text{nom}}(\mathbf{x})\|_2^2 \\ \text{s.t.} \quad L_{\mathbf{f}}h_{\mathcal{S}}(\mathbf{x}) + L_{\mathbf{g}}h_{\mathcal{S}}(\mathbf{x})\mathbf{u} \geq -\gamma(h_{\mathcal{S}}(\mathbf{x})). \end{aligned} \quad (\text{Safety-Filter})$$

B. Outputs and Relative Degree

This manuscript considers safety specifications which can be represented via a set of desired *outputs*. We recall the notion of *relative degree*, which describes the level of differentiation at which a control input affects an output.

Definition 2 (Relative Degree r [17]). A function $\mathbf{y} : \mathbb{R}^n \rightarrow \mathbb{R}^p$ has *relative degree* $r \in \mathbb{N}$ for (1) if:

$$L_{\mathbf{g}}L_{\mathbf{f}}^i \mathbf{y}(\mathbf{x}) \equiv \mathbf{0}, \quad \forall i \in \{0, \dots, r-2\}, \quad (4)$$

$$\text{rank}(L_{\mathbf{g}}L_{\mathbf{f}}^{r-1} \mathbf{y}(\mathbf{x})) = p, \quad \forall \mathbf{x} \in \mathbb{R}^n. \quad (5)$$

Given an output \mathbf{y} with relative degree r , we define a new set of partial coordinates:

$$\vec{\mathbf{y}}(\mathbf{x}) := \begin{bmatrix} \mathbf{y}(\mathbf{x}) \\ \mathbf{y}^{(1)}(\mathbf{x}) \\ \vdots \\ \mathbf{y}^{(r-1)}(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} \mathbf{y}(\mathbf{x}) \\ L_{\mathbf{f}}\mathbf{y}(\mathbf{x}) \\ \vdots \\ L_{\mathbf{f}}^{r-1}\mathbf{y}(\mathbf{x}) \end{bmatrix} \in \mathbb{R}^{pr}, \quad (6)$$

where $\mathbf{y}^{(r)} = \frac{d^r \mathbf{y}}{dt^r}$, leading to the following linear dynamics:

$$\frac{d}{dt} \vec{\mathbf{y}}(\mathbf{x}) = \underbrace{\begin{bmatrix} \mathbf{0} & \mathbf{I}_{p(r-1)} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}}_{\mathbf{A}} \vec{\mathbf{y}}(\mathbf{x}) + \underbrace{\begin{bmatrix} \mathbf{0} \\ \mathbf{I}_p \end{bmatrix}}_{\mathbf{B}} \mathbf{w} \quad (7)$$

$$\mathbf{w} := L_{\mathbf{f}}^r \mathbf{y}(\mathbf{x}) + L_{\mathbf{g}}L_{\mathbf{f}}^{r-1} \mathbf{y}(\mathbf{x})\mathbf{u}, \quad (8)$$

where (8) is an input to (7). When \mathbf{y} has relative degree r , the controller $\mathbf{w} = \hat{\mathbf{k}}(\vec{\mathbf{y}})$ designed for (7) can be transferred back to the controller for (1) as follows²:

$$\mathbf{u} = L_{\mathbf{g}}L_{\mathbf{f}}^{r-1} \mathbf{y}(\mathbf{x})^\dagger \left[\hat{\mathbf{k}}(\vec{\mathbf{y}}(\mathbf{x})) - L_{\mathbf{f}}^r \mathbf{y}(\mathbf{x}) \right]. \quad (9)$$

This partial coordinate transformation is a full coordinate transformation if $pr = n$. The assumption that \mathbf{y} has relative degree r , and the form of the output dynamics (7), enables the employment of various CBF construction methods for the original dynamical system (1) [18]–[21]. Next, we discuss a method of synthesizing CBFs for environmentally relevant safety specifications, as presented in [11].

C. Poisson Safety Functions

We focus on systems for which safety specifications are described in spatial coordinates $\mathbf{y} = (x, y, z) \in \mathbb{R}^3$. Given environmental occupancy data, let Ω be a smooth, open, bounded and connected set representing unoccupied regions and $\partial\Omega$ represent the surfaces of occupied regions. Specifically, $\partial\Omega = \bigcup_i^n \partial\Gamma_i$ where Γ_i is an open, bounded and connected set corresponding to the interior of an occupied

²Condition (5) implies the right psuedo-inverse $L_{\mathbf{g}}L_{\mathbf{f}}^{r-1} \mathbf{y}(\mathbf{x})^\dagger$ exists.

region with n_o denoting the total number of occupied regions. A safety function provides a functional representation of safety for an environment, defined as follows.

Definition 3. (Safety Function [11]) Let $\mathbf{y} = (x, y, z) \in \mathbb{R}^3$ represent coordinates in three dimensional space. We call a function $h : \bar{\Omega} \rightarrow \mathbb{R}$ a safety function of order k on $\bar{\Omega}$ if h is k -times differentiable, $Dh(\mathbf{y}) \neq 0$ when $h(\mathbf{y}) = 0$, and the 0-superlevel set of h characterizes a safe set:

$$\mathcal{C} = \{\mathbf{y} \in \bar{\Omega} : h(\mathbf{y}) \geq 0\}, \quad (10a)$$

$$\partial\mathcal{C} = \{\mathbf{y} \in \bar{\Omega} : h(\mathbf{y}) = 0\}, \quad (10b)$$

$$\text{Int}(\mathcal{C}) = \{\mathbf{y} \in \bar{\Omega} : h(\mathbf{y}) > 0\}. \quad (10c)$$

Given environmental data characterizing the domain $\bar{\Omega}$ through an occupancy map, Poisson safety functions [11] generate safe sets satisfying Def. 3 by solving a Dirichlet problem for Poisson's equation:

$$\begin{cases} \Delta h(\mathbf{y}) = f(\mathbf{y}) & \text{in } \Omega, \\ h(\mathbf{y}) = 0 & \text{on } \partial\Omega, \end{cases} \quad (11)$$

where $\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$ is the *Laplacian* and $f : \Omega \rightarrow \mathbb{R}_{<0}$ is a given forcing function. As discussed in [22], under appropriate regularity assumptions on Ω , a smooth forcing function $f \in C^\infty(\bar{\Omega})$ yields a smooth solution $h \in C^\infty(\bar{\Omega})$ to (11). As demonstrated in [11], this smooth solution characterizes the safe set \mathcal{C} such that $\Omega = \text{Int}(\mathcal{C})$, $\partial\mathcal{C} = \partial\Omega$, and may be used to construct safety filters yielding safe control actions for (1) under appropriate relative degree assumptions.

D. Boundary Flux

Given an occupancy map, $\bar{\Omega}$, where $\hat{\mathbf{n}} : \partial\Omega \rightarrow \mathbb{R}^3$ denotes the outward pointing unit normal, collision avoidance applications require a negative outward directional derivative on the boundary, i.e., negative *boundary flux* $Dh(\mathbf{y}) \cdot \hat{\mathbf{n}}(\mathbf{y}) < 0$ for all $\mathbf{y} \in \partial\Omega$, to encode repulsive gradients on obstacle surfaces. From Hopf's Lemma [23], it follows that solving the Dirichlet problem (11) with a negative forcing function in the interior, i.e., $f(\mathbf{y}) < 0$ for all $\mathbf{y} \in \Omega$, guarantees $Dh(\mathbf{y}) \cdot \hat{\mathbf{n}}(\mathbf{y}) < 0$ for all $\mathbf{y} \in \partial\Omega$.

The ability to prescribe the magnitude of $Dh(\mathbf{y}) \cdot \hat{\mathbf{n}}(\mathbf{y})$ at each point $\mathbf{y} \in \partial\Omega$ provides a way to locally encode desired gradient strengths along obstacle surfaces. This enables the ability to prescribe stronger or weaker repulsive effects, depending on the relative importance of an obstacle (or portion of the obstacle). However, (11) does not provide the ability to prescribe $Dh(\mathbf{y}) \cdot \hat{\mathbf{n}}(\mathbf{y})$ directly in a point-wise manner. Instead, (11) constrains the boundary flux in an integral sense through the forcing function f . In particular, by the divergence theorem [24], all forcing functions satisfy (dropping dependency on \mathbf{y} for brevity):

$$\iiint_{\Omega} f \, dE = \iiint_{\Omega} \overbrace{\nabla \cdot (Dh)}^{\Delta h} \, dE = \overbrace{\iint_{\partial\Omega} Dh \cdot \hat{\mathbf{n}} \, dA}^{\text{Total Flux}}, \quad (12)$$

where dE, dA denote volume and area elements respectively.

One approach of encoding desired point-wise boundary flux magnitudes, proposed in [11], was to introduce an auxiliary vector field—a *guidance field*. In that formulation, the divergence of the guidance field served as a tool for defining the forcing function f . However, the resulting desired point-wise flux condition also held only in the integral sense via a variational problem. Moreover, the role of the guidance field was left implicit, treated merely as an auxiliary tool for defining a forcing function rather than a central object of study. In this work, we make the guidance field explicit and study its central role in enforcing safety specifications directly. We introduce a definition that captures the minimal requirements that such a field must satisfy to serve as a foundation for more expressive notions of safety, enabling obstacle-specific safety behaviors.

III. RISK-AWARE SAFETY-CRITICAL CONTROL USING GUIDANCE FIELDS

In this section, we present a new safety constraint that leverages *guidance fields* to enforce distinct gradient behavior across domain boundaries. This provides a way to assign relative importance to boundary regions, capturing obstacle risk and priorities. We begin by formalizing the notion of a guidance field. Intuitively, a guidance field is a vector field that prescribes repulsive directions along obstacle surfaces, and extends these directions smoothly into the domain.

Definition 4 (Guidance Field). Let $\Omega \subset \mathbb{R}^3$ be an open, bounded, and connected set representing free space with smooth boundary $\partial\Omega$ corresponding to obstacle surfaces, and let $\hat{\mathbf{n}} : \partial\Omega \rightarrow \mathbb{R}^3$ denote the unit normal pointing outward from Ω , i.e., into the obstacles. Given a prescribed negative boundary flux $b : \partial\Omega \rightarrow \mathbb{R}_{<0}$, we call a vector field $\vec{v} \in C^k(\bar{\Omega}; \mathbb{R}^3)$ with $k \geq 1$ a *guidance field* if:

$$\vec{v}(\mathbf{y}) \cdot \hat{\mathbf{n}}(\mathbf{y}) = b(\mathbf{y}), \quad \vec{v}(\mathbf{y}) \parallel \hat{\mathbf{n}}(\mathbf{y}) \quad \text{on } \partial\Omega, \quad (13)$$

and \vec{v} is a C^k extension of the flux into the interior Ω .

This definition ties the guidance field to obstacle boundaries through the flux $\vec{v}(\mathbf{y}) = b(\mathbf{y})\hat{\mathbf{n}}(\mathbf{y})$ on $\partial\Omega$, with negative values of b encoding repulsive gradients. For well-posedness, b must be sufficiently regular to admit a continuous extension into the domain. The C^k regularity condition ensures that derivatives are well defined in the classical sense for the divergence theorem to hold, and to provide continuous derivatives that are convenient for control design. An approach for constructing guidance fields satisfying Def. 4, is based on the *vector Laplace equation* proposed in [11].

A. Laplace Guidance Field

In the vector Laplace formulation, each component of \vec{v} is obtained by a harmonic extension of the boundary data. This produces a smooth interpolation of the boundary flux throughout the domain in each direction, yielding a smooth guidance field satisfying Def. 4. Specifically, consider $\vec{v} = (v_x, v_y, v_z) : \bar{\Omega} \rightarrow \mathbb{R}^3$, with each component satisfying

Laplace's equation subject to Dirichlet boundary conditions:

$$\begin{cases} \Delta v_i(\mathbf{y}) = 0 & \text{in } \Omega, \\ v_i(\mathbf{y}) = b(\mathbf{y})n_i(\mathbf{y}) & \text{on } \partial\Omega, \end{cases} \quad (14)$$

for $i \in \{x, y, z\}$, where $\hat{\mathbf{n}} = (n_x, n_y, n_z) : \partial\Omega \rightarrow \mathbb{R}^3$ denotes the outward unit normal vector such that $\vec{v}(\mathbf{y}) = b(\mathbf{y})\hat{\mathbf{n}}(\mathbf{y})$ on $\partial\Omega$, and $b : \partial\Omega \rightarrow \mathbb{R}_{<0}$ prescribes the outward directional derivative encoding the desired boundary flux encoding repulsive gradients. Although (14) produces a smooth vector field $\vec{v} \in C^\infty(\bar{\Omega}; \mathbb{R}^3)$ that matches the boundary specifications (13), the decoupled nature of the components of \vec{v} in the vector Laplace formulation (14) makes the field generally *nonconservative*, meaning it is not the gradient of a potential [24]. However, since \vec{v} satisfies (13) by construction, it can be directly used to enforce safety without the need of a potential, which we discuss next.

B. Risk-Aware Safety Filters

Given a safety function h defining a safe set \mathcal{C} as in Def. 3, the classical CBF condition (3) uses the gradient Dh to enforce safety. However, directly prescribing desired gradient magnitudes along $\partial\mathcal{C}$ with Dh is typically not possible. We generalize the classical CBF formulation (3) by decoupling the vector field used in the gradient term from the function h . In particular, we introduce the guidance field \vec{v} , which enables local, pointwise design of boundary-normal gradients (i.e., boundary flux) while maintaining safety. In this sense, the CBF condition (3) is reformulated with \vec{v} , opening new directions for safe vector-field generation.

We focus on systems defined by integrator chains as in (7), with the input appearing at the last layer; note, however, that our method can be extended to classes of systems with outputs of non-uniform relative degree [18], [19]. To formalize this, we begin with first-order systems.

1) First Order Systems: Consider the single integrator dynamics (relative degree $r = 1$):

$$\dot{\vec{y}} = \mathbf{w}, \quad (15)$$

where the state $\vec{y} = \mathbf{y} \in \mathbb{R}^3$. Given a guidance field \vec{v} , the following proposition establishes safety for (15).

Proposition 1. (*Forward Invariance of First Order Systems*) Let $\Omega \subset \mathbb{R}^3$ be an open, bounded, and connected set with smooth boundary $\partial\Omega$ and outward pointing normal $\hat{\mathbf{n}} : \partial\Omega \rightarrow \mathbb{R}^3$. Consider the system (15) and a safe set, \mathcal{C} , defined as the 0-super-level set of a safety function $h : \bar{\Omega} \rightarrow \mathbb{R}$ as in Def. 3 and satisfies $h(\mathbf{y}) = 0$ on $\partial\mathcal{C} = \partial\Omega$. Suppose that $\vec{v} \in C^1(\bar{\Omega}; \mathbb{R}^3)$ is a vector field satisfying $\vec{v}(\mathbf{y}) = b(\mathbf{y})\hat{\mathbf{n}}(\mathbf{y})$ on $\partial\Omega$ for a negative boundary flux $b : \partial\Omega \rightarrow \mathbb{R}_{<0}$ as in Def. 4, then for any locally Lipschitz continuous controller $\mathbf{k} : \bar{\Omega} \rightarrow \mathbb{R}^3$ satisfying:

$$\vec{v}(\mathbf{y}) \cdot \mathbf{k}(\mathbf{y}) \geq -\gamma h(\mathbf{y}) \quad \forall \mathbf{y} \in \mathcal{C}, \quad (16)$$

for some $\gamma > 0$, the set \mathcal{C} is rendered forward invariant.

Proof. Since we have that $\vec{v}(\mathbf{y}) = b(\mathbf{y})\hat{\mathbf{n}}(\mathbf{y})$ on $\partial\Omega$ and $h(\mathbf{y}) = 0$ on $\partial\mathcal{C} = \partial\Omega$, then there exists a controller \mathbf{k}

enforcing $\vec{v}(\mathbf{y}) \cdot \mathbf{k}(\mathbf{y}) = b(\mathbf{y})\hat{\mathbf{n}}(\mathbf{y}) \cdot \mathbf{k}(\mathbf{y}) \geq 0$ for all $\mathbf{y} \in \partial\mathcal{C}$. Since $\partial\Omega$ is the 0-level set of h , we have:

$$\hat{\mathbf{n}}(\mathbf{y}) = c(\mathbf{y}) \frac{Dh(\mathbf{y})}{\|Dh(\mathbf{y})\|}, \quad (17)$$

where $c : \partial\Omega \rightarrow \mathbb{R}_{<0}$ from Hopf's Lemma³ [11], [23]. Thus, the enforced inequality can be rewritten as:

$$b(\mathbf{y})\hat{\mathbf{n}}(\mathbf{y}) \cdot \mathbf{k}(\mathbf{y}) = b(\mathbf{y}) \frac{c(\mathbf{y})}{\|Dh(\mathbf{y})\|} Dh(\mathbf{y}) \cdot \mathbf{k}(\mathbf{y}) \geq 0, \quad (18)$$

$$\implies \dot{h}(\mathbf{y}) = Dh(\mathbf{y}) \cdot \mathbf{k}(\mathbf{y}) \geq 0, \quad (19)$$

for all $\mathbf{y} \in \partial\Omega$, where the implication follows from the strict negativity of $b(\mathbf{y})$ and $c(\mathbf{y}) \equiv -1$, the fact that $\|Dh(\mathbf{y})\| > 0$ on $\partial\Omega$, and the single integrator dynamics (15). In particular, [25] provides examples of a locally Lipschitz continuous controllers satisfying (19). Therefore, from Nagumo's theorem, the set \mathcal{C} is a rendered forward invariant for the system (15). \square

The above proposition states that if the guidance field \vec{v} is normal to $\partial\Omega$ towards the interior Ω , then a controller enforcing forward invariance of the safe set \mathcal{C} exists. Given a nominal controller $\mathbf{k}_{\text{nom}} : \bar{\Omega} \rightarrow \mathbb{R}^3$, a safety function h , and a guidance field \vec{v} with negative boundary flux, one example of a safe controller is the QP-based safety filter:

$$\mathbf{k}_{\text{QP}}(\mathbf{y}) = \arg \min_{\mathbf{w} \in \mathbb{R}^3} \|\mathbf{w} - \mathbf{k}_{\text{nom}}(\mathbf{y})\|_2^2 \quad (20)$$

$$\text{s.t. } \vec{v}(\mathbf{y}) \cdot \mathbf{w} \geq -\gamma h(\mathbf{y}), \quad (21)$$

whose closed-form expression is:

$$\mathbf{k}_{\text{QP}}(\mathbf{y}) = \mathbf{k}_{\text{nom}}(\mathbf{y}) + \frac{\text{ReLU}(-a(\mathbf{y}))}{\|\vec{v}(\mathbf{y})\|^2} \vec{v}(\mathbf{y}), \quad (22)$$

where $\text{ReLU}(-a(\mathbf{y})) := \max\{0, -a(\mathbf{y})\}$ is an *activation function* with $a : \bar{\Omega} \rightarrow \mathbb{R}$ given by:

$$a(\mathbf{y}) := \vec{v}(\mathbf{y}) \cdot \mathbf{k}_{\text{nom}}(\mathbf{y}) + \gamma h(\mathbf{y}). \quad (23)$$

The controller (22) modifies the nominal input \mathbf{k}_{nom} in the direction of \vec{v} in a minimally invasive fashion whenever \mathbf{k}_{nom} violates the CBF constraint (21). Specifically, the second term on the right-hand side of (22) is a correction term, and the role of $\text{ReLU}(-a(\mathbf{y}))$ is to activate the modification only when $a(\mathbf{y}) \leq 0$. Concretely, if $a(\mathbf{y}) > 0$, then \mathbf{k}_{nom} satisfies the constraint and the filter remains inactive, and if $a(\mathbf{y}) \leq 0$, the filter is active and the nominal controller is modified along $\vec{v}(\mathbf{y})$. This implies that the filter is triggered into activation exactly on the 0-level set of a , i.e. when $a(\mathbf{y}) = 0$. We define these regions as *activation zones*, whose size and shape are dictated by \mathbf{k}_{nom} , \vec{v} , h and γ as follows:

Definition 5 (*Activation Zone*). Let $\mathbf{k}_{\text{nom}} : \bar{\Omega} \rightarrow \mathbb{R}^3$ be a nominal controller, $\vec{v} : \bar{\Omega} \rightarrow \mathbb{R}^3$ a guidance field, $h : \bar{\Omega} \rightarrow \mathbb{R}$ a safety function and $\gamma > 0$. The *activation zone* is the set:

$$\mathcal{A} := \{ \mathbf{y} \in \bar{\Omega} \mid a(\mathbf{y}) = 0 \}, \quad (24)$$

where $a(\mathbf{y})$ is defined in (23).

³Generally $c(\mathbf{y}) \in \{+1, -1\}$; if $h(\mathbf{y}) > 0$ in Ω , then $c(\mathbf{y}) \equiv -1$ on $\partial\Omega$.

Laplace Guidance Field

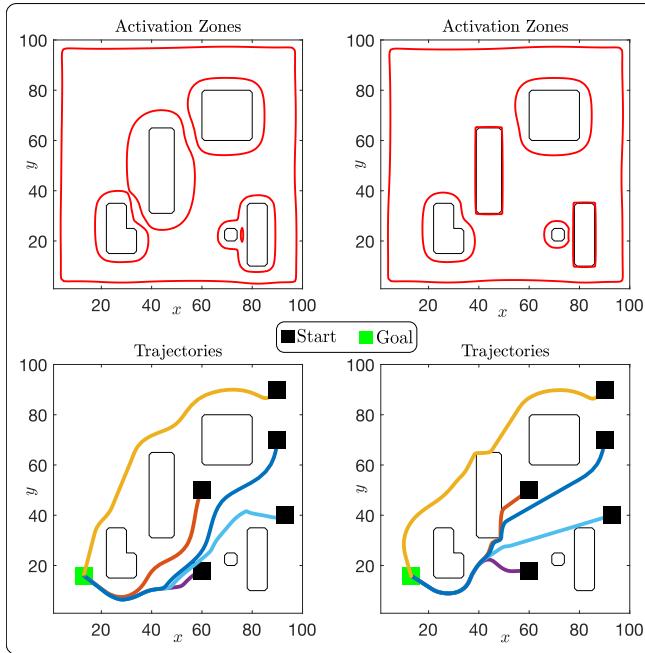


Fig. 2: Comparison of activation zones (23) for varying boundary flux with, decreasing the flux for the center and bottom right obstacles. Reducing $\|b(\mathbf{y})\|$ yields shallower boundary gradients and smaller activation zones, allowing trajectories closer to the obstacle, whereas larger $\|b(\mathbf{y})\|$ produces steeper gradients, larger activation zones and more conservative behavior. **Top row:** Activation zones (24) with $\mathbf{k}_{\text{nom}}(\mathbf{y}) = -\mu D h(\mathbf{y})$, $\mu > 0$, driving in the direction of steepest decrease of h (i.e., the worst-case, adversarial direction). **Bottom row:** Closed-loop trajectories for (15) with (22) and $\mathbf{k}_{\text{nom}}(\mathbf{y}) = -\mu(\mathbf{y} - \mathbf{y}_d)$, driving towards the goal \mathbf{y}_d .

Activation zones reveal the regions where the safety filter responds to each obstacle, resulting in distinctive behaviors around individual obstacles; a larger activation zone means that the filter reacts to that obstacle sooner. Adjusting γ uniformly expands or contracts all activation zones in the same way, resulting in a global effect. In contrast, isolated behavior around obstacles can be achieved by using a guidance field that enforces a pointwise boundary flux $b(\mathbf{y})$ at each $\mathbf{y} \in \partial\Omega$. Figure 2 shows how varying the boundary flux b in the guidance field $\vec{\mathbf{v}}$ generated by the Dirichlet problem for Laplace (14) produces different activation zones.

The above result can be extended to facilitate dynamic environments with moving obstacles [26]. The following corollary establishes the forward invariance for the single integrator system (15) for changing environments.

Corollary 1. (Forward Invariance of Time-Varying Safe Sets)
Let the assumptions of Proposition 1 hold. Let $h : \mathbb{R}^3 \times [0, T] \rightarrow \mathbb{R}_{\geq 0}$ be continuously differentiable in t for each \mathbf{y} , i.e., $h(\mathbf{y}, \cdot) \in C^1([0, T])$ for some $T > 0$, such that the time-varying 0-superlevel set of h characterizes the safe set:

$$\mathcal{C}_T(t) = \{ \mathbf{y} \in \mathbb{R}^3 \mid h(\mathbf{y}, t) \geq 0 \}, \quad t \in [0, T], \quad (25)$$

with $h(\mathbf{y}, t) = 0$ on $\partial\mathcal{C}_T(t)$. Let $\sigma : \mathbb{R}_{\geq 0} \rightarrow [0, \epsilon]$ be a smooth transition function, for some $\epsilon > 0$ satisfying $\sigma(0) = 0$ and $\sigma(a) \rightarrow \epsilon$ as $a \rightarrow \infty$. If there exists a locally Lipschitz

continuous controller $\mathbf{k} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ such that, for any $\gamma > 0$ and all $(\mathbf{y}, t) \in \mathbb{R}^3 \times [0, T]$:

$$\vec{\mathbf{v}}(\mathbf{y}) \cdot \mathbf{k}(\mathbf{y}) + \frac{\|\vec{\mathbf{v}}(\mathbf{y})\|}{\|Dh(\mathbf{y}, t)\| + \sigma(h(\mathbf{y}, t))} \cdot \frac{\partial h}{\partial t}(\mathbf{y}, t) \geq -\gamma h(\mathbf{y}, t), \quad (26)$$

then the set $\mathcal{C}_T(t)$ is rendered forward invariant $\forall t \in [0, T]$.

Proof. The result follows by applying the proof of Proposition 1 to the time-varying case, noting that the coefficient of $\frac{\partial h}{\partial t}$ ensures $\dot{h}(\mathbf{y}, t) \geq 0$ on the boundary. That is, on boundary, since we have have that $\vec{\mathbf{v}}(\mathbf{y}) \parallel Dh(\mathbf{y}, t)$, we have (dropping dependency on (\mathbf{y}, t) for brevity):

$$\vec{\mathbf{v}} = \frac{\|\vec{\mathbf{v}}\|}{\|Dh\|} Dh \text{ on } \partial\mathcal{C}_T(t), \quad (27)$$

and $\sigma(0) = 0$. Then (26) reduces to:

$$\vec{\mathbf{v}} \cdot \mathbf{k} + \frac{\|\vec{\mathbf{v}}\|}{\|Dh\|} \frac{\partial h}{\partial t} = \frac{\|\vec{\mathbf{v}}\|}{\|Dh\|} Dh \cdot \mathbf{k} + \frac{\|\vec{\mathbf{v}}\|}{\|Dh\|} \frac{\partial h}{\partial t} \quad (28)$$

$$= \frac{\|\vec{\mathbf{v}}\|}{\|Dh\|} \left(Dh \cdot \mathbf{k} + \frac{\partial h}{\partial t} \right) \quad (29)$$

$$= \frac{\|\vec{\mathbf{v}}\|}{\|Dh\|} \dot{h} \geq 0. \quad (30)$$

Therefore, by enforcing (26), we equivalently enforce $\dot{h} \geq 0$ on $\partial\mathcal{C}_T(t)$, rendering the set $\mathcal{C}_T(t)$ forward invariant. \square

Naturally, from this result and in analogy with the static case, the corresponding activating function $(\mathbf{y}, t) \mapsto a(\mathbf{y}, t)$ for the dynamic environment is defined as:

$$a := \vec{\mathbf{v}} \cdot \mathbf{k}_{\text{nom}} + \frac{\|\vec{\mathbf{v}}\|}{\|Dh\| + \sigma(h)} \cdot \frac{\partial h}{\partial t} + \gamma h. \quad (31)$$

The associated activation zones are given by the zero level set of the dynamic activation function (31). The above results also hold for time-varying guidance fields $(\mathbf{y}, t) \mapsto \vec{\mathbf{v}}(\mathbf{y}, t)$.

C. High Order Systems

The above results can be extended to high-order systems of the form (7) with relative degree $r \geq 2$ by leveraging CBF backstepping [27]. In the first-order case, the safety function h only needs to be continuous. However, for higher-order systems, additional regularity of h is required. To illustrate this, for convenience of exposition, we consider systems of relative degree $r = 2$. We consider the system $\vec{\mathbf{y}} = [\mathbf{y}, \mathbf{w}]^\top$, where the state is $\vec{\mathbf{y}} = [\mathbf{y}, \dot{\mathbf{y}}]^\top \in \mathbb{R}^4$. Given a twice continuously differentiable safety function $h \in C^2(\bar{\Omega})$ defining the safe set \mathcal{C} , define:

$$h_B = h - \frac{1}{2\mu} \|\dot{\mathbf{y}} - \mathbf{k}_{\vec{\mathbf{v}}}\|^2, \quad (32)$$

with $\mu > 0$ where $\mathbf{k}_{\vec{\mathbf{v}}} \in C^2(\bar{\Omega}; \mathbb{R}^3)$ satisfies $\vec{\mathbf{v}} \cdot \mathbf{k}_{\vec{\mathbf{v}}} \geq -\gamma h$ for all $\mathbf{y} \in \mathcal{C}$ [11], [18]. A smooth example, $\mathbf{k}_{\vec{\mathbf{v}}} \in C^\infty(\bar{\Omega}; \mathbb{R}^3)$, that remains close to the min-norm activation zones is given in [25]. The 0-superlevel set of h_B defines the shrunken set:

$$\mathcal{C}_B = \{ \mathbf{y} \in \mathbb{R}^6 \mid h_B(\mathbf{y}) \geq 0 \} \subset \mathcal{C} \times \mathbb{R}^3. \quad (33)$$

Since $h_B(\mathbf{y}) \leq h(\mathbf{y})$ for all $\mathbf{y} \in \mathcal{C}$, a guidance field $\vec{\mathbf{v}}$ can be used to ensure that all trajectories starting in \mathcal{C} remain

in \mathcal{C} by rendering \mathcal{C}_B safe. This is possible because $\mathbf{y} \in \mathcal{C}_B \cap \partial\mathcal{C}$ implies $\dot{\mathbf{y}} = \mathbf{k}_{\vec{\mathbf{v}}}$ and under this condition, $\vec{\mathbf{v}} \cdot \mathbf{k}_{\vec{\mathbf{v}}} \geq -\gamma h$ guarantees that $\vec{\mathbf{v}}(\mathbf{y}) \cdot \dot{\mathbf{y}} > 0$ at all boundary points. Specifically, this leads to the following safety constraint:

$$\vec{\mathbf{v}} \cdot \dot{\mathbf{y}} - \frac{1}{\mu}(\dot{\mathbf{y}} - \mathbf{k}_{\vec{\mathbf{v}}})\dot{\mathbf{k}}_{\vec{\mathbf{v}}} + \frac{1}{\mu}(\dot{\mathbf{y}} - \mathbf{k}_{\vec{\mathbf{v}}})\mathbf{w} \geq -\gamma h_B. \quad (34)$$

Note that $\mathbf{k}_{\vec{\mathbf{v}}}$ typically depends on h , so computing its gradients requires higher-order derivatives of h . For this reason, constructing h as a Poisson safety function (11) is advantageous, since its smoothness guarantees the necessary regularity [11]. Leveraging the above result, it follows from [27, Theorem 4] that there exists a locally Lipschitz continuous controller rendering \mathcal{C}_B forward invariant. Specifically, if $\mathbf{y}_0 = (\mathbf{y}_0, \dot{\mathbf{y}}_0) \in \mathcal{C}_B$, then $\mathbf{y}(t) \in \mathcal{C}_B$ for all $t \in I_{\max}(\mathbf{y}_0)$. Extensions to higher-order systems with relative degree $r > 2$ follow by the same construction; see [11].

Algorithm 1: Construct Risk-Aware Safety Filter

```

Input:  $\Omega, \partial\Omega, \mathcal{F}, \mathcal{P}, w, \Phi, d, f, \mathbf{k}_{\text{nom}}, \gamma$ 
 $\partial\Omega_d, \Omega_d \leftarrow \text{discretize}(\partial\Omega, \Omega, d)$ 
 $\mathcal{Z} \leftarrow \{(\mathbf{y}, f(\mathbf{y}))\} \text{ for each } \mathbf{y} \in \Omega_d$ 
 $\mathcal{Y} \leftarrow \{\}$ 
for  $\mathbf{y} \in \partial\Omega_d$  do
| feature  $\leftarrow \mathcal{F}(\mathbf{y})$ 
| priority  $\leftarrow \mathcal{P}(\text{feature})$ 
| risk  $\leftarrow w(\text{priority})$ 
|  $b \leftarrow \Phi(w)$ 
|  $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{(\mathbf{y}, b)\}$ 
|  $h \leftarrow \text{DirichletForPoisson}(\Omega_d, \partial\Omega_d, \mathcal{Z})$  // As in (11)
|  $\vec{\mathbf{v}} \leftarrow \text{DirichletForLaplace}(\Omega_d, \partial\Omega_d, \mathcal{Y})$  // As in (14)
|  $\mathbf{k}_{QP} \leftarrow \text{BuildSafetyFilter}(\mathbf{k}_{\text{nom}}, \gamma, h, \vec{\mathbf{v}})$  // As in (21)
Output:  $\mathbf{k}_{QP}$ 
```

IV. ENCODING VARIABLE CONSERVATISM VIA BOUNDARY FLUX

In safety-critical applications, different obstacles typically demand different levels of caution. The *boundary flux* of a guidance field provides a principled way to encode such priorities directly on obstacle surfaces. In particular, the boundary flux b quantifies how strongly the guidance field $\vec{\mathbf{v}}$ points outward along the surface normal, $\vec{\mathbf{v}} = b\hat{\mathbf{n}}$, providing a way to encode desired levels of caution around an obstacle:

$$b(\mathbf{y}) \longrightarrow \text{Level of caution at point } \mathbf{y} \in \partial\Omega.$$

More concretely, by prescribing boundary flux values b , we control how activation zones expand or contract around obstacles: large magnitudes create wider activation zones, while smaller magnitudes yield thinner zones as in Fig. 2.

To make this process constructive, we begin by using a function $\mathcal{F} : \partial\Omega \rightarrow \mathcal{L}$ to associate each obstacle position $\mathbf{y} \in \partial\Omega$ with a *feature* from a set of labels \mathcal{L} that characterizes a desired property of an obstacle or part of an obstacle (e.g., whether the obstacle is a human or a wall). We then use a priority function $\mathcal{P} : \mathcal{L} \rightarrow \mathbb{R}_{\geq 0}$ to order the set of labels \mathcal{L} , incorporating a user-defined *priority ranking*, specifying which obstacles (or parts of obstacles) demand more caution than others according to the user's preference. The priorities

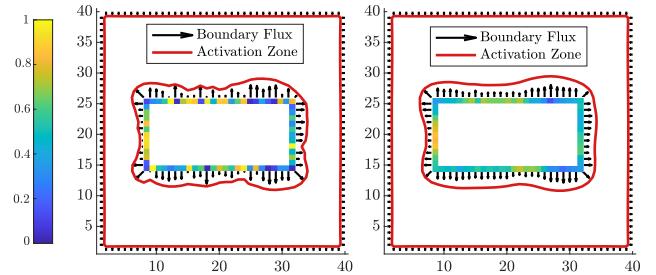


Fig. 3: Activation zones based on obstacle uncertainty. Left: Without smoothing, boundary fluxes vary irregularly. Right: Smoothing forces spatial regularity. High confidence regions (yellow) yield tighter activation zones, while low confidence regions (blue) yield expanded activation zones, reflecting the risk prioritization.

are then converted to “risk” values, which are used to design the flux of the guidance field, yielding the procedure:

State \mapsto Feature \mapsto Priority \mapsto Assigned Risk \mapsto Flux.

To ensure that the assigned risks lie on a consistent scale, we normalize them to lie within the interval $[0, 1]$ with a risk-assignment map $w : \mathbb{R}_{>0} \rightarrow [0, 1]$. If the range of \mathcal{P} is bounded (e.g., probabilities), these values can directly define risk; for example, by rescaling or using the identity map $w(p) = p$ if $p \in [0, 1]$. However, if the range of \mathcal{P} is not bounded (e.g., when priority is induced by obstacle speed), we use a smooth, bounded, monotonic risk-assignment w to induce the $[0, 1]$ bound. Finally, the bounded risk is then mapped to flux values via the function $\Phi : [0, 1] \rightarrow \mathbb{R}_{<0}$.

A. Constructing Risk-Aware Safety Filters

Next, we provide Algorithm 1 for constructing risk-aware safety filters from environment descriptions, priority rules, and risk assignments. Using this algorithm requires the following additional inputs: $d \in \mathbb{R}_{>0}$, the discretization resolution size for solving the Poisson and Laplace problems; f , the forcing function used when solving for the Poisson safety function h in (11); $\mathbf{k}_{\text{nom}} : \mathbb{R}^n \rightarrow \mathbb{R}^m$, the potentially unsafe nominal control action; and $\gamma \in \mathbb{R}_{>0}$, the scalar value used in the safety constraint as in (21). The output of Algorithm 1 is a controller, \mathbf{k}_{QP} , as in (20) that guarantees safety and has tunable activation zones that are a result of the risk assigned to the features in the environment.

This algorithm is intentionally very general and often simplifies significantly through the choices of inputs. To illustrate this simplicity and the practical utility of this algorithm, we provide several examples of how risk-aware safe controllers and their associated activation zones can be generated from relevant environmental features.

V. CASE STUDIES: ENVIRONMENTAL FEATURES

In this section, we provide example applications of Algorithm 1 to three scenarios: (A) uncertain surface boundaries, (B) dynamic obstacles, and (C) objects in the environment with varying levels of user-assigned semantic priority.

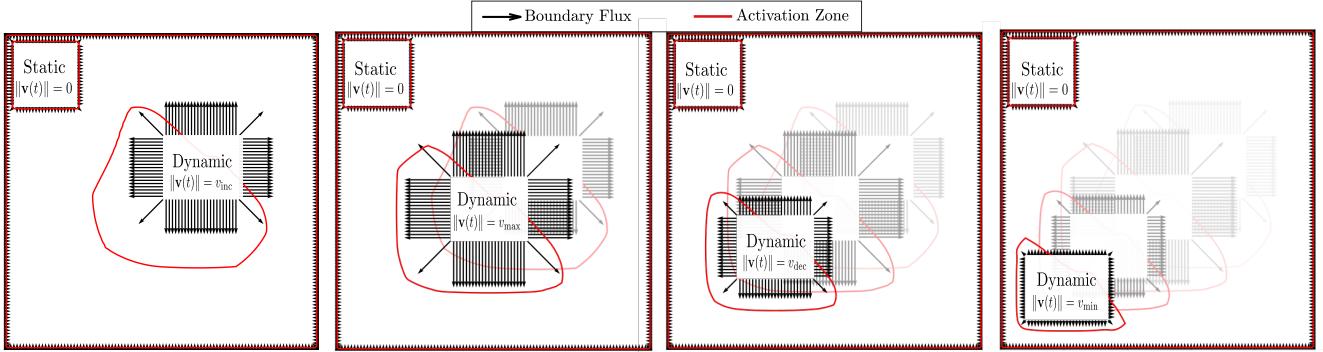


Fig. 4: Time-lapse of the activation zones with a moving object in the environment, defined by the zero level set of the function (31). The object's velocity increases to a maximum before decreasing to a minimum. Larger velocities yield higher boundary fluxes and hence larger activation zones. These zones extend in the direction of motion and are affected by the object's acceleration; they expand during acceleration and contract during deceleration, reflecting the rate of change of the safe set.

A. Geometric Features: Obstacle Uncertainty

While location information helps to estimate an obstacle's position, sensors and mapping algorithms introduce significant uncertainty into these estimates. Thus, in this example, we introduce variable conservatism of the safety filter depending on the surface's uncertainty.

To apply Algorithm 1, we define the feature map to be the probability of occupancy at the surface location:

$$\mathcal{F}(\mathbf{y}) = \mathbb{P}\left(\text{occupied at } \mathbf{y} \mid \sum_{i=1}^{n_{\text{meas}}} \hat{\mathbf{m}}_i\right) \in [0, 1], \quad (35)$$

where $n_{\text{meas}} \in \mathbb{N}$ is the number of noisy sensor measurements $\hat{\mathbf{m}}$. Next we let $\mathcal{P}(p) = 1 - p$ so that low probability (i.e., high uncertainty) are high priority and we let w be the identity function to preserve these risk values. Finally we choose Φ to linearly interpolate between b_{\min} and b_{\max} so that smaller probabilities lead to larger activation zones and larger probabilities lead to smaller activation zones. The results of this method can be seen in Figure 3 where we see that high-confidence regions (yellow) result in activation regions closer to the surface of $\partial\Omega$ and low confidence regions (blue) result in activation regions that are further away from $\partial\Omega$, leading to more conservatism near uncertain regions of the obstacle.

B. Dynamic Features: Obstacle Motion

Motion describes how an obstacle or region evolves over time and introduces additional features such as velocity and acceleration which are continuous and potentially unbounded. These dynamic features may require additional normalization to be incorporated into Algorithm 1.

First, we define a point's "feature" to be the magnitude of its velocity, i.e. $\mathcal{F}(\mathbf{y}) = \|\dot{\mathbf{y}}\|$ for $\mathbf{y} \in \partial\Omega$. Second, since we prefer our system to be more cautious around high-velocity obstacles, we let the priority function \mathcal{P} be the identity function, meaning that faster objects have higher priority. Next, since $\|\dot{\mathbf{y}}\|$ can be unbounded, we normalize this priority value to determine the assigned risk using the function $w(r) = r/(v_{\text{ref}} + r)$ so that:

$$w(\mathcal{P}(\mathcal{F}(\mathbf{y}))) = \frac{\|\dot{\mathbf{y}}\|}{v_{\text{ref}} + \|\dot{\mathbf{y}}\|} \in [0, 1], \quad (36)$$

where $v_{\text{ref}} \in \mathbb{R}_{>0}$ is a reference speed such that $w(\mathcal{P}(\mathcal{F}(\mathbf{y}))) = 0.5$ when the speed of the obstacle at $\mathbf{y} \in \partial\Omega$ is $\|\dot{\mathbf{y}}\| = v_{\text{ref}}$. Finally, we again define Φ as a function that linearly interpolates between b_{\min} and b_{\max} using the risk value. The results of this method can be seen in Figure 4 which shows a time-lapse of a moving obstacle and the evolution of the flux values and dynamic activation zones. Here we see that dynamic obstacles are surrounded by larger activation zones than static obstacles, and the activation zones grow uniformly with the obstacle's speed. The activation zones are aligned with the direction of motion and expand in this direction as it accelerates.

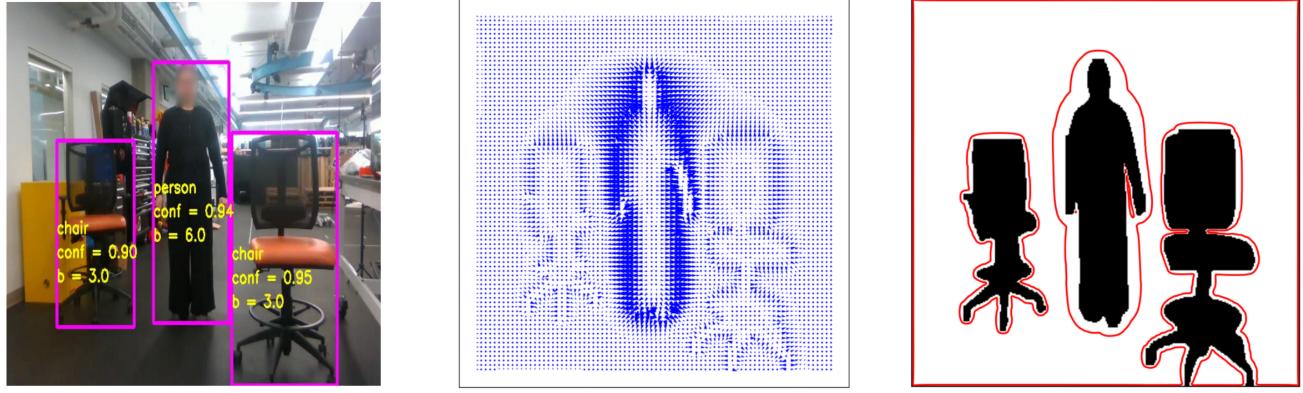
C. Semantic Features: Object Type and Priority

Next, we extend beyond numerical features and consider the case where we require varying levels of conservatism for each obstacle depending on *what kind* of obstacle it is. Semantic features capture *what* an obstacle is, rather than *where* it is or *how* it moves. Unlike geometric and dynamic features which can be measured directly, semantic features depend on classification, where obstacles are assigned to discrete categories or labels.

For this example, we consider the discrete set of obstacle labels $\mathcal{L} = \{\text{wall}, \text{chair}, \text{human}\}$, where we obtain the label from a point $\mathbf{y} \in \partial\Omega$ using \mathcal{F} as $\mathbf{y} \mapsto \mathcal{F}(\mathbf{y}) \in \mathcal{L}$. Next, we assign the priority of these semantic labels to be $\mathcal{P}(\text{wall}) = 1$, $\mathcal{P}(\text{chair}) = 3$, and $\mathcal{P}(\text{human}) = 6$. We then assign the risk value using the exponential function $w(r) = 1 - e^{-\alpha r}$ for $\alpha > 0$ which maps to $[0, 1]$ and applies a high risk to obstacles based on assigned priority. Finally, we again define Φ to be a linear interpolation between the minimum and maximum flux values b_{\min} and b_{\max} associating low risk with low flux and high risk with high flux.

Figure 5 shows the result of applying Algorithm 1 in this fashion. There, the guidance fields and activation zones around the human are much larger than around the chair and wall obstacles, consistent with the assigned priorities.

Remark 1 (Combining features). Features can also be fused to capture richer context depending on design objectives, provided that the combined mapping preserves the intended priority order before flux assignment.



(a) YOLO Semantic Segmentation.

(b) Guidance Field generation.

(c) Activation zones.

Fig. 5: An occupancy map is obtained from semantic segmentation of perception data using a pretrained YOLOv11 model [28] with user designated risk values, assigned according to label classes as by Algorithm 1: $b_{\text{wall}} = 1$, $b_{\text{chair}} = 3$ and $b_{\text{person}} = 6$. The occupancy map and flux information are then used to generate the guidance field and derive activation zones.

VI. CONCLUSION

We presented a method for synthesizing risk-aware safety filters using Poisson safety functions and Laplace guidance fields. By adjusting boundary flux, the approach yields tunable boundary gradients around obstacle surfaces, enabling tunable activation zones and conservatism associated with risk. We demonstrated risk-aware safety across diverse contexts; environments with probabilistic occupancies, dynamic obstacles, and semantic labels. Future directions include extensions to general classes of nonlinear systems with closed-loop hardware implementations in real-world settings.

REFERENCES

- [1] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, “Control barrier function based quadratic programs for safety critical systems,” *IEEE Trans. Autom. Control*, vol. 62, no. 8, pp. 3861–3876, 2017.
- [2] A. Alan, A. J. Taylor, C. R. He, G. Orosz, and A. D. Ames, “Safe controller synthesis with tunable input-to-state safe control barrier functions,” *IEEE Control Systems Letters*, vol. 6, pp. 908–913, 2022.
- [3] K. Long, C. Qian, J. Cortés, and N. Atanasov, “Learning barrier functions with memory for robust safe navigation,” *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4931–4938, 2021.
- [4] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, “Control barrier functions: theory and applications,” in *Proc. Eur. Control Conf.*, pp. 3420–3431, 2019.
- [5] T. G. Molnar, R. Cosner, A. Singletary, W. Ubellacker, and A. Ames, “Model-free safety-critical control for robotic systems,” *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 944–951, 2022.
- [6] R. Cosner, M. Tucker, A. Taylor, K. Li, T. Molnar, W. Ubellacker, A. Alan, G. Orosz, Y. Yue, and A. Ames, “Safety-aware preference-based learning for safety-critical control,” in *Learning for Dynamics and Control Conference*, pp. 1020–1033, PMLR, 2022.
- [7] P. Glotfelter, J. Cortés, and M. Egerstedt, “Boolean componability of constraints and control synthesis for multi-robot systems via nonsmooth control barrier functions,” in *2018 IEEE Conference on Control Technology and Applications (CCTA)*, pp. 897–902, IEEE, 2018.
- [8] J. J. Choi, D. Lee, K. Sreenath, C. J. Tomlin, and S. L. Herbert, “Robust control barrier-value functions for safety-critical control,” in *2021 60th IEEE Conference on Decision and Control (CDC)*, pp. 6814–6821, IEEE, 2021.
- [9] Z. Qin, K. Zhang, Y. Chen, J. Chen, and C. Fan, “Learning safe multi-agent control with decentralized neural barrier certificates,” *arXiv preprint arXiv:2101.05436*, 2021.
- [10] A. Robey, H. Hu, L. Lindemann, H. Zhang, D. V. Dimarogonas, S. Tu, and N. Matni, “Learning control barrier functions from expert demonstrations,” in *2020 59th IEEE Conference on Decision and Control (CDC)*, pp. 3717–3724, Ieee, 2020.
- [11] G. Bahati, R. M. Bena, and A. D. Ames, “Dynamic safety in complex environments: Synthesizing safety filters with poisson’s equation,” in *Robotics: Science and Systems*, 2025.
- [12] G. Bahati and A. D. Ames, “Safe set synthesis with tunable boundary gradients via poisson safety functions,” *IEEE International Conference on Robotics and Automation (ICRA): Workshop on Robot safety under uncertainty from intangible specifications*, 2025.
- [13] A. Ames, X. Xu, J. Grizzle, and P. Tabuada, “Control barrier function based quadratic programs for safety critical systems,” vol. 62, no. 8, pp. 3861–3876, 2017.
- [14] A. Alan, A. J. Taylor, C. R. He, G. Orosz, and A. D. Ames, “Safe controller synthesis with tunable input-to-state safe control barrier functions,” *IEEE Contr. Syst. Lett.*, vol. 6, pp. 908–913, 2022.
- [15] R. K. Cosner, Y. Chen, K. Leung, and M. Pavone, “Learning responsibility allocations for safe human-robot interaction with applications to autonomous driving,” *arXiv preprint arXiv:2303.03504*, 2023.
- [16] P. Akella, A. Dixit, M. Ahmadi, L. Lindemann, M. P. Chapman, G. J. Pappas, A. D. Ames, and J. W. Burdick, “Risk-aware robotics: Tail risk measures in planning, control, and verification [focus on education],” *IEEE Control Systems*, vol. 45, no. 4, pp. 46–78, 2025.
- [17] A. Isidori, *Nonlinear control systems: an introduction*. Springer, 1985.
- [18] M. H. Cohen, R. K. Cosner, and A. D. Ames, “Constructive safety-critical control: Synthesizing control barrier functions for partially feedback linearizable systems,” *IEEE Control Systems Letters*, 2024.
- [19] G. Bahati, R. K. Cosner, M. H. Cohen, R. M. Bena, and A. D. Ames, “Control barrier function synthesis for nonlinear systems with dual relative degree,” *2025 IEEE 64st Conference on Decision and Control (CDC)*, 2025.
- [20] A. J. Taylor, P. Ong, T. G. Molnar, and A. D. Ames, “Safe backstepping with control barrier functions,” in *Proc. Conf. Decis. Control*, pp. 5775–5782, 2022.
- [21] L. Doeser, P. Nilsson, A. D. Ames, and R. M. Murray, “Invariant sets for integrators and quadrotor obstacle avoidance,” in *Proceedings of the American Control Conference*, pp. 3814–3821, 2020.
- [22] D. Gilbarg and N. S. Trudinger, *Elliptic partial differential equations of second order*, vol. 224. Springer, 1977.
- [23] M. H. Protter and H. F. Weinberger, *Maximum principles in differential equations*. Springer Science & Business Media, 2012.
- [24] J. E. Marsden and A. Tromba, *Vector calculus*. Macmillan, 2003.
- [25] M. H. Cohen, P. Ong, G. Bahati, and A. D. Ames, “Characterizing smooth safety filters via the implicit function theorem,” *IEEE Contr. Syst. Lett.*, vol. 7, pp. 3890–3895, 2023.
- [26] R. M. Bena, G. Bahati, B. Werner, R. K. Cosner, L. Yang, and A. D. Ames, “Geometry-aware predictive safety filters on humanoids: From poisson safety functions to cbf constrained mpc,” *IEEE-RAS 24th International Conference on Humanoid Robots (Humanoids)*, 2025.
- [27] A. J. Taylor, P. Ong, T. G. Molnar, and A. D. Ames, “Safe backstepping with control barrier functions,” in *2022 IEEE 61st Conference on Decision and Control (CDC)*, pp. 5775–5782, 2022.
- [28] G. Jocher, J. Qiu, and A. Chaurasia, “Ultralytics YOLO,” Jan. 2023.