SAFER-Splat: A Control Barrier Function for Safe Navigation with Online Gaussian Splatting Maps

Timothy Chen[1*] Aiden Swann[1*] Javier Yu[1*] Ola Shorinwa[1] Riku Murai[2]
Monroe Kennedy III[1] Mac Schwager[1]

Abstract— SAFER-Splat (Simultaneous Action Filtering and Environment Reconstruction) is a real-time, scalable, and minimally invasive action filter, based on control barrier functions, for safe robotic navigation in a detailed map constructed at runtime using Gaussian Splatting (GSplat). We propose a novel Control Barrier Function (CBF) that not only induces safety with respect to all Gaussian primitives in the scene, but when synthesized into a controller, is capable of processing hundreds of thousands of Gaussians while maintaining a minimal memory footprint and operating at 15 Hz during online Splat training. Of the total compute time, a small fraction of it consumes GPU resources, enabling uninterrupted training. The safety layer is minimally invasive, correcting robot actions only when they are unsafe. To showcase the safety filter, we also introduce SplatBridge, an open-source software package built with ROS for real-time GSplat mapping for robots. We demonstrate the safety and robustness of our pipeline first in simulation, where our method is 20-50x faster, safer, and less conservative than competing methods based on neural radiance fields. Further, we demonstrate simultaneous GSplat mapping and safety filtering on a drone hardware platform using only on-board perception. We verify that under teleoperation a human pilot cannot invoke a collision. Our videos and codebase can be found at `https://chengine.github.io/safer-splat`.

I. Introduction

Spatial understanding is an essential component of a robot's autonomy stack, and for many robotic platforms it is achieved through online mapping using the robot's on-board sensor suite. Mapping in robotics has traditionally been built on point-cloud and voxel-based scene representations [1]–[4]. Although these representations can be designed to be lightweight, the sparsity of these representations often limits the resolution and fidelity of the resulting map. Recently, radiance fields, such as Neural Radiance Fields (NeRFs) [5], [6] and Gaussian Splatting (GSplat) [7]–[9], have emerged as photorealistic, high-fidelity scene reconstruction methods built off the same input modalities (i.e., RGB or RGB-D images) as their classical counterparts. In contrast to NeRFs, GSplats represent the environment using explicit, ellipsoidal primitives, lending itself to efficient, dynamic robotic manipulation and navigation algorithms [10]–[14], while offering faster training and rendering speeds than NeRFs. In this work, we propose a Control Barrier Function (CBF) safety filter for robot platforms which leverages a GSplat's simple geometric
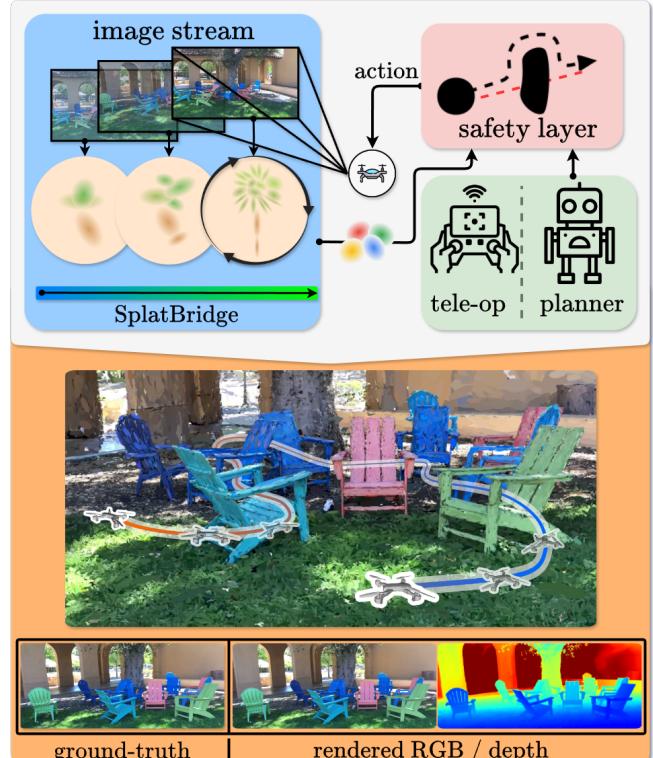

Fig. 1. SAFER-Splat is a safety layer for online robotic GSplat mapping.

primitives for efficient computation of certifiably safe control. SAFER-Splat is a low-level safety filter, meaning it modifies higher-level commands, from a human pilot through teleop, or through a higher-level planner, to deliver collision-free actions with minimal deviation from the action intended by the higher level command.

Although algorithms for safe robotic control have been developed for a broad range of map representations, e.g., [14]–[18], these existing methods either require a pre-built map, making them unsuitable for online scenarios, or require stringent assumptions on the robot's dynamics, sensing modalities, or its nominal controller. SAFER-Splat leverages a novel CBF, closely integrated with the GSplat representation, to derive a lightweight, minimally-invasive safe action filter. To synthesize a controller, we embed the CBF safety constraint into a quadratic program, minimizing the deviation between the desired and actuated control effort. By pruning the CBF constraints to identify a minimal number of constraints, our method scales efficiently to run in real-time with hundred of thousands ellipsoidal primitives in the scene. Together, the

---

* The co-first authors contributed equally.
[1] Stanford University, Stanford, CA, USA.
[2] Imperial College London, London, UK.
Corresponding author: `chengine@stanford.edu`.

safety guarantees, computational efficiency, and scalability of SAFER-Splat enable its superior performance in collision-free robot control in online GSplat scenes. To showcase SAFER-Splat in a full autonomy stack, we also introduce SplatBridge, a ROS bridge for real-time, online GSplat training for robots.

In simulation, we demonstrate that SAFER-Splat outperforms other radiance field-based controllers [19]. We show that the proposed method is more than an order of magnitude faster, not sporadic in performance, and is consistently safer. In real-world experiments with a teleoperated drone, we validate that SAFER-Splat, using SplatBridge as its online GSplat module, with only data from on-board perception, successfully produces collision-free robot motion even when the operator attempts to force collisions with obstacles. While we demonstrate our CBF safety filter with SplatBridge, the concept is modular, and can be composed with other GSplat SLAM techniques (e.g., [20]) to give real-time, photo-realistic scene reconstruction and collision-free robot motion.

II. RELATED WORK

In this section, we review the related literature in radiance fields, SLAM using radiance fields, robot control using barrier functions, and planning and control in radiance fields.

Radiance Fields and SLAM. First introduced in [5], Neural Radiance Fields demonstrated the photorealistic novel view rendering capability of radiance field scene representations, where the environment is represented by spatial density and color fields, parameterized by multi-layer perceptrons (MLPs). In addition to detailed scene color modeling, NeRFs produce high-fidelity, dense geometric scene reconstructions making them a potential alternative map representation for SLAM pipelines. Methods like iMAP [21] and NICE-SLAM [22] simultaneously optimize the NeRF network weights and the robot/camera poses. Meanwhile, NeRF-SLAM [23] and NeRFBridge [24] combine existing visual odometry with online NeRF training for real-time 3D scene reconstruction.

GSplat [7], a successor to NeRFs, retains a similar philosophy on the parametric modeling of scene radiance fields. However, rather than using an MLP-based parametric model, GSplat represents the scene using ellipsoidal primitives with a tile-based rasterization technique for differentiable rendering. Compared to NeRFs, GSplat provides superior training speed and reconstruction accuracy. GSplat has also been integrated into SLAM pipelines [20], [25]–[27]. Specifically, we integrate our safety layer with SplatBridge, a GSplat extension of NerfBridge. Both are open-source, ROS integrated software packages for online training of radiance fields from robot sensors.

Control Barrier Functions. CBFs [28] have become a widely utilized tool to synthesize safe controllers for robotic systems. CBFs provide guaranteed safety by ensuring forward set invariance. Typically CBFs are used in unison with a higher-level goal-oriented planner, or a human tele-operator [29]. In this form, CBFs act as a non-invasive *safety filter* [30], [31].

Some works have addressed the use of CBFs to provide safety active sensing of an environment. In [32], [33], CBFs

are formulated for avoiding robot body frame point clouds from depth or LIDAR inputs. These methods do not build a map of the environment and therefore can only avoid obstacles which are currently in view of the sensors. In [34], [35], CBFs are created for occupancy grid based environment maps. While these methods maintain a map of the environment, they lack the scalability and expressiveness of GSplat representations.

Planning and Control in Radiance Fields. Using NeRFs as an underlying scene representation, NeRF-Nav [17] plans trajectories for differentially flat robots, minimizing a collision cost through gradient descent, but has no safety guarantees and is slow to converge. CATNIPS [18] converts the NeRF into a probabilistic voxel grid to plan safe paths parameterized as Bézier curves. Most structurally similar to our work, [19] (which we call NeRF-CBF) uses NICE-SLAM [22] to train a NeRF and uses the rendered depth map at sampled poses to enforce step-wise safety using a CBF. NeRF-CBF requires a pre-trained NICE-SLAM map, due to the slow execution time of online NICE-SLAM. Furthermore, the discrete CBF has weak safety guarantees, and combined with slow NeRF rendering, introduces scalability issues. SAFER-Splat addresses these limitations by performing online mapping using SplatBridge and is amenable to continuous-time dynamics models.

To our knowledge, Splat-Nav [14] is the only work to propose a planning algorithm for a robot using a GSplat map, although we note that prior work on mapping with Gaussian Mixture Models exists [36], [37]. Splat-Nav converts the GSplat representation into a union of ellipsoids and performs rigorous collision checking to generate smooth, safe trajectories. Similarly to Splat-Nav, we leverage the ellipsoidal interpretation of GSplats to demonstrate safety of our safety filter however, our safety filter is meant to intervene at the low level feedback control layer, whereas Splat-Nav provides a high-level trajectory planning layer.

III. 3D GAUSSIAN SPLATTING

Gaussian Splatting (GSplat), recently introduced in [7], provides a photorealistic 3D scene representation of an environment using 3D ellipsoids (Gaussians) as the underlying geometric primitives. Each ellipsoid is assigned spatial and geometric attributes given by a mean $\mu \in \mathbb{R}^3$ and covariance matrix $\Sigma \in \mathbb{S}_{++}$, in addition to visual attributes given by opacity $\alpha \in [0, 1]$, and spherical harmonics (SH) coefficients for view-dependent visual effects. The entire representation (including all the attributes of all ellipsoids) is seeded from sparse point-cloud(s) of the environment, and optimized using a photometric error from monocular images. These point-clouds can be retrieved from structure-from-motion [38] or derived from robot odometry and sensors in online mapping.

GSplat offers faster rendering and training times compared to NeRF-based methods, and more importantly in our use-case, provides more accurate collision geometry, as shown in [14]. To convert the GSplat into an interpretable representation, we leverage the fact that the minimal geometry of the Splat is the 99% confidence ellipsoid for every Gaussian [7], [14]. By maintaining a positive distance to this representation,

a robot is guaranteed to maintain safety with respect to the geometry of the GSplat. Although the resulting collision geometry from GSplat is relatively accurate, we note that a gap still exists a small gap between the ground-truth geometry and that extracted from the GSplat. We compensate for this mismatch when designing our CBF-based controller, for improved robustness.

We now formally present our definition of the scene representation which will be used by the safety layer. A scene consists of a union of K ellipsoids comprising the GSplat $\mathcal{G} = \{\mathcal{E}_i\}_{i=1}^{K}$, where each ellipsoid $\mathcal{E}_i$ satisfies the standard form: $\mathcal{E}_i = \{x \in \mathbb{R}^3 | (x - \mu_i)^T \Sigma_i^{-1}(x - \mu_i) \le 1\}$ for means μ_i and covariance Σ_i. We decompose the covariances based on the rotation and scaling parameters from the GSplat formulation: $\Sigma_i = R\bar{S}\bar{S}^T R^T$, where $\bar{S} = \sqrt{\chi_3^2(\gamma)}S$ represents the scales of Gaussian primitive i multiplied by the $\gamma\%$th percentile of the chi-squared distribution. For the remainder of the text, we will only refer to the $\gamma\%$ confidence ellipsoids (where γ corresponds to 1Σ in this work) and denote S as their semi-major axes.

IV. CONTROL BARRIER FUNCTIONS

CBFs frame safety guarantees in the lens of forward invariance. Given a CBF candidate $h(x) : \mathbb{R}^n \to \mathbb{R}$ of the robot's state $x \in \mathcal{D}$, we define the safe set as the 0-superlevel set $\Omega = \{x \in \mathcal{D} \mid h(x) \ge 0\}$. The $\text{int}(\Omega)$ is defined where $h(x)$ is strictly positive, whereas the boundary $\partial\Omega$ is precisely where $h(x)$ is 0. In order for a safety algorithm to be safe, we must remain in Ω for all time $t \in \mathbb{R}_+$. The forward invariance can be formalized through the condition [39] (an extension of Nagumo's Theorem [40]): $\dot{h}(x(t)) \ge -\alpha(h(x))$, where α is any class-κ extended function. Intuitively, this constraint implies that the CBF candidate is allowed to decrease at an arbitrarily high rate until it reaches ∂S, at which point it can no longer be negative and become unsafe. Conversely, the constraint is also defined for unsafe behavior, in which the CBF is designed to increase $h(x)$ back into Ω.

In order to synthesize a controller, we first define the continuous-time control-affine dynamics: $\dot{x} = f(x) + g(x)u$, where state transition function $f(x) : \mathcal{D} \to \mathbb{R}^n$, actuation matrix $g(x) : \mathcal{D} \to \mathbb{R}^{n \times m}$, and control effort $u \in \mathbb{R}^m$. Due to space limitations, we limit our discussion to double-integrator systems and refer readers to [29] for a more detailed discussion of CBFs. We can define a formal CBF constraint for double-integrator systems: $\forall x \in \mathcal{D}, \exists u :$

$$\left[(\mathcal{L}_f \mathcal{L}_g h)^T u \ge -\mathcal{L}_f^2 h - \alpha(\mathcal{L}_f h) - \beta(\mathcal{L}_f h + \alpha(h)) \right], \tag{1}$$

where $\alpha, \beta > 0$ are positive selected constants. Eq. (1) is affine with respect to the decision variable u. To synthesize a controller, we embed the CBF constraint into a quadratic program of the form:

$$\begin{aligned} &\underset{u}{\text{minimize}} \ ||u - \bar{u}||_2^2 \\ &\text{subject to } Eq. \ (1), \end{aligned} \tag{2}$$

which can be solved quickly to global optimality and $\bar{u}$ denotes the desired control derived from an external source.

V. COLLISION AVOIDANCE IN GSPLAT MAPS

For a feasible integration with GSplat mapping methods, a safety layer must be scalable to the number of Gaussians in a GSplat and be safe at all timesteps.

CBF Formulation. We propose a Euclidean distance metric, the minimum distance between a spherical robot body (centered at position $\mathbf{p}$ with radius r) and an ellipsoid, in contrast to a Mahalanobis distance metric, noting that non-Euclidean distance metrics generally introduce additional computational bottlenecks. Later in this section, we extend the formulation to ellipsoid-ellipsoid inter-distances. Because distances are invariant to translations and rotations, we can convert the sphere-to-ellipsoid problem to a zero-mean, ellipsoid aligned frame, with the center of the ellipsoid at the origin and the primary axes oriented in the $+x, +y, +z$ directions from largest to smallest. Moreover, any reflections across the primary planes (xy, yz, zx) preserve distances due to symmetry of the ellipsoid. We denote variables in this translated, rotated, and reflected frame with a caret (^).

The distance between the ith ellipsoid and the robot centroid $\hat{\mathbf{p}} = FR^T(\mathbf{p} - \mu_i)$ can be posed as the quadratic program:

$$\begin{aligned} d_i(\hat{\mathbf{p}}) = &\underset{\hat{y}}{\text{minimize}} \ ||\hat{\mathbf{p}} - \hat{y}||_2^2 \\ &\text{subject to } \hat{y}^T S^{-2} \hat{y} \le 1, \end{aligned} \tag{3}$$

where F is the reflection matrix, a diagonal matrix of positive or negative 1 used to reflect points across the primary planes. The inclusion of such a matrix will be apparent when we discuss how to solve this program. We note that this program with one constraint always has a feasible solution.

CBF Solution. Instead of solving the optimization through a solver, which is typically not scalable to millions of Gaussians on CPU and often does not provide derivatives, we choose to numerically solve the dual objective

$$\max_{\lambda} \min_{\hat{y}} \underbrace{||\hat{\mathbf{p}} - \hat{y}||_2^2 + \lambda(\hat{y}^T S^{-2} \hat{y} - 1)}_{L(\hat{y}, \lambda; \hat{\mathbf{p}})}. \tag{4}$$

Considering the case where the robot is in the safe set, from the first-order optimality conditions, we have:

$$\begin{aligned} \frac{\partial}{\partial \lambda} L(\hat{y}^\star, \lambda^\star; \hat{\mathbf{p}}) &= \hat{y}^{\star T} S^{-2} \hat{y}^\star - 1 = 0 \\ \frac{\partial}{\partial \hat{y}} L(\hat{y}^\star, \lambda^\star; \hat{\mathbf{p}}) &= 2(\lambda^\star S^{-2} + \mathcal{I})\hat{y}^\star - 2\hat{\mathbf{p}} = 0. \end{aligned} \tag{5}$$

from which we can compute $\hat{y}^*$. For scenarios in which the robot is penetrating the ellipsoid, solving these conditions will yield a non-zero penetration distance, which is crucial for a well-behaved CBF. The optimality conditions are:

$$\begin{aligned} \hat{y}^*(\lambda, \hat{\mathbf{p}}) &= \left(..., \frac{S_{ii}^2 \hat{\mathbf{p}}_i}{\lambda + S_{ii}^2}, ...\right)^T \\ q(\lambda, \hat{\mathbf{p}}) &= \sum_{i=1}^{n} \frac{y_i^{*2}}{S_{ii}^2} - 1 = \sum_{i=1}^{n} \frac{S_{ii}^2 \hat{\mathbf{p}}_i^2}{(\lambda + S_{ii}^2)^2} - 1 = 0, \end{aligned} \tag{6}$$

where $\hat{\mathbf{p}}_i \ge \mathbf{0}$ for some reflection matrix F.

Note that the summation in Eq. (6) is convex and monotonically decreasing (from $+\infty$ tending to 0) with respect to
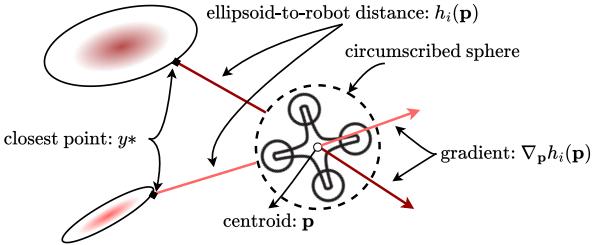
Fig. 2. The CBF is posed as an optimization problem, minimizing the distance between the robot $\mathbf{p}$ and some point y^* on the ellipsoid. We solve the robot-ellipsoid distance program through a bisection search, and return both its gradient and Hessian.

λ for $\lambda \geq \max_i\{-S_{ii}^2\}$. Since $\hat{\mathbf{p}}$ is in the positive orthant, y^* must also be in the positive orthant. Because the expression is monotonic and occupies the range $(0, \infty)$, the equation always has exactly one solution in the feasible range of λ, and can be solved to arbitrary precision using a bisection algorithm [41], parallelized on GPU.

We define the robot-to-ellipsoid CBF as:

$$h_i(\mathbf{p}) = \phi_i(\hat{\mathbf{p}})d_i(\hat{\mathbf{p}}) - (r + \epsilon)^2$$
$$\phi_i(\hat{\mathbf{p}}) = \mathbf{sign}(\hat{\mathbf{p}}^T S^{-2}\hat{\mathbf{p}} - 1), \tag{7}$$

which represents a tractable CBF candidate for spherical robots with radius r and desired buffer distance ϵ navigating within GSplat maps. Note that for $\epsilon = 0$, $h_i(\mathbf{p})$ will be positive in free space, 0 when the robot body just intersects the ellipsoid, and negative when penetrating the ellipsoid. Solving 1 requires the derivatives of the CBF. By the Envelope theorem [42], we can treat the optimized variables as constants when taking the first derivative. Hence,

$$\nabla_{\mathbf{p}}h_i(\mathbf{p}) = \phi_i \frac{\partial L}{\partial \hat{\mathbf{p}}}\frac{\partial \hat{\mathbf{p}}}{\partial \mathbf{p}} = 2\phi_i(\mathbf{p} - y^*)^T, \tag{8}$$

where y^* is the closest ellipsoid point in the original frame.

To compute the Hessian, we make use of the implicit function theorem, which states that partial derivatives of two variables related implicitly by an equation $q(\lambda, \hat{\mathbf{p}}) = 0$ can be expressed as: $\frac{\partial \lambda}{\partial \hat{\mathbf{p}}} = -\left[\frac{\partial q}{\partial \lambda}\right]^{-1}\frac{\partial q}{\partial \hat{\mathbf{p}}}$. This theorem allows the expression of the Hessian as $\nabla_{\mathbf{p}}^2 h_i(p) = \mathbf{H}_i$, where:

$$\mathbf{H}_i = 2\phi_i(\mathbf{I} - \frac{\partial \hat{y}^*(\hat{\mathbf{p}})}{\partial \hat{\mathbf{p}}})$$
$$= 2\phi_i \left[\mathbf{I} - \frac{\partial \hat{y}^*(\lambda, \hat{\mathbf{p}})}{\partial \hat{\mathbf{p}}} + \frac{\partial \hat{y}^*(\lambda, \hat{\mathbf{p}})}{\partial \lambda}\frac{\partial \lambda}{\partial \hat{\mathbf{p}}}\right] \tag{9}$$
$$= 2\phi_i \left[\mathbf{I} - \mathbf{diag}(\frac{S_{ii}^2}{\lambda + S_{ii}^2}) + \frac{\partial \lambda}{\partial q}\frac{\partial \hat{y}^*}{\partial \lambda}\frac{\partial q}{\partial \hat{\mathbf{p}}}\right]$$

where $\frac{\partial q}{\partial \lambda} = -2\sum_{i=1}^n \frac{S_{ii}^2\hat{\mathbf{p}}_i^2}{(\lambda + S_{ii}^2)^3}$, $\frac{\partial \hat{y}^*}{\partial \lambda} = -\left[\frac{S_{ii}^2\hat{\mathbf{p}}_i}{(\lambda + S_{ii}^2)^2}\right]^T_{\forall i}$, and $\frac{\partial q}{\partial \hat{\mathbf{p}}} = 2\left[\frac{S_{ii}^2\hat{\mathbf{p}}_i}{(\lambda + S_{ii}^2)^2}\right]_{\forall i}$. Fig. 2 visualizes the CBF gradients.

Proposition 1. *We can extend the sphere-to-ellipsoid CBF to an ellipsoid-to-ellipsoid CBF, which is applicable to non-spherical robots, by defining a diffeomorphism $\psi : \mathbb{O} \to \mathbb{W}$ that maps the original manifold $\mathbb{O}$ to an equivalent manifold $\mathbb{W}$ where the robot is spherical. We can utilize the sphere-to-ellipsoid CBF in $\mathbb{W}$ and map the control back to $\mathbb{O}$.*

Proof. First, assume that the robot geometry is an ellipsoid $\mathcal{E}_r$ in the original 3D space $\mathbb{O}$, with body-to-world rotation matrix R_r, scaling matrix S_r and centroid $\mathbf{p}$, and the scene ellipsoid's $\mathcal{E}_i$ is as previously defined. By defining the diffeomorphism $\psi : \mathbb{O} \to \mathbb{W}$, where $\psi(x) = \bar{S}^{-1}R_r^T x$, we transform the robot from an ellipsoid in $\mathbb{O}$ to a unit sphere in $\mathbb{W}$. We transform all the ellipsoids representing the environment from $\mathbb{O}$ to $\mathbb{W}$, with the covariance of ellipsoid i given by: $\Sigma_i^{\mathbb{W}} = S_r^{-1}R_r R_i S_i S_i^T R_i^T R_r^T S_r^{-T}$. An eigendecomposition is performed on $\Sigma_i^{\mathbb{W}}$ to retrieve $R_i^{\mathbb{W}}$ and $S_i^{\mathbb{W}}$. Then, we can use our sphere-to-ellipsoid CBF Eq. (7) in $\mathbb{W}$, making sure our dynamics are also expressed in $\mathbb{W}$. Specifically, our double-integrator dynamics are unchanged between $\mathbb{O}$ and $\mathbb{W}$. We only need to ensure that the desired acceleration $\bar{u}^{\mathbb{W}}$ is expressed in $\mathbb{W}$, while the outgoing filtered $u^{\mathbb{O}}$ is expressed in $\mathbb{O}$. $\square$

Controller Synthesis and Computational Efficiency. In this work, we utilize the double-integrator model, which is often used in CBF literature for drone dynamics [32], [43], [44]:

$$\dot{x} = \begin{pmatrix} \dot{\mathbf{p}} \\ \dot{\mathbf{v}} \end{pmatrix} = \begin{pmatrix} \mathbf{v} \\ \mathbf{a} \end{pmatrix} = \underbrace{\begin{pmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}\begin{pmatrix} \mathbf{p} \\ \mathbf{v} \end{pmatrix}}_{f} + \underbrace{\begin{pmatrix} \mathbf{0} \\ \mathbf{I} \end{pmatrix}}_{g} u, \tag{10}$$

where u is the commanded acceleration. This assumption is not restrictive, given that drones typically have a lower-level attitude-stabilizing controller that enables the drone to track double-integrator dynamics. Moreover, our CBF can be generalized to a higher-fidelity dynamics models for drones, as demonstrated in [30], [45].

The CBF constraint for double-integrator dynamics can be formed using (1), which can be embedded into a minimally-invasive quadratic program (2). We prune the set of constraints in (2) for better efficiency, by reasoning about the dynamic feasibility of the system. The redundant constraints are never active, so their exclusion does not compromise safety. This type of pruning yields more savings in denser GSplats.

Theorem 1. *The quadratic program in (2) is always feasible, i.e., a feasible control input u always exist for the CBF controller (2), for robots with double-integrator dynamics, provided that $\mathbf{p} \in \Omega$. Consequently, the safe set is forward invariant, for the closed-loop system.*

Proof. Given the dynamical model (10), we compute the following Lie derivatives:

$$\mathcal{L}_f h_i = (\nabla_{\mathbf{p}}h_i)^T v$$
$$\mathcal{L}_f^2 h_i = f^T\left[\nabla_x^2 h_i\right]f + (\nabla_x h_i)^T\left[\frac{\partial f}{\partial x}\right]f = v^T\mathbf{H}_i v$$
$$\mathcal{L}_f\mathcal{L}_g h_i = f^T\left[\nabla_x^2 h_i\right]g + (\nabla_x h_i)^T\left[\frac{\partial f}{\partial x}\right]g$$
$$= (\nabla_{\mathbf{p}}h_i)^T = 2\phi_i(\mathbf{p} - y^*).$$

We can write (1) as the following half-space constraints:

$$2\phi_i(\mathbf{p} - y^*)^T(u + (\alpha + \beta)v) \geq -v^T\mathbf{H}_i v - \alpha\beta h_i. \tag{11}$$
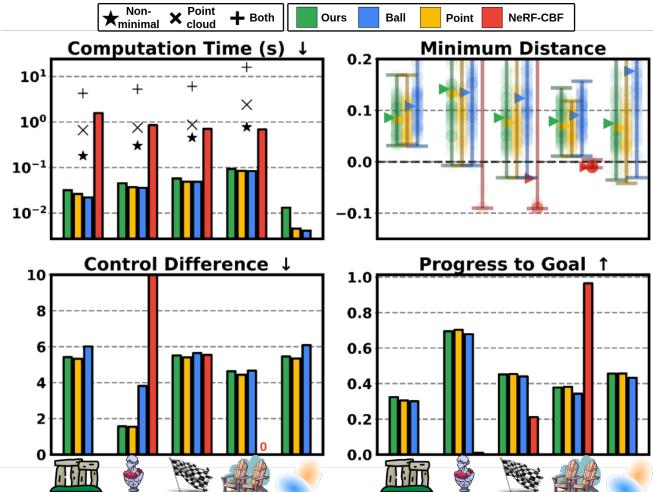
Fig. 3. Computation time (lower is better), minimum distance to collision (above zero is safe), magnitude of control effort correction (lower is better), and progress to goal (higher is better) for five different scenes indicated by the icons. SAFER-Splat (ours, green), is compared with our CBF on a conservative bounding sphere of each ellipsoid (blue), our CBF on the GSplat means (yellow), and NeRF-CBF [19] (red). Each trajectory is visualized as a translucent dot in the minimum distance and the wedge indicates the mean.

Note that for all safe states x such that $h_i(x) > 0$, $\phi_i = 1$ and $\frac{S_{i_i}^2}{\lambda + S_{i_i}^2} < 1$. Moreover, $\frac{\partial \lambda}{\partial q} \frac{\partial y^*}{\partial \lambda} \frac{\partial q}{\partial \mathbf{P}}$ is of the form aBB^T where $a \in \mathbb{R}_{++}$, forming a PSD matrix. $\mathbf{H}_i$ is positive-definite, hence the RHS (11) is always negative and $u = -(\alpha + \beta)v$ is a feasible solution. $\qquad \square$

Finding a closed-form feasible point further allows us to avoid the costly routine of finding interior points, which is required to prune polytope constraints. When the robot is in collision, the corresponding constraints (11) do not necessarily contain the origin. However, there are typically very few ellipsoids in collision at any one time. Therefore, we are free to prune constraints for ellipsoids that are not in collision and keep all in-collision constraints. Thus, collision and collision-free scenarios have approximately the same computation savings, which amounts to roughly pruning 99.9% of Gaussians.

VI. EXPERIMENTS

Simulation Results. We compare SAFER-Splat (green) to NeRF-CBF [19] (red), in addition two variants of our method: one which uses a conservative bounding sphere of each ellipsoid (blue) and one that uses the GSplat means (yellow) in Fig. 3. A Mahalanobis distance variant was also tested but proved too ill-behaved to show. NeRF-CBF is trained using Nerfacto [46] on the same dataset as the GSplat. We examine each method in five scenes: *Stonehenge* (100K Gaussians), *Statues* (200K), and *Flightgate* (300K) (from [14]); in addition to *Adirondacks* (500K), and *low-res Flightgate* (15K). All controllers are executed for 100 trajectories. Our method strikes a favorable trade-off compared to other methods.

The computation time of SAFER-Splat and its variants are similar and more than an order of magnitude faster than NeRF-CBF. We also plot timing results without constraint pruning (∗), on a point cloud (about $6\times$ more points than

ellipsoids) represented by $\times$, and a combination of both ($+$), demonstrating that our real-time performance is attributed to both constraint pruning and the sparse GSplat representation.

To assess safety, we compute the squared distance of the trajectories to the high-res GSplat 7 (instead of COLMAP due to the better dense reconstruction from the GSplat). NeRF-CBF attains major violations (data points less than 0) compared to our method and its variants. Our method and its variants can become unsafe due to dynamics noise, but infrequently. We expect that in sparse reconstructions with large, elongated Gaussians, e.g., in the early stages of GSplat mapping, our method strikes a promising tradeoff between the conservativeness of the ball variant and the high potential for collision of the means-only variant. We see this behavior beginning to appear in *low-res Flightgate* (15K Gaussians).

We utilize the objective of 2 (labeled Control Difference) and progress to the goal ($\min_t ||\mathbf{p}(t) - \mathbf{p}_0||/||\mathbf{p}_f - \mathbf{p}_0||$) as proxies for conservativeness. We do not expect the progress to be 1 because all methods are reactive and utilize a simple PD controller to navigate to the goal. We observe that NeRF-CBF is sporadic in its performance, with it failing completely on *Stonehenge* and yielding only a few valid trajectories in *Statues* and *Flightgate*. Again, our method and the point variant achieve similar results, which are slightly less conservative than the ball variant.

Hardware Platform. In our hardware experiments, we use a 220 mm Starling 2 drone. The platform maintains a pose estimate using visual inertial odometry (VIO) and can track position, velocity, and acceleration commands with a low-level PX4 controller. The camera images, point clouds, and VIO poses are sent through ROS2 [47] to an off-board desktop computer with RTX 4090 GPU via 5 GHz WiFi. The high-level drone acceleration commands come from a teleoperator and are relayed to the off-board computer which transmits CBF corrected commands back to the drone's low-level controller. SAFER-Splat's ROS nodes are made available in our codebase. Note that the scenes used in the real-world experiments contain visual fiducials (ArUco) and motion capture cameras, but these are *not* used by our VIO or GSplat mapping modules for pose estimation.

SplatBridge. The incoming stream of RGB images, time-of-flight point clouds, and estimated poses are used to train a GSplat in real-time. We utilize an extension of NerfBridge [24], called SplatBridge, to interface with the GSplat implementation in Nerfstudio [46]. At every keyframe, SplatBridge loads the image into a running buffer and seeds Gaussian primitives from the ToF point-cloud. Asynchronous to the sensor stream, SplatBridge optimizes the Gaussian attributes and keyframe camera poses, initialized at the VIO estimates, minimizing the loss proposed by [7]. The SAFER-Splat safety layer updates its reference to the Gaussian primitives from SplatBridge before each query to the CBF.

Hardware Experiments. In three real-world experiments, SAFER-Splat successfully enabled safe teleoperation even when the operator repeatedly attempted to force a collision by driving the drone towards obstacles. Fig. 4 shows our sample
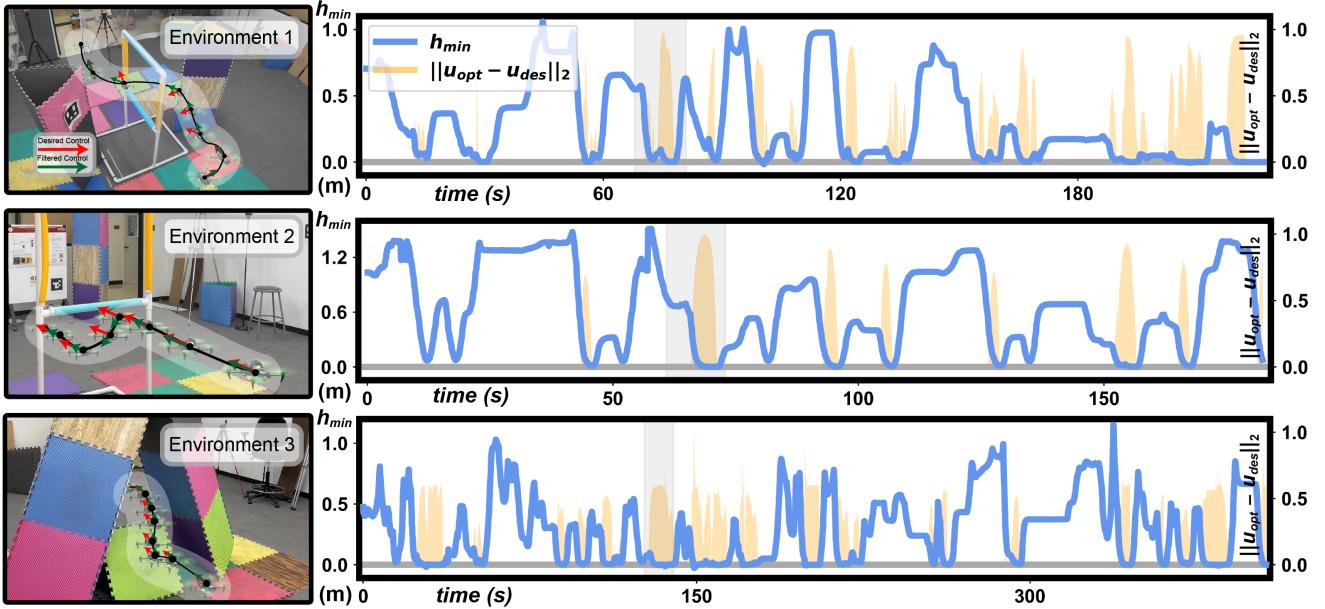
Fig. 4.   We showcase SAFER-Splat across three real-world scenes. A snapshot of the drone trajectory is shown on the left panel for each of the three scenes, annotated with the approximate desired and filtered control directions. In each of these trajectories, the operator attempted to directly hit the obstacle. On the right, we show the full flight trajectory, showing robustness with repeated collision avoidance. The pictured portion of each trajectory is highlighted in grey.
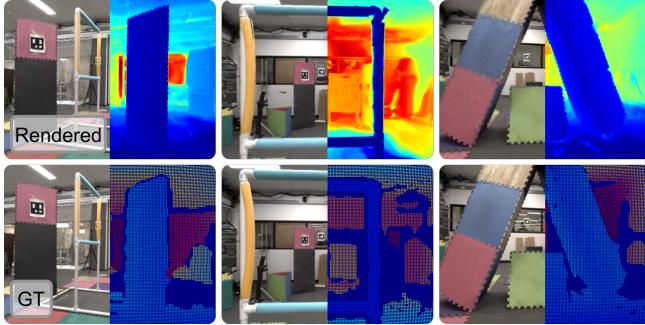

Fig. 5.   Top row: Overlayed color and depth renders from the GSplats produced using SAFER-Splat in the real-world experiments. Bottom row: Overlayed corresponding keyframe color image and the sparse ToF PC.

VII. Conclusion

We propose a real-time 3D reconstruction and safe robot action filtering pipeline called SAFER-Splat for GSplats, which are photo-realistic, efficient to optimize, and interpretable. The proposed safety layer utilizes a CBF formulated for ellipsoidal robot geometries, and leverages the favorable properties of the GSplat representation for scalable and rigorous safety guarantees. SAFER-Splat is generalizable to arbitrary high-level planners, minimally-invasive, and well-behaved, resulting in smooth robot motion. The proposed safety layer is coupled with an online GSplat mapping module to enable safe navigation in unknown environments. In simulations, we show that SAFER-Splat out-performs comparable methods based on NeRFs, and in real-world experiments, we demonstrate that SAFER-Splat can simultaneously optimize a 3D reconstruction and perform as a safe action filter for a teleoperated drone.

VIII. Limitations and Future Work

SAFER-Splat assumes that the underlying GSplat map is relatively accurate with respect to the real-world, which does not hold in some situations, especially with dynamic objects. Future work will seek to improve SAFER-Splat in these cases by accounting for dynamic objects in an uncertainty-aware optimization framework. Other future work will aim to improve the robustness of SplatBridge to inaccurate estimated camera poses, which could affect the accuracy of the GSplat.

We also aim in future work to extend SAFER-Splat to semantic mapping and semantic-aware safety. Furthermore, the generalizability of SAFER-Splat to arbitrary high-level planners enables our method to act as a safety filter for, e.g., AI-based Vision-Language-Action (VLA) models.

trajectories from each scene along with the full plots of our safety value versus time, demonstrating that the safety layer avoids collision ($h_{min} > 0$) and only turns on close to the boundary (i.e., minimally invasive). Our method performed online optimization on the parameters of between 150,000 and 300,000 Gaussians in each scene while simultaneously computing safety filtered actions at 15Hz. We constructed our GSplat maps entirely while the drone was flying, and our CBF enabled after a brief warm-up period (5-30 seconds of flight). During flights in Environments 1 and 3, there were brief violations of the CBF, $h_{min} < 0$, in all cases these violations were on the order of 1-4mm and due to minor failures of the drone's low-level controller in accurately tracking the safety layer's outputs. Likely these tracking failures were caused by rotor induced aerodynamic effects when close to obstacles, and in no cases did they result in loss of control of the drone. Full recordings of each of these real-world flights are available on our project's website. Fig. 5 showcases the high visual accuracy of each environment's GSplat when trained via SplatBridge.

REFERENCES

[1] J. Ryde and H. Hu, "3d mapping with multi-resolution occupied voxel lists," *Autonomous Robots*, vol. 28, pp. 169–185, 2010.

[2] P. Kim, J. Chen, and Y. K. Cho, "Slam-driven robotic mapping and registration of 3d point clouds," *Automation in Construction*, vol. 89, pp. 38–48, 2018.

[3] M. Palieri, B. Morrell, A. Thakur, K. Ebadi, J. Nash, A. Chatterjee, C. Kanellakis, L. Carlone, C. Guaragnella, and A.-a. Agha-Mohammadi, "Locus: A multi-sensor lidar-centric solution for high-precision odometry and 3d mapping in real-time," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 421–428, 2020.

[4] K. M. Jatavallabhula, A. Kuwajerwala, Q. Gu, M. Omama, T. Chen, A. Maalouf, S. Li, G. Iyer, S. Saryazdi, N. Keetha *et al.*, "Conceptfusion: Open-set multimodal 3d mapping," *arXiv preprint arXiv:2302.07241*, 2023.

[5] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," in *European Conference on Computer Vision (ECCV)*, 2020.

[6] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla, and P. P. Srinivasan, "Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 5855–5864.

[7] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3D Gaussian splatting for real-time radiance field rendering," *ACM Transactions on Graphics*, vol. 42, no. 4, July 2023. [Online]. Available: https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/

[8] Z. Yu, A. Chen, B. Huang, T. Sattler, and A. Geiger, "Mip-splatting: Alias-free 3d gaussian splatting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 19 447–19 456.

[9] A. Swann, M. Strong, W. K. Do, G. S. Camps, M. Schwager, and M. Kennedy III, "Touch-gs: Visual-tactile supervised 3d gaussian splatting," *arXiv preprint arXiv:2403.09875*, 2024.

[10] G. Lu, S. Zhang, Z. Wang, C. Liu, J. Lu, and Y. Tang, "Manigaussian: Dynamic gaussian splatting for multi-task robotic manipulation," *arXiv preprint arXiv:2403.08321*, 2024.

[11] Y. Zheng, X. Chen, Y. Zheng, S. Gu, R. Yang, B. Jin, P. Li, C. Zhong, Z. Wang, L. Liu *et al.*, "Gaussiangrasper: 3d language gaussian splatting for open-vocabulary robotic grasping," *arXiv preprint arXiv:2403.09637*, 2024.

[12] O. Shorinwa, J. Tucker, A. Smith, A. Swann, T. Chen, R. Firoozi, M. D. Kennedy, and M. Schwager, "Splat-mover: Multi-stage, open-vocabulary robotic manipulation via editable gaussian splatting," in *8th Annual Conference on Robot Learning*, 2024.

[13] J. Abou-Chakra, K. Rana, F. Dayoub, and N. Sünderhauf, "Physically embodied gaussian splatting: A realtime correctable world model for robotics," in *8th Annual Conference on Robot Learning*, 2024.

[14] T. Chen, O. Shorinwa, J. Bruno, J. Yu, W. Zeng, K. Nagami, P. Dames, and M. Schwager, "Splat-nav: Safe real-time robot navigation in gaussian splatting maps," 2024. [Online]. Available: https://arxiv.org/abs/2403.02751

[15] L. Díaz-Vilariño, P. Boguslawski, K. Khoshelham, H. Lorenzo, and L. Mahdjoubi, "Indoor navigation from point clouds: 3d modelling and obstacle detection," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 41, pp. 275–281, 2016.

[16] F. Gao, W. Wu, W. Gao, and S. Shen, "Flying on point clouds: Online trajectory generation and autonomous navigation for quadrotors in cluttered environments," *Journal of Field Robotics*, vol. 36, no. 4, pp. 710–733, 2019.

[17] M. Adamkiewicz, T. Chen, A. Caccavale, R. Gardner, P. Culbertson, J. Bohg, and M. Schwager, "Vision-only robot navigation in a neural radiance world," *IEEE Robotics and Automation Letters (RA-L)*, vol. 7, no. 2, pp. 4606–4613, 2022.

[18] T. Chen, P. Culbertson, and M. Schwager, "Catnips: Collision avoidance through neural implicit probabilistic scenes," *arXiv preprint arXiv:2302.12931*, 2023.

[19] M. Tong, C. Dawson, and C. Fan, "Enforcing safety for vision-based controllers via control barrier functions and neural radiance fields," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 10 511–10 517.

[20] C. Yan, D. Qu, D. Wang, D. Xu, Z. Wang, B. Zhao, and X. Li, "GS-SLAM: Dense visual slam with 3d gaussian splatting," *arXiv preprint arXiv:2311.11700*, 2023.

[21] E. Sucar, S. Liu, J. Ortiz, and A. Davison, "iMAP: Implicit mapping and positioning in real-time," in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

[22] Z. Zhu, S. Peng, V. Larsson, W. Xu, H. Bao, Z. Cui, M. R. Oswald, and M. Pollefeys, "Nice-SLAM: Neural implicit scalable encoding for SLAM," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022.

[23] A. Rosinol, J. J. Leonard, and L. Carlone, "NeRF-SLAM: Real-Time Dense Monocular SLAM with Neural Radiance Fields," *arXiv preprint arXiv:2210.13641*, 2022.

[24] J. Yu, J. E. Low, K. Nagami, and M. Schwager, "Nerfbridge: Bringing real-time, online neural radiance field training to robotics," *arXiv preprint arXiv:2305.09761*, 2023.

[25] V. Yugay, Y. Li, T. Gevers, and M. R. Oswald, "Gaussian-SLAM: Photo-realistic dense slam with gaussian splatting," *arXiv preprint arXiv:2312.10070*, 2023.

[26] H. Matsuki, R. Murai, P. H. Kelly, and A. J. Davison, "Gaussian splatting slam," *arXiv preprint arXiv:2312.06741*, 2023.

[27] N. Keetha, J. Karhade, K. M. Jatavallabhula, G. Yang, S. Scherer, D. Ramanan, and J. Luiten, "Splatam: Splat track & map 3d gaussians for dense rgb-d slam," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 21 357–21 366.

[28] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for safety critical systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 3861–3876, 2017.

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

[30] A. Singletary, A. Swann, Y. Chen, and A. D. Ames, "Onboard safety guarantees for racing drones: High-speed geofencing with control barrier functions," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2897–2904, 2022.

[31] A. Singletary, A. Swann, I. D. J. Rodriguez, and A. D. Ames, "Safe drone flight with time-varying backup controllers," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 4577–4584.

[32] M. De Sa, P. Kotaru, and K. Sreenath, "Point cloud-based control barrier function regression for safe and efficient vision-based control," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. Yokohama, Japan: IEEE, May 2024, p. 366–372. [Online]. Available: https://ieeexplore.ieee.org/document/10610647/

[33] B. Dai, R. Khorrambakht, P. Krishnamurthy, and F. Khorrami, "Sailing through point clouds: Safe navigation using point cloud based control barrier functions," *IEEE Robotics and Automation Letters*, vol. 9, no. 9, pp. 7731–7738, 2024.

[34] S. Zhou, S. Papatheodorou, S. Leutenegger, and A. P. Schoellig, "Control-barrier-aided teleoperation with visual-inertial slam for safe mav navigation in complex environments," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. Yokohama, Japan: IEEE, May 2024, p. 17836–17842. [Online]. Available: https://ieeexplore.ieee.org/document/10611280/

[35] G. Raja, T. Mökkönen, and R. Ghabcheloo, "Safe control using occupancy grid map-based control barrier function (ogm-cbf)," 2024. [Online]. Available: https://arxiv.org/abs/2405.10703

[36] K. Goel, W. Tabib, and N. Michael, "Rapid and high-fidelity subsurface exploration with multiple aerial robots," in *Experimental Robotics: The 17th International Symposium*. Springer, 2021, pp. 436–448.

[37] K. Goel, N. Michael, and W. Tabib, "Probabilistic point cloud modeling via self-organizing gaussian mixture models," *IEEE Robotics and Automation Letters*, vol. 8, no. 5, pp. 2526–2533, 2023.

[38] J. L. Schönberger and J.-M. Frahm, "Structure-from-motion revisited," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[39] P. Wieland and F. Allgöwer, "Constructive safety using control barrier functions," *IFAC Proceedings Volumes*, vol. 40, no. 12, pp. 462–467, 2007.

[40] M. Nagumo, "Über die lage der integralkurven gewöhnlicher differentialgleichungen," *Proceedings of the Physico-Mathematical Society of Japan. 3rd Series*, vol. 24, pp. 551–559, 1942.

[41] D. Eberly, "Distance from a point to an ellipse, an ellipsoid, or a hyperellipsoid," 2013, accessed: 2024-09-08. [Online]. Available: https://www.geometrictools.com/Documentation/DistancePointEllipseEllipsoid.pdf

[42] E. Silberberg, "The viner–wong envelope theorem," *The Journal of Economic Education*, vol. 30, no. 1, pp. 75–79, 1999.

[43] V. M. Gonçalves, P. Krishnamurthy, A. Tzes, and F. Khorrami, "Control barrier functions with circulation inequalities," *IEEE Transactions on Control Systems Technology*, vol. 32, no. 4, pp. 1426–1441, 2024.

[44] A. Singletary, K. Klingebiel, J. Bourne, A. Browning, P. Tokumaru, and A. Ames, "Comparative analysis of control barrier functions and artificial potential fields for obstacle avoidance," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 8129–8136.

[45] 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*, pp. 1–1, 2024.

[46] M. Tancik, E. Weber, R. Li, B. Yi, T. Wang, A. Kristoffersen, J. Austin, K. Salahi, A. Ahuja, D. McAllister, A. Kanazawa, and E. Ng, "Nerfstudio: A framework for neural radiance field development," in *SIGGRAPH*, 2023.

[47] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot operating system 2: Design, architecture, and uses in the wild," *Science robotics*, vol. 7, no. 66, p. eabm6074, 2022.