2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Formal Safety Verification and Refinement for Generative Motion Planners via Certified Local Stabilization

Devesh Nath* Haoran Yin* Glen Chou

***Abstract*—We present a method for formal safety verification of learning-based generative motion planners. Generative motion planners (GMPs) offer advantages over traditional planners, but verifying the safety and dynamic feasibility of their outputs is difficult since neural network verification (NNV) tools scale only to a few hundred neurons, while GMPs often contain millions. To preserve GMP expressiveness while enabling verification, our key insight is to imitate the GMP by stabilizing references sampled from the GMP with a small neural tracking controller and then applying NNV to the closed-loop dynamics. This yields reachable sets that rigorously certify closed-loop safety, while the controller enforces dynamic feasibility. Building on this, we construct a library of verified GMP references and deploy them online in a way that imitates the original GMP distribution whenever it is safe to do so, improving safety without retraining. We evaluate across diverse planners, including diffusion, flow matching, and vision-language models, improving safety in simulation (on ground robots and quadcopters) and on hardware (differential-drive robot).**

I. Introduction

Motion planning has been transformed by generative models like diffusion and conditional flow matching (CFM) [1], [2], which learn multimodal trajectory distributions and enable generative motion planners (GMPs) that produce diverse plans from inputs like language or images [3]–[6]. However, ensuring that GMP-generated trajectories satisfy safety and dynamic feasibility is difficult: GMPs often contain millions of parameters, making neural network verification (NNV) [7] intractable, limiting their use in safety-critical settings [6]. NNV provides hard guarantees via set-based reachability but only scales to controllers with a few hundred neurons [8]–[11]. More scalable statistical methods [3], [12]–[14] yield weaker probabilistic guarantees or require prohibitive samples over long horizons. Thus, existing work trades off between expressive large models lacking hard guarantees and small verifiable models unable to capture complex behaviors.

To bridge this gap, we propose **SaGe-MP** (Safe Generative Motion Planning), a method that provides hard safety and dynamic feasibility guarantees for GMP-generated motion plans. Our key insight is that while NNV tools cannot directly certify the GMP, they can certify a *small* neural *tracking controller* that locally stabilizes the system around GMP-sampled references. We use such a controller in our method, yielding closed-loop dynamics that are readily verifiable via NNV. Reachability analysis of the resulting closed-loop system yields hard assurances of safety and dynamic feasibility over a continuum of inputs. Here, the GMP acts only as an open-loop plan generator, while verification is performed on the closed-loop dynamics induced by tracking a *fixed* GMP plan, resulting in a smaller computational graph that makes NNV tractable. By tracking GMP references under the true dynamics, the controller also projects potentially dynamically-infeasible GMP plans onto feasible trajectories. To preserve the original GMP behavior if possible, we develop a trajectory-library approach: multiple GMP references are sampled offline, certified as safely trackable via NNV, and deployed online in a way that mimics the potentially multimodal GMP output. In this sense, our method is a lightweight GMP refinement that enhances safety and dynamic feasibility without costly GMP retraining. Our contributions are:

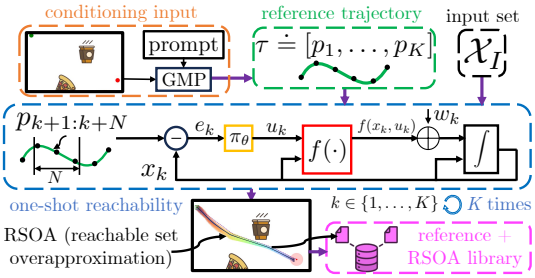

Fig. 1: An overview of our method, which samples GMP references and stabilizes them with a small neural tracking controller (Sec. V-A, verifies safety via NNV-based reachability (Sec. V-B-V-C), and imitates the original GMP with a trajectory library (Sec. V-D).

1) An NNV-based method for formal safety verification of large GMPs that decouples trajectory generation from neural feedback loop verification, preserving planner expressiveness while providing hard safety guarantees.
2) A method to certify a neural trajectory tracking controller that stabilizes dynamically-infeasible GMP references, producing safe, feasible trajectories.
3) A trajectory-library method that stores certified-safe GMP plans and executes them online in a way that preserves the original behavior whenever safe. We prove sample complexity bounds for a target imitation error.
4) Extensive simulation and real-world validation demonstrating safe stabilization of references from diverse generative models (diffusion, flow matching, VLMs, neural ODEs) on challenging nonlinear systems (e.g., 12D quadcopter, and learned NN dynamics).

For space, we provide further details in the appendix of [15].

II. Related Work

A. Generative Motion Planners (GMPs)

GMPs generate diverse, human-like trajectories from rich inputs such as images or language [2] and can be trained via imitation learning, unlike traditional trajectory optimization or sampling-based planning [16], which are limited to low-dimensional inputs. They also capture multimodal trajectory distributions, representing distinct homotopy classes that serve as a diverse set of solutions to the task.

However, ensuring safety of GMP-generated trajectories is difficult. Existing methods encourage safety with planning

The authors are with the Georgia Institute of Technology, Atlanta, GA, USA. {dnath7, hyin95, chou}@gatech.edu

costs [4] or use control barrier functions (CBFs) as corrections [1], [5], [17], but often assume simplified dynamics (e.g., single integrator). For nonlinear robot dynamics [18], it is difficult to find CBFs; neural CBFs can be learned for such systems, but only heuristically encourage safety [13]. In contrast, we provide hard guarantees for nonlinear NN dynamics by overriding unsafe GMP plans with a certified tracking controller that stabilizes around a nearby safe GMP plan. Enforcing dynamic feasibility of GMP plans is also difficult. Predicting action sequences [19] ensures feasibility but yields discontinuous outputs that GMPs predict poorly [20]. Some methods enforce feasibility on linear systems [21]. Others project GMP references onto the manifold of feasible plans under nonlinear dynamics [6] but do not guarantee feasibility and have high replanning cost under model error. Our method instead stabilizes GMP references with a tracking controller, projecting them onto the dynamically-feasible manifold and rejecting errors.

B. Verification of NN Controllers

Formal methods. Reachability analysis verifies that system trajectories remain in a safe set [22]. Set-based methods propagate input and disturbance sets to compute reachable set overapproximations (RSOAs). Methods based on sum-of-squares optimization [23], [24] assume polynomial dynamics and analytical controllers to perform reachability around motion plans. In contrast, for the NN controllers that we consider in this work, abstraction tools like CROWN [7] provide affine bounds on outputs and enable RSOA computation [8], [11], [25]. As the NN size grows, however, the computed RSOAs become highly conservative, revealing a tradeoff between model expressiveness and verification tractability. Thus, NNV-based reachability is limited to small NN controllers (hundreds of neurons) [8]. Symbolic and one-shot reachability [8], [25] reduces conservativeness [9], but remain limited by NN size. Direct verification of GMPs, with millions of neurons, is intractable. We bridge this gap by stabilizing GMP samples with a small, verified tracking controller, preserving GMP expressiveness while certifying safety over a continuum of initial states.

Statistical methods. Statistical methods that bound RSOA soundness probabilities scale better but give weaker guarantees. Conformal prediction (CP) [12] ensures reachable set coverage under an exchangeability assumption between the training and test distributions [26], which feedback-controlled, time-varying data often violates. Adaptive CP relaxes this but only offers weaker long-run average coverage assurances. Simulation-based methods [14], [27], [28] approximate reachable sets probabilistically, but sample complexity grows quickly with horizon length. In contrast, our method provides hard guarantees robust to distribution shifts, scaling to GMP references hundreds of timesteps long.

III. Preliminaries

We consider discrete-time nonlinear dynamical systems

$$x_{k+1} = f(x_k, u_k, w_k), \tag{1}$$

where $k \in \{1, \dots, K\}$ is the discrete timestep. The terms $x_k \in \mathcal{X} \subseteq \mathbb{R}^n$, $u_k \in \mathcal{U} \subseteq \mathbb{R}^m$, and $w_k \in \mathcal{W} \subseteq \mathbb{R}^d$ represent the state, control input, and an external disturbance, respectively. We assume that the initial state x_I lies within a known set $\mathcal{X}_I \subseteq \mathcal{X}$ and that the disturbance w_k is drawn from a compact set $\mathcal{W} \doteq \{w \mid \|w\|_\infty \leq \overline{w}\}$, where $\overline{w}$ is known *a priori*, though we note that these bounds can also be estimated from data [29], [30]. Denote $\mathsf{Int}(\underline{a}, \overline{a}) = \{a \mid \underline{a} \leq a \leq \overline{a}\}$, as a hyper-rectangular interval, where $\underline{a}, \overline{a} \in \mathbb{R}^A$ and the inequalities $\leq$ are interpreted element-wise, i.e., $\underline{a}_i \leq a_i \leq \overline{a}_i$, $1 \leq i \leq A$. Denote the a-ball as $\mathcal{B}_a(c) \doteq \{x \mid \|x - c\|_\infty \leq a\}$.

A. Computational Graph Robustness Verification

A computational graph (CG) is a directed acyclic graph representing the sequence of mathematical operations on an input [8], such as a neural network or the nonlinear function (1). For a graph G with input set $\mathcal{Z} \subseteq \mathbb{R}^{n_i}$ and output $G(\mathcal{Z}) \subseteq \mathbb{R}^{n_o}$, we can compute a guaranteed overapproximation of $G(\mathcal{Z})$ using CROWN-based [7] tools like `auto_LiRPA` [31]. These tools provide guaranteed affine lower and upper bounds, $\underline{G}$ and $\overline{G}$, on the output $G(\mathcal{Z})$ for any hyper-rectangular input set $\mathcal{Z}$. This is formalized in the following proposition from [8].

Proposition 1 (CG Robustness [31])**.** *For CG G and hyper-rectangular $\mathcal{Z} \doteq \{z \in \mathbb{R}^{n_i} \mid \underline{z} \leq z \leq \overline{z}\}$, there are affine functions $\underline{G}$ and $\overline{G}$ such that for all $z \in \mathcal{Z}$, $\underline{G}(z) \leq G(z) \leq \overline{G}(z)$. The inequalities hold element-wise and $\underline{G}(z) = \Psi z + \alpha$, $\overline{G}(z) = \Phi z + \beta$, with $\Psi, \Phi \in \mathbb{R}^{n_o \times n_i}$ and $\alpha, \beta \in \mathbb{R}^{n_o}$.*

B. Reachability Analysis

Reachability analysis computes the set of all states that a dynamical system can reach from given initial conditions, under admissible controls and disturbances. For brevity, the policy is written as $u_k = \pi_\theta(x_k, \eta_k)$, where θ are trainable parameters and $\eta_t \in \mathcal{I}$ are external conditioning inputs, e.g., a desired reference trajectory. The closed-loop dynamics are

$$x_{k+1} = f(x_k, \pi_\theta(x_k, \eta_k), w_k) \doteq \tilde{f}(x_k), \tag{2}$$

where the dependence on control and noise is implicit in (2).

Exact Reachability: We define the exact reachable set of $\tilde{f}$ at time $k+1$, assuming the state at time k lies in the hyper-rectangular set $\mathcal{X}_k \doteq \{x \mid \underline{x}_k \leq x \leq \overline{x}_k\}$ as:

$$\begin{aligned} R_{k+1}(\mathcal{X}_k \mid \eta_k) &= \{x \mid x = \tilde{f}(x_k),\ x_k \in \mathcal{X}_k\} \\ &= \left\{ x \;\middle|\; \begin{array}{l} x = f(x_k, \pi_\theta(x_k, \eta_k), w_k), \\ x_k \in \mathcal{X}_k,\ w_k \in \mathcal{W} \end{array} \right\}. \end{aligned} \tag{3}$$

Since computing exact reachable sets for nonlinear NN-controlled systems is intractable [22], we compute reachable set over-approximations (RSOAs) $\hat{R} \supseteq R$ using NNV tools [7] and applying Proposition 1. We denote the certified region of attraction (ROA) as the set of initial states $\mathcal{A} \subseteq \mathcal{X}$ for which $\hat{R}_k \subseteq \mathcal{S}$ for all $k \in \{1, \dots, K\}$ and $\hat{R}_K \subseteq \mathcal{X}_G$.

RSOA Computation: The RSOA can be computed symbolically (i.e., over time horizon $N \geq 1$ at once). The one-step case $N = 1$ is commonly used [22]; however, increasing N yields tighter bounds than one-step methods by avoiding per-step overapproximation error [8], [25]. Specifically, let $\tilde{f}^N(\cdot) = \tilde{f} \circ \cdots \circ \tilde{f}(\cdot)$ denote N compositions of $\tilde{f}$ and let the F^N denote its CG. The N-step RSOA from timestep k is:

$$\hat{R}^N_{k+N}(\mathcal{X}_k \mid \eta_k) = \mathsf{Int}\Big(\min_{x \in \mathcal{X}_k} \underline{F}^N(x), \max_{x \in \mathcal{X}_k} \overline{F}^N(x) \Big), \tag{4}$$

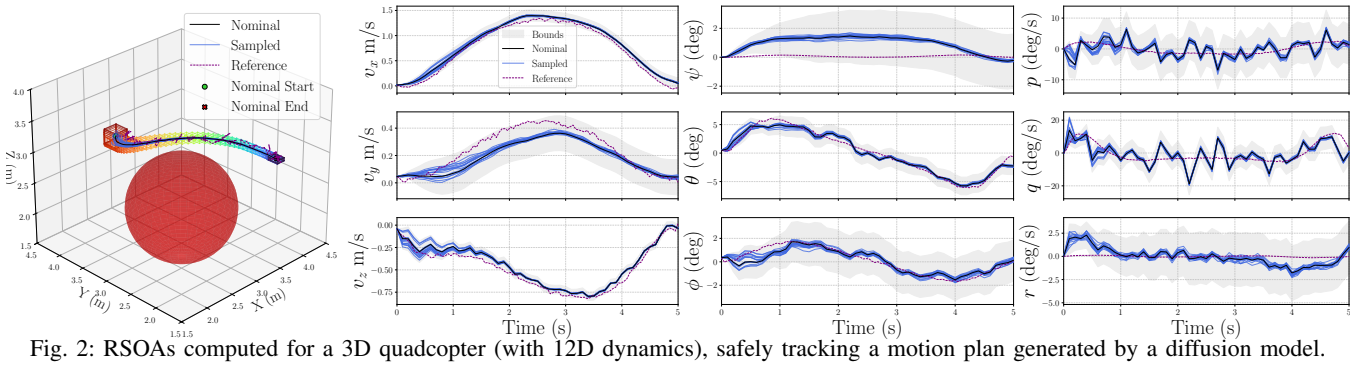

Fig. 2: RSOAs computed for a 3D quadcopter (with 12D dynamics), safely tracking a motion plan generated by a diffusion model.

where the notation $\hat{R}^j_m(\mathcal{X}_k \mid \eta_k)$ denotes the RSOA at timestep $m \in \{k, \ldots, k+j\}$, computed symbolically over a length-j horizon starting from initial set $\mathcal{X}_k$. In (4), $\underline{F}^N$ and $\overline{F}^N$ are affine bounding functions for F^N obtained using `auto_LiRPA`. If $k = 1$ and $N = K$, we compute the full K-step reachable set from $\mathcal{X}_I$, known as one-shot reachability [25]. Partitioning $\mathcal{X}_I$ into smaller sub-intervals yields tighter bounds [10]. While partitioning and symbolic methods yield tighter RSOAs, they are more computationally demanding than one-step methods [8], [10].

C. Generative Motion Planners

We use the following motion planners $P : \mathcal{X}_I \times \mathcal{X}_G \times \mathcal{E} \to \mathcal{T}$ that map an initial state $x_I \in \mathcal{X}_I$, goal $x_g \in \mathcal{X}_G$, and task data $e \in \mathcal{E}$ (e.g., a natural language command) to generate a reference trajectory $\tau = [p_1, \ldots, p_K] \in \mathcal{T} \subseteq \mathbb{R}^{n \times K}$. We denote learned parameters by ν.

Neural ODEs (NODEs) [32] learn a neural vector field $\dot{x}_t = \phi_\nu(x_t, t)$ which is numerically integrated from x_I to generate a discrete-time trajectory.

Conditional Flow Matching (CFM) [33] learns an ODE $\dot{z}_t = \psi_\nu(z_t, t)$ transporting a latent sample z_0 from a prior to a target trajectory distribution, yielding $\tau = z_1$ by integration.

Diffusion Models [34] learn a network ϵ_ν to reverse a multi-step noising process, iteratively denoising an initial sample $z_T \sim \mathcal{N}(0, I)$ in order to generate a clean trajectory τ.

Vision-Language Models (VLMs) [35] act as zero-shot planners that generate position waypoints from an image and text, which are linearly interpolated to create trajectory τ.

IV. Problem Statement

We are given dynamics (1), disturbance set $\mathcal{W}$, initial state set $\mathcal{X}_I \subseteq \mathcal{X}$, goal set $\mathcal{X}_G \subseteq \mathcal{X}$, a safe set $\mathcal{S} \subseteq \mathcal{X}$ encoding safety and task constraints, and a planner $P : \mathcal{X}_I \times \mathcal{X}_G \times \mathcal{E} \to \mathcal{T}$, which maps an initial state $x_I \in \mathcal{X}_I$, a goal $x_{\text{goal}} \in \mathcal{X}_G$, and environment/task data (e.g., images, natural language) to a K-step reference trajectory $\tau \in \mathcal{T} \subseteq \mathbb{R}^{n \times K}$. References from P may not be dynamically feasible (i.e., consistent with (1)) or safe (i.e., contained in $\mathcal{S}$). We assume nothing further about P; it may be stochastic and can represent both learned and traditional planners. We aim to (1) design a controller that replicates the behavior of P whenever it is safe to do so and (2) formally verify that the closed-loop system safely reaches $\mathcal{X}_G$. Specifically, we aim to solve:

Problem 1 (Design). *Train an NN control policy $\pi_\theta : \mathcal{X} \times \mathcal{I} \to \mathcal{U}$ that, given the current state x and parameters η, generates closed-loop trajectories from $\mathcal{X}_I$ that (1) resemble the open-loop trajectories produced by P while (2) ensuring safety and dynamic feasibility.*

Problem 2 (Verification). *Verify that for all $x_I \in \mathcal{X}_I$, the system reaches $\mathcal{X}_G$ at timestep K under (2) while staying in $\mathcal{S}$ for all $k \in \{1, \ldots, K\}$.*

V. Methodology

We overview our method, SaGe-MP (Fig. 1), which learns a trajectory tracking controller (Sec. V-A), formulates the closed-loop tracking dynamics for a fixed GMP plan as a CG (Sec. V-B), and performs NN verification on this CG to compute the RSOA and ensure safety (Sec. V-C). Finally, we sample distinct GMP references to construct a library of certified-safe RSOAs and use it to provide a runtime algorithm to preserve the original GMP behavior (Sec. V-D).

A. Learned tracking controller design

Since GMPs typically generate reference trajectories that are not dynamically-feasible, the resulting error must be counteracted, e.g., with replanning or a trajectory-tracking controller. We opt for the latter; in this section, we describe our method for training and formally verifying a tracking controller. Note that we train a *single* small tracking controller that is *trajectory-conditioned*; we do not train a new tracking controller for different references and show that our controller generalizes well to different references in Sec. VI.

We consider tracking controllers $\pi_\theta : \mathcal{X} \times \mathcal{T} \to \mathcal{U}$ that map a state x and reference trajectory $\tau \in \mathcal{T}$ to a control input. For brevity, at timestep k, we also refer to this controller as $u_k = \pi_\theta(e_k)$, where $\pi_\theta : \mathbb{R}^{n \times N} \to \mathbb{R}^m$ has learnable parameters θ and e_k collects reference errors over a future horizon of N references $p_{k+1}, \ldots, p_{k+N}$:

$$e_k \doteq [p_{k+1} - x_k,\ p_{k+2} - x_k,\ \ldots,\ p_{k+N} - x_k], \tag{5}$$

Conditioning on future errors makes π_θ focus on reducing tracking error and anticipating changes, specializing to local error correction instead of global planning, which is handled by the expressive GMP. Thus π_θ can remain small (≤ 100 neurons), which aids verification by keeping the unrolled CG compact and the resulting RSOA less conservative (cf. baselines in Figs. 3, 4). We represent π_θ as a small multi-layer perceptron (MLP) with leaky-ReLU activations and train it to produce dynamically-consistent corrective actions by minimizing the loss $\mathcal{L}(\theta) = \lambda_1 \mathcal{L}_{\text{track}} + \lambda_2 \mathcal{L}_{\text{state}} + \lambda_3 \mathcal{L}_{\text{ctrl}}$, which combines a mean squared tracking error $\mathcal{L}_{\text{track}}$ with

smooth penalties for state ($\mathcal{L}_{\text{state}}$) and control ($\mathcal{L}_{\text{ctrl}}$) limit violations over a horizon N:

$$\mathcal{L}_{\text{track}} = \tfrac{1}{N}\sum_{j=k}^{k+N} \|x_{j+1} - p_{j+1}\|^2, \tag{6}$$

$$\mathcal{L}_{\text{state}} = \tfrac{1}{N}\sum_{j=k}^{k+N} \left(\sigma(x_{j+1} - x_U) + \sigma(x_L - x_{j+1})\right), \tag{7}$$

$$\mathcal{L}_{\text{ctrl}} = \tfrac{1}{N}\sum_{j=k}^{k+N} \left(\sigma(u_j - u_U) + \sigma(u_L - u_j)\right), \tag{8}$$

where x_k is the closed-loop state at timestep k under controller π_θ and $\sigma(\cdot)$ is the softplus function, which penalizes violations of hyper-rectangular state bounds $\mathsf{Int}(x_L, x_U)$ and control bounds $\mathsf{Int}(u_L, u_U)$. To optimize $\mathcal{L}$, we compute gradients by rolling out N-step closed-loop trajectories under the dynamics (1), which we assume is differentiable.

B. Tracking Neural Feedback Loop (T-NFL)

We define the Tracking Neural Feedback Loop (T-NFL), denoted Γ, as a CG that models the evolution of the closed-loop tracking dynamics over the execution horizon K. This graph is constructed by unrolling the K-fold recursive application of the single-step closed-loop dynamics, $x_{k+1} = f(x_k, \pi_\theta(e_k), w_k)$, where π_θ is the tracking controller from Sec. V-A and e_k denotes the N-step lookahead reference tracking error. Specifically, the T-NFL is a function Γ that maps an initial state $x_I \in \mathcal{X}_I$, an *a priori* fixed reference $\tau \in \mathcal{T}$ sampled from the GMP, and a disturbance sequence $\mathbf{w} = \{w_1, \ldots, w_K\} \in \mathcal{W}^K \subseteq \mathbb{R}^{d\times K}$ to the resulting closed-loop trajectory $\Gamma(x_I, \tau, \mathbf{w}) = [\Gamma_1(x_I, \tau, \mathbf{w}), \ldots, \Gamma_K(x_I, \tau, \mathbf{w})] \doteq \xi = \{x_1, \ldots, x_K\} \in \mathbb{R}^{n\times K}$, i.e.,

$$\begin{aligned}\Gamma_k(x_I, \tau, \mathbf{w}) \doteq f(\ldots, f(f(x_I, \tau_{1:N}, w_1), \tau_{2:N+1}, w_2),\\ \ldots, \tau_{k-1:k+N-2}, w_k),\end{aligned} \tag{9}$$

where we denote $\mathcal{W}^K = \mathcal{W} \times \ldots \times \mathcal{W}$ as the K-fold Cartesian product of $\mathcal{W}$. In particular, at every timestep k, the next N references $p_{k+1}, \ldots, p_{k+N}$ are provided to the tracking controller. To guarantee that N valid future references are always available, we extend the reference trajectory by padding it with the final point p_K until its length reaches $K + N$. The process for generating the closed-loop trajectory is detailed in Alg. 1 and shown in Fig. 1.

Algorithm 1 T-NFL

Input: Initial state x_I; Reference trajectory τ; Disturbance sequence $\mathbf{w} = \{w_1, \ldots, w_K\}$
Output: Closed-loop trajectory rollout ξ.

$x_0 \leftarrow x_I, \quad \xi \leftarrow [\,]$
for $k = 1, \ldots, K$:
  $x_{k+1} \leftarrow f(x_k, \pi_\theta(p_{k+1} - x_k, \ldots, p_{k+N} - x_k), w_k)$
  $\xi \leftarrow [\xi, x_{k+1}]$
return ξ

C. Safety Verification with Reachability Analysis

To formally verify the safety of a generated plan, we compute an RSOA for the full-horizon closed-loop trajectory using one-shot reachability analysis [25]. We leverage the T-NFL's structure as a single computational graph, Γ, which represents the K-timestep closed-loop system rollout, from $k = 0, \ldots, K-1$. This graph is passed to the `auto_LiRPA` library [31], which propagates set-based inputs through the graph to bound the final output set.

Algorithm 2 Generation of RSOA library $\mathcal{C}$

Input: GMP P; initial set $\mathcal{X}_I$; goal set $\mathcal{X}_G$; RSOA budget C; environment $\mathcal{E}$; disturbance set $\mathcal{W}$;
Output: A set of safe GMP references and RSOAs, $\mathcal{C}$.

$\mathcal{C} \leftarrow \emptyset$
while $|\mathcal{C}| < C$:
  $\tau^c \leftarrow P(\mathcal{X}_I, \mathcal{X}_G, \mathcal{E})$
  if not COLLISION$(\tau, \mathcal{E})$:
    $\hat{R}^K_{1:K}(\mathcal{X}_I \mid \tau^c)$, success $\leftarrow$ FINDRSOA$(\mathcal{X}_I, \tau^c, \mathcal{W})$
    if success **and not** COLLISION$(\hat{R}^K_{1:K}(\mathcal{X}_I \mid \tau^c), \mathcal{E})$:
      Add $(\tau^c, R^K_{1:K}(\mathcal{X}_I \mid \tau^c))$ to $\mathcal{C}$
return $\mathcal{C}$

Specifically, we encode the uncertain inputs using hyper-rectangular sets. The set of initial states is defined as $\mathcal{X}_I = \{x \mid \|x - x_I\|_\infty \le \epsilon\}$ and the set of possible disturbances at each step is defined as $\mathcal{W} = \{w \mid \|w\|_\infty \le \overline{w}\}$. In contrast, the reference trajectory τ is fixed and passed as a deterministic tensor. `auto_LiRPA` [31] passes these inputs through Γ to find vector-valued affine bounding functions, $\underline{\Gamma}$ and $\overline{\Gamma}$, for the K-step closed-loop trajectory ξ. This enables us to compute the RSOA at each timestep k by considering specific components of these functions, $\underline{\Gamma}_k$ and $\overline{\Gamma}_k$, which bound the state of the closed-loop system at timestep k for all $\mathbf{w} \in \mathcal{W}^k$ and for all initial conditions $x_I \in \mathcal{X}_I$. This hyper-rectangular set, $\hat{R}^K_k(\mathcal{X}_I \mid \tau)$, is found by determining the range of these affine bounding functions over the input sets. Since the functions are affine and the input sets are hyper-rectangles, this optimization is tractable and computed efficiently by `auto_LiRPA`:

$$\hat{R}^K_k(\mathcal{X}_I \mid \tau) \doteq \mathsf{Int}\left(\min_{\substack{x_I\in\mathcal{X}_I\\ \mathbf{w}\in\mathcal{W}^k}} \underline{\Gamma}_k(x_I, \tau, \mathbf{w}), \max_{\substack{x_I\in\mathcal{X}_I\\ \mathbf{w}\in\mathcal{W}^k}} \overline{\Gamma}_k(x_I, \tau, \mathbf{w})\right). \tag{10}$$

We can use these RSOAs $\hat{R}^K_{1:K}(\mathcal{X}_I \mid \tau)$ to formally verify reach-avoid properties for the closed-loop system. Specifically, we can ensure that the closed-loop system reaches goal set $\mathcal{X}_G$ by checking if $\hat{R}^K_K(\mathcal{X}_I \mid \tau) \subseteq \mathcal{X}_G$ and that the system is safe and satisfies task constraints by checking if $\hat{R}^K_k(\mathcal{X}_I \mid \tau) \subseteq \mathcal{S}$ for all $k \in \{1, \ldots, K\}$. These guarantees hold for all disturbance sequences that may be drawn from $\mathcal{W}^K$ and all initial conditions drawn from $\mathcal{X}_I$.

D. RSOA libraries for multimodality and improved safety

A key advantage of GMPs is their multimodality: producing diverse, viable trajectories for a task, potentially from different homotopy classes. GMPs are typically trained on multimodal expert demonstration data which encodes desirable behaviors. However, when reproduced by the GMP, the resulting plans may not be safe or dynamically feasible. Our goal in Sec. V-D is to build on the method of Sec. V-C to balance two objectives: 1) preserve similarity to the GMP when safe while 2) increasing the safety rate when starting from the certified ROA $\mathcal{A}$ with our method relative to the executing the raw GMP samples, avoiding costly retraining.

Preserving multimodality. While following the closed-loop tracking policy defined by a single certified RSOA (and corresponding reference trajectory) ensures robust constraint satisfaction, it also collapses the closed-loop system into a

unimodal trajectory distribution, which may not reflect the potential multimodality of the original GMP distribution.

To address this, we build upon Sec. V-C by building a library $\mathcal{C}$ of C reference trajectories sampled from the GMP and corresponding safe RSOAs $\mathcal{C} \doteq \{(\tau^i, \hat{R}^K_{1:K}(\mathcal{X}_I \mid \tau^i)\}_{i=1}^C$. At runtime, our method chooses a reference from $\mathcal{C}$ to track in order to better imitate the GMP distribution. Our method for building $\mathcal{C}$ is presented in Alg. 2. It repeatedly samples references τ^c from the GMP P and, after an initial constraint check (ensuring that $\tau^c_K \in \mathcal{X}_G$ and $\tau^c_k \in \mathcal{S}$ for all k) and computes the closed-loop RSOA $\hat{R}^K_{1:K}(\mathcal{X}_I \mid \tau^c)$ for tracking τ^c under $\pi_\theta(x, \tau^c)$. If the RSOA satisfies the constraints, i.e., $\hat{R}^K_K(\mathcal{X}_I \mid \tau^c) \subseteq \mathcal{X}_G$ and $\hat{R}^K_k(\mathcal{X}_I \mid \tau^c) \subseteq \mathcal{S}$ for all k, it is added to the library $\mathcal{C}$; if not, the process repeats until C safe RSOAs are found; C is a maximum RSOA computation budget and can be tuned based on observed imitation error. For efficiency, this process can be parallelized on the GPU across multiple references τ^c.

At execution time, a new reference τ^{samp} starting from x_I^{samp} is sampled from the GMP P. We select the closest reference trajectory in $\mathcal{C}$, as measured by the minimum ℓ_2 error

$$E_\tau(\tau^{\text{samp}}, \mathcal{C}) \doteq \min_{i \in \{1,\dots,C\}} \sum_{k=1}^K \|\tau^i_k - \tau^{\text{samp}}_k\|_2^2, \quad (11)$$

where we denote i^* as the $\arg\min$ of (11). Finally, to obtain a safely-stabilizable, dynamically-feasible trajectory, we evaluate the closed-loop dynamics obtained by tracking τ^{i^*} under π_θ, yielding the closed-loop trajectory $\xi = \Gamma(x_I^{\text{samp}}, \tau^{i^*}, \mathbf{w})$.

Theoretical Analysis. In the following, for the specific case where the open-loop dynamics (1) are deterministic, i.e., $x_{t+1} = f(x_t, u_t)$, we derive finite-sample bounds on the rate at which the closed-loop imitation error

$$E_\xi(\tau^{\text{samp}}, \mathcal{C}) \doteq \|\Gamma(x_I^{\text{samp}}, \tau^{i^*}) - \Gamma(x_I^{\text{samp}}, \tau^{\text{samp}})\| \quad (12)$$

decreases as the size of the library $\mathcal{C}$ increases. At a high level, this result shows that if τ^{samp} can be safely stabilized under π_θ, as C increases, $\Gamma(x_I^{\text{samp}}, \tau^{i^*})$ converges to $\Gamma(x_I^{\text{samp}}, \tau^{\text{samp}})$ – the trajectory that would have been executed by directly tracking τ^{samp}. That is, as the library grows, our method becomes less invasive and more closely approximates the GMP trajectory distribution.

Definition 1 (μ-safe stabilization)**.** *A reference τ starting from x_I to be μ-safely stabilizable if $\hat{R}^K_k(\mathcal{B}_\mu(x_I) \mid \tau) \subseteq \mathcal{S}$ for all $k \in \{1, \dots, K\}$ and $\hat{R}^K_K(\mathcal{B}_\mu(x_I) \mid \tau) \subseteq \mathcal{X}_G$.*

Here, L is the Lipschitz constant of the T-NFL with respect to the reference, i.e., L satisfies $\|\Gamma(x_I, \tau) - \Gamma(x_I, \tau^{\text{samp}})\| \leq L\|\tau - \tau^{\text{samp}}\|$. The following result proves a bound on C such that with probability at least $1 - \alpha$, the error between the original and certified closed-loop trajectories (12) $\|\xi^{\text{samp}} - \xi^{i^*}\|$ is at most ϵ, i.e., $\Pr\big[E_\xi(\tau^{\text{samp}}, \mathcal{C}) \leq \epsilon\big] \geq 1 - \alpha$:

Theorem 1 (Minimal invasiveness)**.** *Assume that A) the open-loop dynamics* (1) *are deterministic, B)* (1) *and π_θ are Lipschitz continuous in all arguments, C) $\mathcal{X}_I = \mathcal{B}_\mu(x_I)$, and D) Alg. 2 samples reference trajectories i.i.d. from the GMP P. Let ϵ be a pre-specified error threshold and τ^{samp} be an μ-safely stabilizable reference generated from P with the properties that A) for $\delta = \epsilon/L$, Pr(P draws sample from $\mathcal{B}_\delta(\tau^{samp})$) $\doteq p_{samp} \in (0, 1]$ and that B) for all $\tau \in \mathcal{B}_\delta(\tau^{samp})$, τ is also μ-safely stabilizable. After C i.i.d. draws from P,*

$$Pr\big[E_\xi(\tau^{samp}, \mathcal{C}) \leq \epsilon\big] \geq 1 - \exp(-p_{samp}C), \quad (13)$$

and to ensure that $Pr\big[E_\xi(\tau^{samp}, \mathcal{C}) \leq \epsilon\big] \geq 1 - \alpha$ for $\alpha \in (0, 1)$, one can select $C \geq \log(1/\alpha)/p_{samp}$.

Proof. If τ^{samp} is μ-safely stabilizable, and the probability of sampling $\tau \in \mathcal{B}_\delta(\tau^{\text{samp}})$ is p_{samp} per draw, the chance of zero successes in C i.i.d. trials is $1 - (1 - p_{\text{samp}})^C \geq 1 - \exp(-p_{\text{samp}}C)$. When some $\tau \in \mathcal{B}_\delta(\tau^{\text{samp}})$ is sampled, it will be added to $\mathcal{C}$ as it is μ-safely stabilizable. Then, at runtime, when τ^{samp} is sampled, from the definition of Lipschitz continuity, we have that $\|\Gamma(x_I, \tau) - \Gamma(x_I, \tau^{\text{samp}})\| \leq L\|\tau - \tau^{\text{samp}}\|$ for all τ. L is guaranteed to exist and be finite since f and π_θ are Lipschitz and f is deterministic. Since $\delta = \epsilon/L$ and there exists some τ in $\mathcal{C}$ with probability at least $1 - \exp(-p_{\text{samp}}C)$ that satisfies $\|\tau - \tau^{\text{samp}}\| \leq \delta$, we have that $E_\xi(\tau^{\text{samp}}, \mathcal{C}) \leq \|\Gamma(x_I, \tau) - \Gamma(x_I, \tau^{\text{samp}})\| \leq L\|\tau - \tau^{\text{samp}}\| \leq L\delta \leq \epsilon$. □

In Sec. VI, we empirically validate that by collecting even a small number of RSOAs in $\mathcal{C}$, we can reduce the trajectory distribution shift with respect to the GMP P.

Algorithm 3 Safe Multimodal Runtime Execution

Input: initial state x_I; GMP reference τ^{samp}; library $\mathcal{C}$; tracking controller π_θ; disturbances $\mathbf{w} = \{w_1, \dots, w_K\}$
Output: Closed-loop trajectory rollout ξ.

```
E_τ(τ^samp, C), i* ← Solve (11)
τ^i ← Reference i* from C
ξ ← Γ(x_I, τ^{i*}, w)
return ξ
```

Verifying large sets of initial conditions. We can also extend our approach to verify safe forward invariance for large initial condition sets $\mathcal{X}_I$ by blending multiple GMP-generated reference trajectories with the tracking controller π_θ. In contrast, naïvely stabilizing a large $\mathcal{X}_I$ with π_θ around a single reference causes the RSOA to grow out of bounds in practice.

To prevent this, we sample multiple reference trajectories τ^i from the GMP, each with different initial conditions $x_I^i \in \mathcal{X}_I$, and stabilize a smaller neighborhood $\mathcal{X}_I^i \doteq \mathcal{B}_\epsilon(x_I^i) \subseteq \mathcal{X}_I$ of initial conditions to τ^i using π_θ. This relies on only local trajectory robustness, enabling the tracking controller to remain relatively simple. As in Alg. 2, we can form a library of references and RSOAs $\mathcal{C} \doteq \{(\tau^i, \hat{R}^K_{1:K}(\mathcal{X}_I^i \mid \tau^i)\}_{i=1}^C$. Then, by selecting these initial sets to partition $\mathcal{X}_I$, i.e., $\mathcal{X}_I = \bigcup_{i=1}^C \mathcal{X}_I^i$, at execution time, if $x_I \in \mathcal{X}_I^i$, we can use $\mathcal{C}$ and track τ^i using π_θ; the corresponding RSOA certifies that $\tilde{f}$ stays in $\mathcal{S}$ for all $k \in \{1, \dots, K\}$.

VI. Results

In this section, we compare to baselines (Sec. VI-A), evaluating on nonlinear and learned unicycle and quadcopter dynamics (Sec. VI-B-VI-D), and demonstrating our method on multimodal GMPs and verification of large $\mathcal{X}_I$ sets (Sec. VI-E-VI-F. were conducted on a desktop computer equipped with an Intel i9-14900K CPU, 64GB of RAM, and an NVIDIA RTX 4090 GPU. For all GMPs other than the pre-trained VLM, we generated training data via trajectory optimization. See App. A in [15] for further experiment details.

A. Baseline Comparisons

We motivate our method by highlighting three challenges in improving the safety of learned planners. First, large NN tracking controllers yield loose RSOAs as computed by NNV tools, motivating smaller controllers. Second, sampling-based verification of RSOAs is highly sample-inefficient over long horizons, motivating NNV tools. Finally, neural CBFs safety filters can be unsound and lead to violations, motivating formally verified controllers.

Loose RSOAs for large NNs. We illustrate the tradeoff between NN size and closed-loop RSOA volume. As shown in Fig.3, larger state feedback NN policies improve expert data recreation error (blue) and shrink the exact reachable set $R^K_{1:K}(\mathcal{X}_I)$ but enlarge the computed RSOA $\hat{R}^K_{1:K}(\mathcal{X}_I)$ due to the resulting NNV conservativeness (red). For example, Fig. 4 shows a neural ODE with three hidden layers of width 256, which is treated as a planner that replicates the desired closed-loop dynamics, yields a conservative RSOA that grows unbounded while the trajectories remain bounded.

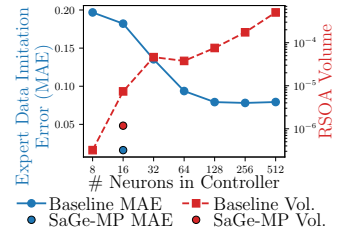

Fig. 3: We visualize the trade-off between controller capacity and RSOA volume. While larger models (more neurons) reduce tracking error, they inflate the reachable set volume, leading to more conservative guarantees.

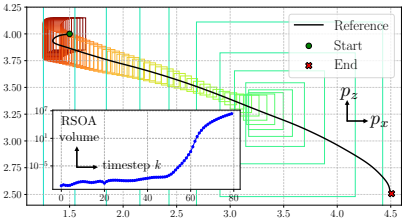

Fig. 4: RSOA for neural ODE (756 neurons). The rapid RSOA growth shows a looseness issue for NNV with larger models; this is reflected in the RSOA volumes as well (inset).

Inefficiency of sampling-based safety analysis. Sampling-based methods are computationally inefficient at uncovering rare but critical failures. In a 2D quadcopter scenario (Fig. 5), our NN tracking controller follows a $K = 100$-length reference trajectory (black) generated by a neural ODE while avoiding an obstacle (red). After 10^6 simulations, only seven collisions were found, whereas the computed RSOA intersects the obstacle, immediately certifying the possibility of a violation. The high cost of sampling highlights its limitations and the advantage of reachability-based formal verification.

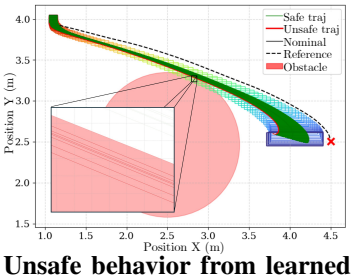

Fig. 5: Limitations of sampling-based safety analysis. While 10^6 random simulations uncovered just seven collisions over a $K = 100$ horizon, our RSOA-based method can directly certify the potential for collision.

Unsafe behavior from learned safety filters. A baseline neural CBF [13] fails to reliably enforce safety. In tracking a neural ODE-generated reference under unicycle dynamics (Fig. 6), the learned CBF overrides the controller when inputs appear unsafe, yet only 46.4% of 1000 rollouts from $\mathcal{X}_I$ remain safe (orange). This is because typically neural CBFs only enforce the CBF conditions on sampled states [13], compromising the guarantees. In contrast, our method formally certifies that the closed-loop system is 100% safe.

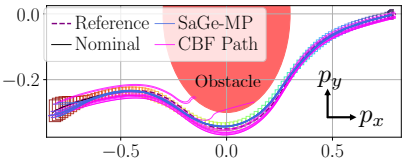

Fig. 6: A neural CBF baseline (pink) frequently fails to ensure safety, while SaGe-MP (blue) guarantees safety.

B. Unicycle

Simulation results on analytical and learned dynamics. We evaluate our method on unicycle dynamics [16, Eq. 13.18], discretized with a $0.1s$ forward-Euler step, tracking trajectories for $K = 100$ timesteps. To test applicability to learned models, we train an MLP (3 layers, 64 neurons) on 2×10^6 transitions from the ground truth. A NODE planner is trained on ground-truth trajectories, and a tracking controller π_θ with two hidden layers of width 8 is trained to follow them; the same controller is used in both cases.

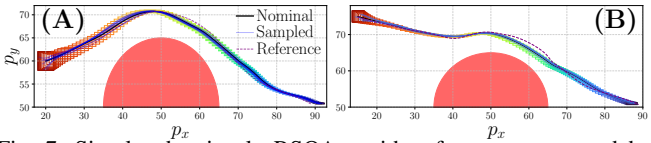

Fig. 7: Simulated unicycle RSOAs, with references generated by (A) NODE + analytical dynamics, (B) NODE + learned dynamics.

As shown in Fig. 7, under both analytical (a) and learned (b) dynamics, our T-NFL safely tracks neural ODE reference trajectories, producing RSOAs that formally verify safety and goal-reaching. RSOA computation times were $100.78s$ and $51.95s$, respectively. This experiment shows that our method can follow NODE-generated plans and certify closed-loop safety for both analytical and learned dynamics.

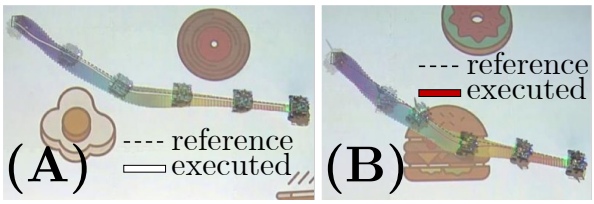

Fig. 8: Hardware deployment: a plan is generated from an image of the scene and a prompt using a VLM, which is safely executed by our controller on an differential-drive robot.

Hardware results with natural language task constraints. We demonstrate our framework on a physical wheeled robot (Fig. 8). A VLM (Gemini 2.5 Pro) [35] generated reference trajectories from images and text prompts. We evaluate two examples: one where the VLM generates a collision-free path to the goal (Fig. 8a), and another where it reaches the target while moving over a food item and treating other objects as obstacles (Fig. 8b). See App. A in [15] for prompt details.

The VLM produces reference trajectories as discussed in Sec. III-C. The same tracking controller from the simulation experiments is used to follow these trajectories. At runtime on hardware, the vehicle stayed within the computed RSOA (Fig. 8). Despite being trained only on NODE-generated references, the controller successfully tracked VLM-generated trajectories, with formal verification providing guaranteed

RSOA computation even out-of-distribution. These results show that our controller generalizes to unseen reference trajectory distributions and safely stabilizes plans from images and language, ensuring task completion from any state in $\mathcal{X}_I$.

C. Planar Quadcopter

Next, we evaluate our framework on a planar, 6D quadcopter model (App. A in [15]), time-discretized with a timestep of 0.05s for a horizon of $K = 100$. The task is to navigate a 5×5 m^2 workspace containing a circular obstacle.

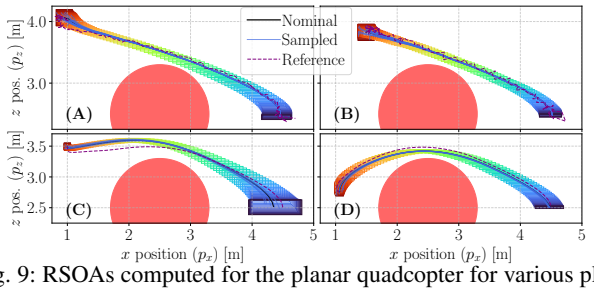

Fig. 9: RSOAs computed for the planar quadcopter for various planners, verifying safety: (A) diffusion + analytical dynamics (AD); (B) CFM + AD; (C) NODE + learned dynamics; (D) NODE + AD.

Analytical dynamics. As shown in Fig. 9, our framework certifies plans from diverse learned planners, including diffusion, CFM, and neural ODE models. The tracking controller (2 hidden layers, 8 neurons each) reliably follows the generated trajectories, yielding collision-free closed-loop RSOAs and certifying safety for all $x_I \in \mathcal{X}_I$. RSOA computation times are comparable to before, averaging 123.21s for analytical dynamics and 59.85s for the learned model, using 4 and 25 initial partitions, respectively.

As a surrogate for measuring the increased safety rate when starting from the certified ROA with our method relative to the GMP, we consider a fixed input set $\mathcal{X}_I$ and compute empirical safety rates for 1) open-loop and 2) closed-loop execution of GMP plans. Out of 126 references sampled from the diffusion planner, 1) 78.5% and 2) 94.4% were safe starting from $\mathcal{X}_I$. For the same input set, we also evaluated the CFM-based planner, which yielded 68.5% and 81.5% safety out of 108 samples. In contrast, we are able to certify 100% safety; i.e., $\mathcal{X}_I \subseteq \mathcal{A}$ for diffusion and CFM.

Overall, this experiment suggests that SaGe-MP stabilizes noisy, dynamically-infeasible references from diffusion and CFM (Fig. 9, purple), increasing the safety rate by certifying the ROA and effectively "projecting" them onto the set of feasible trajectories that safely reach the goal, improving on the approximate projection of [6].

Learned dynamics. To illustrate scalability to higher-dimensional learned dynamics, we trained an NN dynamics model (3 hidden layers, 64 neurons each) from transitions of the analytical model, along with a tracking controller (2 layers, 16 neurons each). The resulting closed-loop RSOAs again show safe tracking (Fig. 9c). These results highlight the planner-agnostic nature of our method and ability to safely track trajectories from GMPs on underactuated dynamics.

D. 3-D Quadcopter

We test scalability on an analytical 12-state 3D quadcopter model [36], discretized with a 0.05s timestep over a $K = 100$ horizon. The task is set in a $5 \times 5 \times 5$ m^3 workspace with a spherical obstacle of radius 1m. We train a tracking controller π_θ, with 2 hidden layers of 8 neurons each, to track a diffusion-generated reference. As shown in Fig. 2, the computed RSOA guarantees safety: the controller tracks the reference trajectory while the certified RSOA remains collision-free, computed in 2153.45s with 4 uniform partitions of $\mathcal{X}_I$. This demonstrates that our method can scale to certifiably stabilize GMP-generated references on high-dimensional, underactuated nonlinear systems.

E. Safe, Multimodal Planning

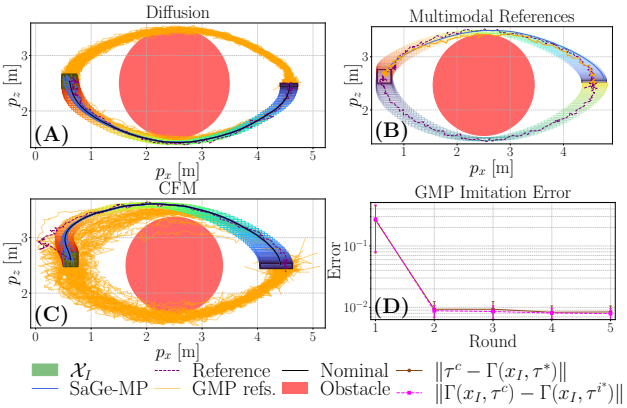

Fig. 10: We find one safe RSOA via Alg. 2 for diffusion (A) and CFM (C) planners. We illustrate our trajectory library approach (B), which can capture multiple modes of the GMP and reduce the imitation error relative to the original GMP (D).

We validate that generating multiple plans from $\mathcal{X}_I$ preserves the multimodality of the GMP. For analytical planar quadcopter dynamics (Fig. 10c), at a deliberately selected $\mathcal{X}_I$, the diffusion-based planner produces two modes - above and below the obstacle. We sample five safe references from the GMP and compute their RSOAs, visualizing two in Fig. 10c. At runtime, an initial state $x_I \in \bigcup_{i=1}^{C} \mathcal{B}_\epsilon(x_I^i)$ is measured, and the GMP is queried to return a reference trajectory that starts from x_I. We follow Alg. 3, showing that tracking the selected reference with π_θ can well-imitate the GMP trajectory distribution even with $C = 2$. As C increases, the imitation error (12) decreases (Fig. 10d); the raw error between the original GMP reference τ and our closed-loop rollout $\Gamma(x_I, \tau^{i^*})$ also decreases. Similarly, with analytical unicycle dynamics and a VLM-based GMP, our method certifies safe RSOAs around several sampled references; three are shown in Fig. 11a. Moreover, as a surrogate for measuring the increase in safety rate when starting from the certified ROA $\mathcal{A}$ relative to the GMP, we are able to certify 100% safety, i.e., $\mathcal{X}_I \subseteq \mathcal{A}$, while naïvely sampling and stabilizing plans with π_θ from the VLM leads to only 16% safety over 50 trials, shown in Fig. 11b. These results suggest our method can capture multimodal GMP behavior while improving safety rates relative to the original GMP. Moreover, this shows that the invasiveness of our approach decreases as C increases, with small error even for small C.

F. Verifying forward invariance for large $\mathcal{X}_I$ sets

We demonstrate blending GMP samples with the tracking controller π_θ to verify safe goal reachability for large initial sets $\mathcal{X}_I$. We partition $\mathcal{X}_I$ into a grid, sample a NODE-based reference from each cell center, and compute the

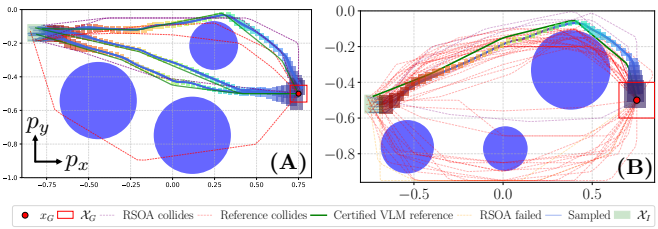

Fig. 11: (A) We illustrate our trajectory-library sampling approach on a VLM planner. VLM references that are unsafe (red) or have RSOAs in collision (purple) are discarded until a set of certified-safe RSOAs and references are collected (green). (B) Naïvely executing the VLM as a planner leads to only 16% safety starting from the green $\mathcal{X}_I$, whereas with our method enables 100% safety.

corresponding RSOA. Fig. 12 shows certified collision-free RSOAs (green) and those in collision (red). Once computed offline, these RSOAs allow fast evaluation of which subsets of $\mathcal{X}_I$ remain safe under new obstacle configurations, via collision-checking between hyper-rectangular RSOAs and the environment, taking 0.122 ± 0.003s seconds. While grid-based partitioning works best in low dimensions, this experiment shows pre-certification enables efficient real-time safety queries; moreover, to reduce partitioning in higher dimensions, one can employ adaptive gridding.

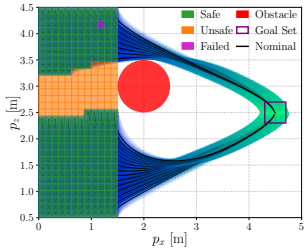

Fig. 12: We verify forward invariance for a large set of initial conditions (green) by blending our tracking controller with several sampled references from a NODE-based planner, on analytical planar quadcopter dynamics.

VII. Conclusion

We present SaGe-MP, a method for formally verifying the safety and dynamic feasibility of motion plans from large GMPs. Our approach stabilizes GMP-sampled references with a small neural tracking controller and applies NNV to the resulting closed-loop dynamics, enabling rigorous reachability guarantees even when the GMP is too large for direct analysis. By certifying multiple references offline and deploying them online via a trajectory library, our method preserves the GMP's multimodal behavior while increasing safety rate when starting from a certified-safe region of attraction. A limitation of our approach is that RSOA computational costs grow with state dimension and time horizon, limiting applicability in real-time, dynamic settings. Future work will explore linearizing the system dynamics to enable faster, less conservative reachability computations while bounding the linearization error.

References

[1] W. Xiao, T. Wang, C. Gan, R. M. Hasani, M. Lechner, and D. Rus, "Safediffuser: Safe planning with diffusion probabilistic models," in *ICLR*, 2025.

[2] E. Chisari, N. Heppert, M. Argus, T. Welschehold, T. Brox, and A. Valada, "Learning robotic manipulation policies from point clouds with conditional flow matching," in *CoRL*, 2024.

[3] X. Dai, D. Yu, S. Zhang, and Z. Yang, "Safe flow matching: Robot motion planning with control barrier functions," 2025.

[4] J. a. Carvalho, A. T. Le, M. Baierl, D. Koert, and J. Peters, "Motion planning diffusion: Learning and planning of robot motions with diffusion models," in *IROS*, 2024.

[5] F. Nawaz, T. Li, N. Matni, and N. Figueroa, "Learning complex motion plans using neural odes with safety and stability guarantees," 2024.

[6] J. Bouvier, K. Ryu, K. Nagpal, Q. Liao, K. Sreenath, and N. Mehr, "DDAT: diffusion policies enforcing dynamically admissible robot trajectories," *RSS*, 2025.

[7] H. Zhang, T. Weng, P. Chen, C. Hsieh, and L. Daniel, "Efficient neural network robustness certification with general activation functions," in *NeurIPS*, 2018, pp. 4944–4953.

[8] N. Rober and J. P. How, "Constraint-aware refinement for safety verification of neural feedback loops," *IEEE Control. Syst. Lett.*, 2024.

[9] M. Everett, G. Habibi, C. Sun, and J. P. How, "Reachability analysis of neural feedback loops," *IEEE Access*, vol. 9, 2021.

[10] M. Everett, G. Habibi, and J. P. How, "Robustness analysis of neural networks via efficient partitioning with applications in control systems," *IEEE Control. Syst. Lett.*, vol. 5, no. 6, pp. 2114–2119, 2021.

[11] S. Jafarpour, A. Harapanahalli, and S. Coogan, "Efficient interaction-aware interval analysis of neural network feedback loops," *TAC*, 2024.

[12] A. Dixit, L. Lindemann, S. X. Wei, M. Cleaveland, G. J. Pappas, and J. W. Burdick, "Adaptive conformal prediction for motion planning among dynamic agents," in *L4DC*, vol. 211, 2023, pp. 300–314.

[13] C. Dawson, Z. Qin, S. Gao, and C. Fan, "Safe nonlinear control using robust neural lyapunov-barrier functions," in *CoRL*, 2022.

[14] T. Lew, L. Janson, R. Bonalli, and M. Pavone, "A simple and efficient sampling-based algorithm for general reachability analysis," 2022.

[15] D. Nath, H. Yin, and G. Chou, "Formal safety verification and refinement for generative motion planners via certified local stabilization," 2025. [Online]. Available: https://arxiv.org/abs/2509.19688

[16] S. LaValle, *Planning Algorithms*. Cambridge University Press, 2006.

[17] K. Mizuta and K. Leung, "Cobl-diffusion: Diffusion-based conditional robot planning in dynamic environments using control barrier and lyapunov functions," in *IROS*, 2024, pp. 13 801–13 808.

[18] H. Dai and F. Permenter, "Convex synthesis and verification of control-lyapunov and barrier functions with input constraints," in *ACC*, 2023.

[19] A. Li, Z. Ding, A. B. Dieng, and R. Beeson, "Diffusolve: Diffusion-based solver for non-convex trajectory optimization," in *L4DC*, 2025.

[20] A. Ajay, Y. Du, A. Gupta, J. B. Tenenbaum, T. S. Jaakkola, and P. Agrawal, "Is conditional generative modeling all you need for decision making?" in *ICLR*, 2023.

[21] D. Gadginmath and F. Pasqualetti, "Dynamics-aware diffusion models for planning and control," 2025.

[22] M. Althoff, G. Frehse, and A. Girard, "Set propagation techniques for reachability analysis," *Annu. Rev. Control. Robotics Auton. Syst.*, 2021.

[23] A. Majumdar and R. Tedrake, "Funnel libraries for real-time robust feedback motion planning," *IJRR*, vol. 36, no. 8, pp. 947–982, 2017.

[24] G. Chou, N. Ozay, and D. Berenson, "Safe output feedback motion planning from images via learned perception modules and contraction theory," in *International Workshop on the Algorithmic Foundations of Robotics*. Springer, 2022, pp. 349–367.

[25] S. Chen, V. M. Preciado, and M. Fazlyab, "One-shot reachability analysis of neural network dynamical systems," in *ICRA*, 2023.

[26] A. Muthali, H. Shen, S. Deglurkar, M. H. Lim, R. Roelofs, A. Faust, and C. J. Tomlin, "Multi-agent reachability calibration with conformal prediction," in *CDC*, 2023.

[27] D. Sun and S. Mitra, "Neureach: Learning reachability functions from simulations," in *TACAS*, 2022.

[28] F. Jiang, G. Chou, M. Chen, and C. J. Tomlin, "Using neural networks to compute approximate and guaranteed feasible hamilton-jacobi-bellman pde solutions," 2017.

[29] C. Knuth, G. Chou, N. Ozay, and D. Berenson, "Planning with learned dynamics: Probabilistic guarantees on safety and reachability via lipschitz constants," *IEEE Robotics Autom. Lett.*, 2021.

[30] C. Knuth, G. Chou, J. Reese, and J. Moore, "Statistical safety and robustness guarantees for feedback motion planning of unknown underactuated stochastic systems," in *ICRA*, 2023.

[31] K. Xu, Z. Shi, H. Zhang, Y. Wang, K. Chang, M. Huang, B. Kailkhura, X. Lin, and C. Hsieh, "Automatic perturbation analysis for scalable certified robustness and beyond," in *NeurIPS*, 2020.

[32] T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud, "Neural ordinary differential equations," in *NeurIPS*, 2018.

[33] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow matching for generative modeling," in *ICLR*, 2023.

[34] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *NeurIPS*, 2020.

[35] Gemini Team *et al.*, "Gemini: A family of highly capable multimodal models," 2023.

[36] F. Sabatino, "Quadrotor control: Modeling, nonlinear control design, and simulation," M.S. thesis, KTH, Stockholm, Sweden, 2015.