Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 58-28 (2024) 863–868

SODA-RRT: Safe Optimal Dynamics-Aware Motion Planning ⋆

Nariman Niknejad * **Ramin Esmzad** * **Hamidreza Modares** *

* *Department of Mechanical Engineering, Michigan State University, East Lansing, MI 48824*

(e-mails: {niknejad,esmzadra,modaresh}@msu.edu).

Abstract: This paper introduces a performance-aware motion planning approach that generates collision-free paths with guaranteed optimality using invariant sets. The proposed planner constructs a sequence of conflict-free invariant sets, within which closed-loop trajectories maintain safety and performance criteria. Randomly generated waypoints serve as the center for these invariant sets, which are then connected to form a path from the initial to the target point. For each waypoint, an optimization problem determines the largest conflict-free zone and a safe-optimal controller. The novel algorithm termed Safe Optimal Dynamics-Aware Motion Planning (`SODA-RRT`), incorporates performance-reachability between connected waypoints, thus reducing the need for frequent re-planning. The method's efficacy is demonstrated through spacecraft motion planning scenarios involving debris avoidance, showcasing its potential for real-world applications.

Keywords: Motion planning, Invariant sets, Linear matrix inequalities (LMIs), Collision avoidance, Optimal Control, Safe Control.

1. INTRODUCTION

Motion planning involves finding a collision-free path for a robot or vehicle from an initial state to a target state in a given environment (LaValle, 2006). It becomes a daunting challenge to design efficient motion planners that account for dynamic environments, uncertainties, and system physical constraints (Kavraki and LaValle, 2016). Various motion planners have been presented in the literature, including sampling-based methods like rapidly exploring random tree (RRT)(LaValle and Kuffner, 2001), behavior-based approaches (Mataric, 1992), potential fields(Ge and Cui, 2002), and optimization-based techniques like MPC (Mayne et al., 2000). Despite the efficiency of sampling-based approaches, planners that rely exclusively on generating random waypoints often fail to consider 1) the system's dynamic limitations and constraints (LaValle, 2006; Karaman and Frazzoli, 2011), and 2) the performance of the traversed trajectories between selected waypoints. More specifically, these planners typically focus on generating kinematically feasible trajectories without accounting for the system's actual dynamics or control constraints (LaValle, 2006; Karaman and Frazzoli, 2011; Kavraki et al., 1996), and the existence of a controller that can make the system reach to one waypoint from another with acceptable performance. Ignoring the system dynamics and the competence of the low-level controller to deliver an acceptable performance can lead to frequent re-planning and poor planning. As a result, the generated trajectories may be far from optimal or even infeasible when executed on the real system, as they do not inherently guarantee safety or performance (LaValle, 2006; Karaman and Frazzoli, 2011; Kavraki et al., 1996; Schouwenaars et al., 2001).

In sharp contrast to classic methods, selecting *persistently feasible* and *performance-aware* reachable invariant sets offers a framework to ensure safety and stability by explicitly considering the system's dynamics and constraints (Borrelli et al., 2017). These methods compute a set of states for which the system will always remain within the set under a given control policy, guaranteeing that the planned trajectory stays within safe regions (Danielson et al., 2016). Invariant sets can be computed offline (Rakovic et al., 2005). Methods to find invariant sets include level sets of optimal control Lyapunov functions (Kousik et al., 2020), reachability analysis (Althoff and Dolan, 2014), and control barrier functions (Ames et al., 2019). However, these existing invariant-based approaches do not guarantee any performance of the resulting trajectory of the closed-loop system.

This paper introduces a novel performance-aware motion planning approach that generates collision-free paths with some degree of performance using invariant sets. It extends previous works by creating connected conflict-free invariant sets that ensure the closed-loop system's trajectories respect safety and optimality. This is achieved by randomly generating waypoints, forming invariant sets around them, and connecting them to create a sequence of sets from the initial to the target point. For each waypoint, the optimization problem finds the largest conflict-free zone

⋆ This work is partly supported by the Department of Navy award N00014-22-1-2159 and partly by the National Science Foundation under award ECCS-2227311.

10.1016/j.ifacol.2025.01.104

and its controller, ensuring safe traversal with guaranteed performance. Unlike conventional RRT methods, SODA-RRT accounts for the performance-reachability of connected waypoints, avoiding those that can only be reached with poor performance or no safety guarantee. This removes the need for frequent re-planning and provides performance guarantees besides safety. This paper also paves the way for the usage of physics-informed data-driven approaches (Niknejad and Modares, 2023). The effectiveness of SODA-RRT is demonstrated through spacecraft motion planning to avoid debris.

2. NOTATIONS AND PRELIMINARIES

Throughout this paper, I, $\mathbb{N}$, $\mathbb{R}$, and 0 denote the identity matrix, natural numbers, real numbers, and zero matrices, respectively. $Q(\preceq,\succeq)0$ implies Q is negative or positive semi-definite, and $tr(A)$ represents the trace of matrix A.

Definition 1. A convex and compact set $\mathcal{S}$ that includes the origin as its interior point is called a C-set. Also, for a set $\mathcal{S}$, we denote by $int(\mathcal{S})$ its interior.

Definition 2. A polytope is formed by the intersection of a finite number of half-spaces and denoted as $\mathcal{P} = \{x \in \mathbb{R}^n : W^T(l)x \leq G(l) \text{ for } l = 1, ..., m\} \subseteq \mathbb{R}^n$.

Definition 3. An ellipsoidal set is represented by $\mathcal{E}(P, c)$ and is defined as

$$\mathcal{E}(P, c) := \{x \in \mathbb{R}^n : (x - c)^T P^{-1}(x - c) \leq 1\}, \quad (1)$$

where $P^{-1} \in \mathbb{R}^{n\times n}$ is a symmetric positive definite matrix and the center of the ellipsoidal set is denoted by vector $c \in \mathbb{R}^n$.

This paper aims to design performance-aware safe motion planners for discrete-time linear time-invariant (LTI) systems described by

$$\begin{aligned} x(t+1) &= Ax(t) + Bu(t) + d(t), \\ y(t) &= Cx(t), \end{aligned} \quad (2)$$

where $x(t) \in \mathbb{R}^n$ denote the state vector, $u(t) \in \mathbb{R}^m$ represent the input vector, and $y(t) \in \mathbb{R}^o$ be the output vector of interest, which corresponds to the space in which the planning is performed. $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{m\times n}$ are the state and input and $C \in \mathbb{R}^{o\times n}$ is the output matrix. The system is under disturbance $d(t) \in \mathbb{R}^n$ with Gaussian distribution of $\mathcal{N}(0, I)$. In this paper, we assume that the planner has full knowledge of the states of the system $x(t)$.

Assumption 1. The pair (A, B) is stabilizable.

Assumption 2. The output $y(t)$ is subject to constraints

$$y(t) \in \mathcal{Y}. \quad (3)$$

The output set $\mathcal{Y} \subseteq \mathbb{R}^m$ is typically non-convex; however, it can be characterized as the union of convex sets

$$\mathcal{Y} = \bigcup_{\kappa\in\mathcal{I}_\mathcal{Y}} \mathcal{Y}_\kappa, \quad (4)$$

where the index set $\mathcal{I}_\mathcal{Y}$ is finite ($|\mathcal{I}_\mathcal{Y}| < \infty$) and each component set $\mathcal{Y}_\kappa \subseteq \mathbb{R}^m$ is a compact polytope defined by a set of linear inequalities

$$\mathcal{Y}_\kappa = \big\{y : W^T_{\mathcal{Y}_\kappa}(l)y \leq G_{\mathcal{Y}_\kappa}(l) \text{for } l = 1, ..., |\mathcal{I}_\mathcal{Y}|\big\}. \quad (5)$$

For the purpose of the tracking used by the motion planner, for the system (2) with $\bar{y}$, an equilibrium pair $(\bar{x}, \bar{u})$ is defined as

$$M \begin{bmatrix} \bar{x} \\ \bar{u} \end{bmatrix} = \begin{bmatrix} 0 \\ \bar{y} \end{bmatrix}, \quad (6)$$

where

$$M := \begin{bmatrix} A - I & B \\ C & 0 \end{bmatrix}. \quad (7)$$

Assumption 3. The matrix M is invertible.

The equilibrium pair $(\bar{x}, \bar{u})$ will change over time as the desired output $\bar{y}$ varies to guide the system from the starting point to the endpoint. To ensure safety throughout the trajectory, this paper employs the concept of λ-contractive sets, which will be defined next.

Definition 4. (Blanchini, 1999) *$\lambda-$Contractive and Positive Invariant Sets:* Consider the system (2). Let $\lambda \in (0, 1]$. The set $\mathcal{E}(P, c)$ is considered λ-contractive if for any $x(t) \in \mathcal{E}(P, c)$ it holds that $x(t + 1) \in \lambda\mathcal{E}(P, c)$. When $\lambda = 1$, $\mathcal{E}(P, c)$ is positive invariant.

Finally, to connect a sequence of sets that covers the initial and target points, the following definition of graphs is provided.

Definition 5. A directed graph denoted as $\mathcal{G} = (V, E)$, comprises a set of vertices V and a set of directed edges E. Each directed edge $(u, v) \in E$ is an ordered pair of adjacent vertices, where u is the tail (starting vertex) and v is the head (ending vertex). A path in a directed graph is a sequence of vertices connected by directed edges. A graph search algorithm aims to find a path between vertices in the graph, satisfying given criteria.

2.1 Motion-Planning Problem

This paper presents SODA-RRT, a motion-planning algorithm that steers the system output $y(t)$ from $y(0) \in \text{int}(\mathcal{Y})$ to $y_f \in \text{int}(\mathcal{Y})$ through a sequence of randomly generated overlapping invariant sets. Within these sets, safety and acceptable performance are guaranteed by a learned controller, subject to the system dynamics. Unlike classic approaches, SODA-RRT ensures safe and performance-aware reachability of planned trajectories by replacing the Euclidean distance metric in standard RRT with a set membership-based criterion. The multi-step SODA-RRT algorithm problem is formally described below.

Problem 1. Consider the system (2). Let $y(0) \in \text{int}(\mathcal{Y})$ and $y_f \in \text{int}(\mathcal{Y})$ be the initial and target points, respectively. The objective is to find a sequence of equilibrium points $(\bar{x}_i, \bar{u}_i)$, $i = 1, ..., |\mathcal{I}|$ in graph $G = (\mathcal{I}, E)$ along with low-level controllers to traverse along them with the guarantees of reaching the target point safely and with acceptable performance. For a given equilibrium point $(\bar{x}(t), \bar{u}(t))$, a tracking control law for the time step $t \in \mathbb{N}$ is given by

$$u(t) = K(t)\big(x(t) - \bar{x}(t)\big) + \bar{u}(t), \quad (8)$$

where $K(t)$ is the feedback associated with an invariant set. The steps to this problem are formulated as follows.

(1) Randomly sample the output space $\mathcal{Y}$ to find a candidate waypoint y_i. Take this waypoint y_i as a virtual equilibrium output from which an equilibrium point $(\bar{x}_i, \bar{u}_i)$ is found using (6).
(2) For each candidate waypoint y_i, formalize a feedback control design problem that finds a control gain K_i

that leads to an invariant set $\mathcal{S}_i$ centered at the candidate waypoint. The invariant set will incorporate performance and safety constraints.

(3) If at least one of the previously accepted waypoints is within the invariant set associated with the candidate waypoint y_i, then store the corresponding control gain K_i, its invariant set $\mathcal{S}_i$, and the equilibrium pair $(\bar{x}_i, \bar{u}_i)$ in graph $\mathcal{G}$. Otherwise, reject the candidate. The stored values will be used to construct the controller sequence $\mathcal{U}_p = \{u(t),\ \forall t \in \mathbb{N}\}$ using (8).

(4) Repeat steps 1-3 until an acceptable waypoint is found within the vicinity of the target point y_f.

Theorem 1 provides sufficient conditions for the existence of a path that solves Problem 1 for the general case of path planning using invariant sets.

Theorem 1. (Danielson et al., 2016) Let a graph be formed by connecting the equilibrium points obtained by solving steps of Problem 1. If the graph $\mathcal{G} = (V, E)$ has a path from the initial equilibrium output $V_{init} = y_0$ to a node V_{target} within the neighborhood of the target output y_f with a radius of r_f, then the Problem 1 can be solved.

3. CONTROL DESIGN FOR STEP 2 OF PROBLEM 1

To clarify the second step of Problem 1, without loss of generality, let us first consider a regulation problem for the system (2) with the equilibrium pair $(\bar{x} = 0, \bar{u} = 0)$. In this case, the control law (8) simplifies to

$$u_{fb}(t) = K\, x(t), \tag{9}$$

where $K \in \mathbb{R}^{m\times n}$. The target is to find a control feedback gain K that balances safety and performance. To this end, we formalize an optimization problem that leads to finding a conflict-free set or zone that covers a given waypoint. That is, inside this set, the system can traverse from the waypoint to any other point safely and with an acceptable performance. To find an ellipsoidal conflict-free zone, we parameterize the control gain K using an ellipsoidal λ-contractive safe set $\mathcal{E}(P_s, 0)$. We impose a constraint into the LQR problem to ensure that the conflict-free ellipsoidal set is inside the polyhedral safe set $\mathcal{X}_{ph}$. By optimizing the LQR cost along with maximizing the size of the conflict-free zone, a trade-off between safety (which is respected inside the conflict-free zone) and performance is achieved. Thus, a control goal is found that guarantees safety and some performance level inside the conflict-free zone.

Assumption 4. For the system (2) and the cost function (11), the pair $(A, \sqrt{Q})$ is detectable.

3.1 Problem Statement

Problem 2. Consider system (2) and Assumptions 1 and 4. Find a feedback control gain K, same as in (9), and an associated $\lambda-$contractive ellipsoidal set $\mathcal{E}(P_s, 0) = x(t)^T P_s^{-1} x(t) \leq 1$ contained is the polyhedral safe set $\mathcal{X}_{ph}$ for the states of the system

$$\mathcal{X}_{ph} = \left\{\zeta \in \mathcal{R}^n \quad : \quad W^T(l)\zeta \leq 1, \quad l = 1, \cdots, n_l\right\}, \tag{10}$$

that ensure the stability and safety of the linear dynamical system (2) while preserving some degree of optimality for the following cost

$$\mathcal{J} = \sum_{t=0}^{\infty} \mathbb{E}\big[x(t)^T Q x(t) + u(t)^T R u(t)\big], \tag{11}$$

where $Q \in \mathbb{R}^{n\times n} \succ 0, R \in \mathbb{R}^{m\times m} \succ 0$.

Theorem 2. Consider the system (2) and Assumptions (1) and (4). Then, the safety-parameterized optimal state feedback control gain K that solves Problem 2 is given by

$$K = F P_s^{-1}, \tag{12}$$

where $P_s \in \mathbb{R}^{n\times n}$ and $F \in \mathbb{R}^{m\times n}$ are the solutions (if they exist) to the following optimization problem

$$\begin{aligned}
\min_{\gamma, F, P_o, L, P_s, H} \quad & \alpha_1\gamma - \alpha_2 \log\det P_s && \text{(13a)}\\
\text{s.t.} \quad & \begin{bmatrix} P_o - I & AP_s + BF \\ * & H \end{bmatrix} \succeq 0, && \text{(13b)}\\
& P_o \succeq I, && \text{(13c)}\\
& \begin{bmatrix} \lambda P_s & AP_s + BF \\ * & P_s \end{bmatrix} \succeq 0, && \text{(13d)}\\
& \begin{bmatrix} H & P_s \\ * & P_o \end{bmatrix} \succeq 0, && \text{(13e)}\\
& \begin{bmatrix} L & F \\ * & H \end{bmatrix} \succeq 0, && \text{(13f)}\\
& W^T(l) P_s W(l) \leq 1, \quad l = 1, \cdots, n_l, && \text{(13g)}\\
& P_s \succeq 0, && \text{(13h)}\\
& tr(QP_o) + tr(RL) \leq \gamma, && \text{(13i)}
\end{aligned}$$

where $\alpha_1, \alpha_2 > 0$ are design parameters to tune the safety-optimality balance. Besides, the set associated with P_s is a conflict-free and performance-aware ellipsoid.

Proof. Consider the following LQR optimization problem (De Oliveira et al., 2002) for the system (2)

$$\begin{aligned}
\min_{\gamma, K, P_o, L} \quad & \gamma && \text{(14)}\\
\text{s.t.} \quad & (A + BK) P_o (A + BK)^T - P_o + I \preceq 0,\\
& P_o \succeq I,\\
& L - K P_o K^T \succeq 0,\\
& tr(QP_o) + tr(RL) \leq \gamma.
\end{aligned}$$

If one parameterizes the feedback gain K considering the $\lambda-$contractivity of the maximum size ellipsoidal safe set characterized by $\mathcal{E}(P_s, 0) = x(t)^T P_s^{-1} x(t)$ inside the given polyhedral safe set $\mathcal{X}_{ph}$ (10) as

$$\begin{aligned}
\min_{F, P_s} \quad & -\log\det P_s && \text{(15)}\\
\text{s.t.} \quad & \begin{bmatrix} \lambda P_s & (AP_s + BF)^T \\ * & P_s \end{bmatrix} \succeq 0,\\
& P_s \succ 0,\\
& W(l)^T P_s W(l) \leq 1, \quad l = 1, \cdots, n_l.
\end{aligned}$$

where $\lambda \in (0, 1)$ is the contractivity factor of the safe set defined by P_s and $F = KP_s$. Combining (14) and (15) leads to (13) with tuning parameters for the costs (α_1 and α_2) to balance safety and optimality. Also, the matrix H represents the relation between the Lyapunov stability matrix P_s and the controllability Gramian P_o. □

4. MOTION-PLANNING METHODOLOGY AND ALGORITHM

Invariant sets around equilibrium pairs $(\bar{x}_i, \bar{u}_i)$, $i = 1, 2, ..., |\mathcal{I}|$, decompose motion planning problem into regu-

lation subproblems (Problem 2), each steering the system between invariant sets while ensuring safety and performance. Solving these subproblems sequentially simplifies motion planning into components focusing on specific trajectory segments. This section introduces tools and methods to address all steps of Problem 1 combined.

4.1 General Properties of SODA-RRT Graph

At the execution step, the idea is for the control input $u(t)$ as in (8) at each time step t to be selected from a set of previously computed controllers satisfying safety and optimality locally. For $i \in \mathcal{I}$, where $\mathcal{I}$ is the index set of the local controllers, one has $(\bar{x}_i, \bar{u}_i)$ as an equilibrium pair. The equilibrium pair relates to a selected desired output $\bar{y}_i \in \mathcal{Y}$ with (7). In the vicinity of each equilibrium state $\bar{x}_i$, one can describe an associated ellipsoidal positive invariant set as $\mathcal{E}(P_i, \bar{x}_i)$ in (1). A control feedback gain K_i associated with each P_i guarantees the invariance of the ellipsoidal set $\mathcal{E}(P_i, \bar{x}_i)$ when applying $u_i(t) = K_i(x(t) - \bar{x}_i) + \bar{u}_i$. The method in Section 3 generates an ellipsoid with an associated control gain around each equilibrium pair. The graph generation rule connects a tail to a head if the control ellipsoid at the center of the head contains the tail, ensuring that the system's states stay within the ellipsoids during the transitions, as shown in Fig. 1. This rule allows us to find a series of invariant

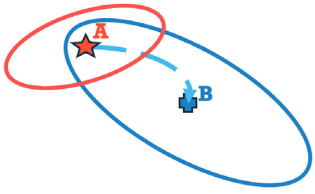

Fig. 1. System trajectory from tail (A) to head (B) while staying inside the control gain's blue ellipsoid.

ellipsoids, overlapping with each other, by which one can perform a gain scheduling that results in a safe and optimal transition from a starting point to a target through the invariant sets. Specifically, the motion-planning algorithm picks a control input in the form of (8) at each step. A graph search constructs this set. For all edges of the graph $\mathcal{G}(\mathcal{I}, E)$, the indices $\mathcal{I}$ determine the control gains to be used. If there is an ellipsoid containing two vertices (i, j), there can be an edge between the two vertices and it is possible to transition between vertices. One can switch the controller from the one active around vertex i once the states enter the ellipsoid of the next vertex j.

4.2 Graph Generation

Below is the Algorithm used to generate the graph $\mathcal{G} = (\mathcal{I}, E)$ based on the general specifications that are brought in Subsection 4.1. If this algorithm is run successfully all aspects of Problem 1 are answered.

In Algorithm 1, A, B are the system (2) matrices, r_f is the radius of the circle around the target point, $\mathbb{P}$ is the probability of sampling from the target area or the rest of the output space, and o is the output space dimension. Nodes are specified with q, including q_{init}, q_{samp}, q_{app}, and $q_{nearest}$ as the initial, sampled, approved,

Algorithm 1:
$\mathcal{G} = (\mathcal{I}, E) \leftarrow$ `SODA-RRT`$(q_{init}, q_{target}, \lambda, A, B, R, \mathbb{P})$

$q_{app} \leftarrow q_{init}$
$\mathcal{G} \leftarrow$ `InitGraph`()
$\mathcal{G} \leftarrow$ `AddNode`$(\emptyset, q_{init}, \mathcal{G})$
$\mathcal{G} \leftarrow$ `AddCtrl`$(\emptyset, \emptyset, q_{init}, \mathcal{G})$
while *EucDist*$(q_{app}, q_{target}) \leq r_f$ **do**
  $q_{samp} \leftarrow$ `RandSamp`$(q_{init}, q_{target}, \mathbb{P}, o)$
  $b_{samp} \leftarrow$ `FindBorder`$(q_{samp}, \mathcal{Y}_1, \ldots, \mathcal{Y}_j)$
  $P_{samp}, K_{samp} \leftarrow$ `GenEl`$(A, B, \lambda, q_{samp}, b_{samp})$
  $InsideEl, q_{nearest} \leftarrow$ `ChParent`$(q_{samp}, P_{samp}, \mathcal{G})$
  if $InsideEl == True$ **then**
    $q_{app} \leftarrow q_{samp}$
    $\mathcal{G} \leftarrow$ `AddNode`$(q_{nearest}, q_{app}, \mathcal{G})$
    $\mathcal{G} \leftarrow$ `AddCtrl`$(K_{samp}, P_{samp}, q_{app}, \mathcal{G})$
  end
end

and the nearest node selected for parent, respectively. `EucDist` finds the Euclidean distance between its inputs. `GenEl`$(A, B, \lambda, q_{samp}, b_{samp})$ generates a positive invariant ellipsoidal set with q_{samp} at its center. After finding the convex polyhedral set around each sampled location using Algorithm 3 (outputting b_{samp}), LMIs (13) are solved by transferring the coordinates of q_{samp} to the origin to find the largest conflict-free ellipsoidal set P_{samp} and feedback gain K_{samp}.

$$Cost_E \leftarrow \texttt{ElCost}(q, q_c, P) = (q - q_c)^T P^{-1} (q - q_c). \quad (16)$$

The matrix P_{samp} obtained from (13) can be used as a measure of proximity. The `GenEl`$(A, B, \lambda, q_{samp}, b_{samp})$ function generates an ellipsoidal set $\mathcal{E}(P_{samp}, q_{samp})$ in the state space $\mathbb{R}^n$. However, since the motion planning is performed in the output space $\mathbb{R}^o$, it is necessary to project this ellipsoidal set onto the lower-dimensional output space. The $P_{Proj} \leftarrow$ `Proj`(P) function accomplishes this task by mapping the ellipsoidal set from the state space to the output space.

Algorithm 2:
$InsideEl, q_{nearest} \leftarrow$ `ChParent`$(q_{app}, P_{samp}, \mathcal{G})$

$Cost_G \leftarrow \infty$
$P_{proj} \leftarrow$ `Proj`(P_{samp})
$InsideEl \leftarrow False$
$q_{nearest} \leftarrow \emptyset$
for $q \in \mathcal{G}$ **do**
  $Cost_E =$ `ElCost`(q, q_{app}, P_{proj}) **if**
  $Cost_E < Cost_G$ *and* $Cost_E \leq 1$ **then**
    $Cost_G \leftarrow Cost_E$
    $q_{nearest} \leftarrow q$
    $InsideEl \leftarrow True$
  end
end
return $InsideEl, q_{nearest}$

Algorithm 2 proposes a novel method for selecting the parent node based on the ellipsoidal cost (16). It iterates through all vertices of $\mathcal{G}$ and identifies the vertex with the minimum cost of traversing to q_{samp}. Unlike RRT, which connects vertices based on proximity, our approach establishes connections based on set invariance and optimality

criteria derived from the system dynamics, ensuring a safe and optimized trajectory.

4.3 Finding Border

We construct a convex polyhedral set at each step after sampling the output space. In our paper, we focus on finding a convex set around each sampled point by leveraging prior knowledge of the convex sets inside the nonconvex set, as shown in Fig. 2, where $\mathcal{Y} = \mathcal{Y}_1 \cup \ldots \cup \mathcal{Y}_4$. After sampling, the location is tagged based on its polyhedron, and its boundary is defined by determining the distance to the polyhedron's borders. To ensure that the constraint sets overlap, the tag may change with a user-defined probability among the convex sets into which the location falls.

Algorithm 3:
$b_{samp} \leftarrow$ `FindBorder`$(q_{samp}, \mathcal{Y}_1, \ldots, \mathcal{Y}_\phi)$

Check which constraint set $\mathcal{Y}_i$ for $i = 1, ..., \phi$ sampled location q_{samp} falls into
if q_{samp} *has multiple tags* **then**
  $\rho =$ `rand` $\{1, ..., \tau\}$
  Assign the boundaries of $\mathcal{Y}_\rho$ to sampled location q_{samp}
else
  Assign the boundaries of the found $\mathcal{Y}$ to sampled location q_{samp}

In Algorithm 3, ϕ and τ denote the number of constraint sets in the nonconvex output space and those containing a sampled location, respectively.

4.4 Execution

After sampling the space, performing a graph search, and finding a viable path from the initial state x_{init} to the target state x_{target}, our algorithm will run Algorithm 4. For each invariant ellipsoid $\mathcal{E}_i$, we define $P_i, K_i, \bar{x}_i, \bar{u}_i$ as the set parameter, feedback gain, equilibrium state, and feedforward term, respectively. Algorithm 4 searches for path in the graph $\mathcal{G} = (\mathcal{I}, E)$ for a sequence of invariant sets $\mathcal{E}_0, ..., \mathcal{E}_N \in \mathcal{I}$, where $\mathcal{E}_0$ contains the initial state x_{init} and $\mathcal{E}_N$, x_{target} which corresponds to the final output y_{target}. At each time step $t \in \mathbb{N}$, Algorithm 4 uses the control input $u(t) = K_{\mathcal{E}_t}\big(x(t) - \bar{x}_{\mathcal{E}_t}\big) + \bar{u}_{\mathcal{E}_t}$ where $\bar{x}_{\mathcal{E}_t} = [q_{\mathcal{E}_t}\ 0]^T$. The control parameters are updated when the state $x(t)$ enters a new ellipsoid $\mathcal{E}_t = \mathcal{E}_{next}$. It should be mentioned that the new ellipsoid is not necessarily the next invariant set after $\mathcal{E}_i$ and thus at each step we need to check all the future possible invariant sets in $\mathbb{S}_{future} := \{\mathcal{E}_{i+1}, ..., \mathcal{E}_N\}$.

5. SIMULATION EXAMPLE

This section presents an application [1] of `SODA-RRT` to the problem of spacecraft docking maneuvers. The relative dynamics model of a pair of spacecraft is presented below (Wie, 1998).

$$\begin{aligned} \ddot{z}_1 &= 3r^2 z_1 + 2r\dot{z}_2 + v_1 \\ \ddot{z}_2 &= -2r\dot{z}_1 + v_2 \end{aligned} \tag{17}$$

[1] https://github.com/NarimanNiknejad/SODA-RRT

Algorithm 4: `Execution`()

initial control parameters $K = K_{\mathcal{E}_0}, \bar{x} = \bar{x}_{\mathcal{E}_0}, \bar{u} = \bar{u}_{\mathcal{E}_0}, y(0) = Cx_{init}, x(0) = x_{init}$
$t = 0$
while *EucDist*$(y(t), q_{target}) \leq r_f$ **do**
  for $\mathcal{E}_s \in \mathbb{S}_{future}$ **do**
    if $y(t) \in \mathcal{E}_s$ **then**
      update gain and equilibrium pair $K = K_{\mathcal{E}_s}, \bar{x} = \bar{x}_{\mathcal{E}_s}, \bar{u} = \bar{u}_{\mathcal{E}_s}$
  generate control input $u(t) = K(x(t) - \bar{x}) + \bar{u}$
  let $x(t+1) = Ax(t) + Bu(t)$ & $y(t) = Cx(t)$
  $t = t + 1$

where the states of the system $x = [z_1, z_2, \dot{z}_1, \dot{z}_2]^T$ has relative radial and orbital positions and velocities, the inputs $u = [v_1, v_2]^T$ are the normalized thrusts in the direction of radial and orbital velocities, and $y(t)$ are the system outputs. Linearizing the model about a circular orbit yields $r = 1.1 \times 10^{-1} sec^{-1}$. One can put (17) in the state space format. The system matrices are as follows

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 3r^2 & 0 & 0 & 2r \\ 0 & 0 & -2r & 0 \end{bmatrix}, B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}, C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{18}$$

The dynamics of the spacecraft (18) is discretized with the zero-order hold technique and with a sample time of 30 *sec*.

We examine the following scenario concerning planning a maneuver around a piece of debris, as depicted in Fig. 2. The debris, a square of $16\,m$, is positioned at $[25, 25]^T\,m$. The output set $\mathcal{Y}$ is derived from the difference between the bounding box $[-50, 50] \times [-50, 50]\,m$ and the debris set. To form the component sets $\mathcal{Y}_1, \ldots, \mathcal{Y}_4$, which constitute the output set $\mathcal{Y} = \mathcal{Y}_1 \cup \ldots \mathcal{Y}_4$, each constraint defining the debris set is inverted and intersected with the bounding box. The collection of convex subsets $\mathcal{Y}_1, \ldots, \mathcal{Y}_4$ is also visualized in Fig. 2.

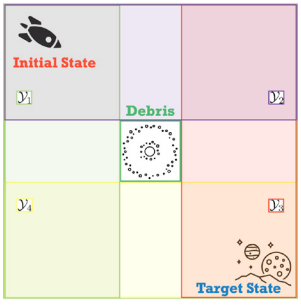

Fig. 2. The nonconvex set $\mathcal{Y} = \mathcal{Y}_1 \cup \ldots \mathcal{Y}_4$ can be covered by four convex sets shown in different colors.

The spacecraft begins at $y_i = [-30, 30]^T\,m$ and the destination is located at $y_f = [30, -30]^T\,m$. The contraction ratio is set to $\lambda = 0.95$. The LQR weightings are chosen as $R = 10^2 \times I_2$, $Q = diag[10^{-4}, 10^{-4}, 10^2, 10^2]$.

The probability of sampling from the target area with $r_f = 5m$ is set to $\mathbb{P} = 10\%$. We evaluate `SODA-RRT` against LQR control and benchmark it against `LQR-RRT`,

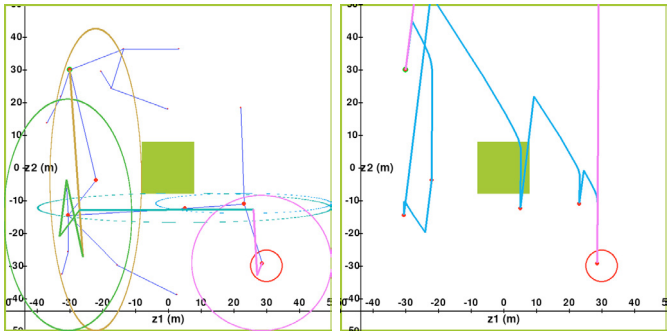

Fig. 3. (a) SODA-RRT Algorithm; ellipsoidal sets and trajectories (associated ones are shown with the same color); and (b) LQR with RRT waypoints (blue) and LQR (pink) trajectories.

which uses `SODA-RRT` vertices but LQR gains. Simulations reveal that LQR fails to ensure safety, while `LQR-RRT` fails to evade collisions due to relying on LQR gain lacking inherent safety guarantees. Moreover, `LQR-RRT` may have a higher trajectory cost than `SODA-RRT` since it sequentially follows waypoints, unlike Algorithm 4, which selects gains based on safe ellipsoidal sets. We conduct $Ts = 200$ simulations and compute the average trajectory cost using the following quadratic cost

$$\bar{\mathcal{J}}_{Ts} = \frac{1}{Ts}\sum_{k=0}^{Ts}\sum_{t=0}^{T}\mathbb{E}\big[x_k(t)^T Q x_k(t) + u_k(t)^T R u_k(t)\big].$$

Table 1 shows that `SODA-RRT` provides safety and potentially can reduce trajectory cost compared to regular `RRT` with LQR gain, due to its overlapping positive invariant sets. Fig. 3 illustrates that the trajectories remain within the ellipsoidal sets, ensuring safety without breaching boundaries or colliding with the obstacle.

Table 1. Different Motion Planning Algorithms Comparison

Algorithm	**Safety Breach**	**Average Cost** ($\bar{\mathcal{J}}_{200}$)
`SODA-RRT`	0/200	5.008×10^{05}
`LQR-RRT`	173/200	1.385×10^{06}
`LQR`	200/200	2.105×10^{05}

6. CONCLUSION

This paper presents a novel motion planning approach that generates collision-free paths with guaranteed safety and optimality using interconnected invariant sets. The proposed algorithm, `SODA-RRT`, advances existing methods by constructing the largest possible conflict-free invariant sets through a single-shot optimization process, balancing safety and performance. This technique effectively trades off between collision avoidance and trajectory optimality. The algorithm's efficacy is demonstrated through spacecraft motion planning scenarios involving debris avoidance. Future research directions include real-time implementation and the incorporation of data- and physics-informed sets to handle uncertain systems, further enhancing the method's robustness and applicability to complex, dynamic environments.

REFERENCES

Althoff, M. and Dolan, J.M. (2014). Online verification of automated road vehicles using reachability analysis. *IEEE Transactions on Robotics*, 30(4), 903–918.

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

Blanchini, F. (1999). Set invariance in control. *Automatica*, 35(11), 1747–1767.

Borrelli, F., Bemporad, A., and Morari, M. (2017). *Predictive control for linear and hybrid systems*. Cambridge University Press.

Danielson, C., Weiss, A., Berntorp, K., and Di Cairano, S. (2016). Path planning using positive invariant sets. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, 5986–5991. IEEE.

De Oliveira, M.C., Geromel, J.C., and Bernussou, J. (2002). Extended H 2 and H norm characterizations and controller parametrizations for discrete-time systems. *International journal of control*, 75(9), 666–679.

Ge, S.S. and Cui, Y.J. (2002). Dynamic motion planning for mobile robots using potential field method. *Autonomous robots*, 13, 207–222.

Karaman, S. and Frazzoli, E. (2011). Sampling-based algorithms for optimal motion planning. *The international journal of robotics research*, 30(7), 846–894.

Kavraki, L.E. and LaValle, S.M. (2016). Motion planning. In *Springer handbook of robotics*, 139–162. Springer.

Kavraki, L.E., Svestka, P., Latombe, J.C., and Overmars, M.H. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4), 566–580.

Kousik, S., Vaskov, S., Bu, F., Johnson-Roberson, M., and Vasudevan, R. (2020). Bridging the gap between safety and real-time performance in receding-horizon trajectory design for mobile robots. *The International Journal of Robotics Research*, 39(12), 1419–1469.

LaValle, S.M. (2006). *Planning algorithms*. Cambridge University Press.

LaValle, S.M. and Kuffner, J.J. (2001). Randomized kinodynamic planning. *The International Journal of Robotics Research*, 20(5), 378–400.

Mataric, M.J. (1992). Behavior-based control: Main properties and implications. In *Proceedings, IEEE International Conference on Robotics and Automation, Workshop on Architectures for Intelligent Control Systems*, 46–54. Citeseer.

Mayne, D.Q., Rawlings, J.B., Rao, C.V., and Scokaert, P.O. (2000). Constrained model predictive control: Stability and optimality. *Automatica*, 36(6), 789–814.

Niknejad, N. and Modares, H. (2023). Physics-informed data-driven safe and optimal control design. *IEEE Control Systems Letters*.

Rakovic, S.V., Kerrigan, E.C., Kouramas, K.I., and Mayne, D.Q. (2005). Invariant approximations of the minimal robust positively invariant set. *IEEE Transactions on Automatic Control*, 50(3), 406–410.

Schouwenaars, T., De Moor, B., Feron, E., and How, J. (2001). Mixed integer programming for multi-vehicle path planning. In *European Control Conference (ECC)*, 2603–2608. IEEE.

Wie, B. (1998). *Space vehicle dynamics and control*. AIAA.