MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

Invariant-set motion-planner for unicycle dynamics under closed-loop feedback linearization

Leung, Jordan; Di Cairano, Stefano

TR2026-064 May 28, 2026

Abstract

This paper develops an invariant-set motion- planner (ISMP) for vehicles with unicycle-like dynamics. ISMPs operate by determining a sequence of obstacle-free positively invariant (PI) sets for the closed-loop system that connect the initial and target state. In this paper, we derive PI sets under a dynamic feedback linearization law. We demonstrate that the PI sets are simple in geometry and we provide an efficient algorithm for obstacle-free scaling. The PI sets are then used to develop a graph-based search for finding an obstacle-free path between two states. The approach is demonstrated in an obstacle-rich simulated environment.

American Control Conference (ACC) 2026

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

Invariant-set motion-planner for unicycle dynamics under closed-loop feedback linearization

Jordan Leung[1*] and Stefano Di Cairano[1]

***Abstract*— This paper develops an invariant-set motion-planner (ISMP) for vehicles with unicycle-like dynamics. ISMPs operate by determining a sequence of obstacle-free positively invariant (PI) sets for the closed-loop system that connect the initial and target state. In this paper, we derive PI sets under a dynamic feedback linearization law. We demonstrate that the PI sets are simple in geometry and we provide an efficient algorithm for obstacle-free scaling. The PI sets are then used to develop a graph-based search for finding an obstacle-free path between two states. The approach is demonstrated in an obstacle-rich simulated environment.**

I. Introduction

Invariant-set motion-planners (ISMPs) are algorithms for generating dynamically feasible collision-free trajectories for autonomous vehicles and robots between an assigned initial and target state [1]–[8]. The trajectories are generated by identifying an overlapping sequence of obstacle-free positive invariant (PI) sets that connect the initial and target states. Dynamic feasibility is achieved by the controller that defines the PI sets for the closed-loop system, while obstacle avoidance is achieved by ensuring the PI sets do not overlap with obstacles. The motion planning problem is then abstracted to a graph search, where each vertex corresponds to an equilibrium of the system with an associated PI set.

The primary advantage of ISMPs is that the majority of the computational effort (i.e., the graph construction) can be performed offline. Then, the online path planning procedure only requires execution of a graph search algorithm for which collision checking is not needed since it is implied by the PI sets. In addition, the planned trajectories are inherently robust [6] due to the use of feedback control and the output space does not need to be densely sampled since each PI set can cover large volumes of the space.

ISMPs share similarities with other graph-based motion-planners. For example, closed-loop rapidly-exploring random trees (RRTs) [9], [10] construct graphs by simulating motion between two references; reachability-based methods [11], [12] construct graphs by checking reachability between two vertices; and LQR-trees [13] construct graphs by checking feasibility of a two-point boundary value problem between two vertices. We refer the reader to [1]–[8] for a more comprehensive comparison of ISMPs to these methods.

This paper develops an ISMP for the nonlinear unicycle model for applications to autonomous ground robots. An ISMP for unicycle-like vehicles [7] has been developed using a polar coordinated based feedback law from [14]. However, the obstacle-free construction of PI sets in [7] is complicated by the fact that the Lyapunov function is defined in polar coordinates. Hence, directly assessing whether or not a sub-level set of the Lyapunov function is obstacle-free is a non-convex problem. This is addressed in [7] by relying on convex overapproximations in Cartesian coordinates, but this method does not fully leverage the simplicity of the two-dimensional environment since it relies on dual problems defined in higher dimensions spaces.

In our proposed approach, we adopt a dynamic feedback linearization strategy [15] that yields semi-elliptical (i.e., an ellipse intersected with a half-space) PI sets in the Cartesian coordinates of the system. As a result, obstacle-free scalings for these PI sets can be computed by solving a two-dimensional quadratic program using simple geometric approaches. Our method also allows for backward motion to be penalized (or forbidden) through definition of the edge weights in the graph. Moreover, the velocity signal over the planned trajectory is continuous everywhere aside from transitions between forward and backward motion.

Notation: Let $(x, y) = [x^T \ y^T]^T$ for $x, y \in \mathbb{R}^n$. Let $\mathbb{S}^n_{\succ 0}$ represent the set of symmetric positive definite matrices in $\mathbb{R}^{n\times n}$. Let $|\mathcal{S}|$ denote the cardinality of a finite-set $\mathcal{S}$. Let $\mathbb{N}_{[1,N]} = \mathbb{N} \cap [1, N]$. For two angles $\phi_1, \phi_2 \in (-\pi, \pi]$, let $\phi_1 \oplus \phi_2 := \mathbb{P}_\pi(\phi_1 + \phi_2)$ and $\phi_1 \ominus \phi_2 := \mathbb{P}_\pi(\phi_1 - \phi_2)$, where $\mathbb{P}_\pi$ represents the wrapping back onto $(-\pi, \pi]$. As usual, the 2D rotation matrix is:

$$R(\phi) = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix}.$$

For a set $\mathcal{S} = \{(x, y, \cdot) \in \mathbb{R}^n \mid ...\}$ with $n > 2$, let $\mathcal{S}_{xy} = \{(x, y) \mid \exists (x, y, \cdot) \in \mathcal{S}\}$ be the projection on (x, y). A graph $\mathcal{G} = (\mathcal{I}, \mathcal{E}, \mathcal{W})$ is a set of vertices $\mathcal{I}$ together with a set of ordered pairs $\mathcal{E} \subseteq \mathcal{I} \times \mathcal{I}$ called edges and a set of weights $\mathcal{W}$ associated with each edge (i.e., $|\mathcal{E}| = |\mathcal{W}|$). A path is a sequence of adjacent vertices. A graph search is an algorithm for finding a path through a graph. The cost of a path is the sum of the weights associated with each edge in the path. A shortest path is a path with the lowest possible cost.

II. Background and Problem Statement

This paper addresses the planning and control for vehicles with unicycle-like dynamics. Let $q_g = (x_g, y_g, \phi_g)$ represent the position and orientation of the vehicle in a fixed global coordinate system. We define the following local coordinate system associated with a target equilibrium $\bar{q} = (\bar{x}, \bar{y}, \bar{\phi})$.

[1] J. Leung and S. Di Cairano are with Mitsubishi Electric Research Laboratories. Emails: {leung, dicairano}@merl.com.

[*] Corresponding author.

Definition 1. *A vector $q = (x, y, \phi)$ is said to be the coordinates of the vehicle expressed in the frame defined by the equilibrium $\bar{q} = (\bar{x}, \bar{y}, \bar{\phi})$ if*

$$\begin{bmatrix} x \\ y \end{bmatrix} := \mathcal{R}_z(-\bar{\phi}) \begin{bmatrix} x_g - \bar{x} \\ y_g - \bar{y} \end{bmatrix}, \quad \phi := \phi_g \ominus \bar{\phi}. \tag{1}$$

We use the notation $q|_{\bar{q}}$ to denote this relationship explicitly when necessary.

The dynamics of the vehicle in the coordinates of $\bar{q}$ are

$$\dot{q} = \begin{bmatrix} v\cos(\phi) \\ v\sin(\phi) \\ \omega \end{bmatrix}, \tag{2}$$

where $q = (x, y, \phi)$ is the state, $u = (v, \omega)$ are the control inputs, and $q = (x, y, \phi) = 0 \iff q_g = \bar{q}$.

A. Stabilization of (2)

The ISMP is developed for the closed-loop dynamics of the unicycle under the dynamic feedback linearization strategy in [15], which is described next. We define an output vector $\eta = (x, y)$ and write its second derivative:

$$\ddot{\eta} = \begin{bmatrix} \dot{v}\cos\phi - v\dot{\phi}\sin\phi \\ \dot{v}\sin\phi + v\dot{\phi}\cos\phi \end{bmatrix} = \begin{bmatrix} \cos\phi & -v\sin\phi \\ \sin\phi & v\cos\phi \end{bmatrix} \begin{bmatrix} a \\ \omega \end{bmatrix},$$

where $a := \dot{v}$. Under the assumption that $v \neq 0$, the matrix is non-singular and $\ddot{\eta}$ can be re-parameterized in terms of so-called virtual inputs $\mu \in \mathbb{R}^2$:

$$\begin{aligned} &\ddot{\eta} = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix} =: \mu, && \dot{v} = a, \\ &a = \mu_1\cos\phi + \mu_2\sin\phi, && \omega = \frac{\mu_2\cos\phi - \mu_1\sin\phi}{v}. \end{aligned} \tag{3}$$

The velocity command v is an internal state of the dynamic feedback law (3). The singularity at $v = 0$ can be avoided by appropriate selection of feedback gains and the initialization $v(0)$. Let $\mathcal{Q} := \mathbb{R}^2 \times (-\pi, \pi]$ and define

$$\mathcal{Q}^* = \left\{ q \in \mathcal{Q} \;\middle|\; \begin{array}{c} (x = 0, \cos\phi \geq 0) \text{ OR} \\ (y = 0, \cos(\phi) = -1) \end{array} \right\}.$$

as the subset of $\mathcal{Q}$ that requires special attention. The remaining part $\mathcal{Q}\backslash\mathcal{Q}^*$ is partitioned into two regions:

$$\mathcal{Q}^r = \{q \in \mathcal{Q}\backslash\mathcal{Q}^* \mid x \geq 0\}, \quad \mathcal{Q}^l = \{q \in \mathcal{Q}\backslash\mathcal{Q}^* \mid x < 0\}.$$

We now restate the primary result from [15].

Result 1. *([15, Proposition 1]) Consider the unicycle system (2) and the control law (3) with*

$$\mu_1 = -k_{px}x - k_{dx}\dot{x}, \quad \mu_2 = -k_{py}y - k_{dy}\dot{y}. \tag{4}$$

The closed-loop is exponentially convergent from any initial condition $q(0) = q_0 \in \mathcal{Q}\backslash\mathcal{Q}^*$ to the origin and $x(t) \neq 0$ and $v(t) \neq 0$ for all $t < \infty$, if:

A1 *The control gains $k_{pi} > 0$, $k_{di} > 0$ $(i = x, y)$ satisfy*

$$k_{dx}^2 - 4k_{px} = k_{dy}^2 - 4k_{py} > 0$$
$$k_{dy} - k_{dx} > 2\sqrt{k_{dy}^2 - 4k_{py}}.$$

A2 *The velocity initialization $v(0) = v_0$ satisfies*

$$v_0 < 0 \text{ (backward motion) if } q_0 \in \mathcal{Q}^r$$
$$v_0 > 0 \text{ (forward motion) if } q_0 \in \mathcal{Q}^l,$$

and $v_0 \neq 2(k_{dy} - k_{dx})^{-1}(k_{px}x_0\sin\phi_0 - k_{py}y_0\cos\phi_0)$.

A3 *The longitudinal position x_0 and speed $\dot{x}_0$ satisfy*

$$\dot{x}_0 \geq \lambda_{1x}x_0 \text{ if } q_0 \in \mathcal{Q}^r, \quad \dot{x}_0 \leq \lambda_{1x}x_0 \text{ if } q_0 \in \mathcal{Q}^l,$$

where $\lambda_{1x} = \frac{1}{2}(-k_{dx} - \sqrt{k_{dx}^2 - 4k_{px}})$.

B. Problem Statement and ISMP Approach

We can now precisely define the motion-planning problem.

Definition 2. *An obstacle $\mathcal{O}_k = \{p_g \in \mathbb{R}^2 \mid A_k p_g \leq b_k\}$ is a compact polyhedral set[1] of global coordinates $p_g = (x_g, y_g)$ that the vehicle is forbidden to occupy. The notation $\mathcal{O}_k|_{\bar{q}}$ is used to represent the coordinates of the obstacle expressed in the frame defined by an equilibrium $\bar{q}$, i.e.,*

$$\mathcal{O}_k|_{\bar{q}} = \{p \mid A_k R(\bar{\phi})p \leq b_k - A_k\bar{p}\},$$

Problem 1. *Consider system (2) in closed-loop with (3)-(4) in global coordinates:*

$$\begin{aligned} &\dot{x}_g(t) = v(t)\cos\phi_g(t), && \dot{v}(t) = a(t), \\ &\dot{y}_g(t) = v(t)\sin\phi_g(t), && \begin{bmatrix} a(t) \\ \omega(t) \end{bmatrix} = \kappa(q_g(t), \bar{q}(t), v(t)), \\ &\dot{\phi}_g(t) = \omega(t), \end{aligned} \tag{5}$$

where $\kappa : (q_g, \bar{q}, v) \mapsto (a, \omega)$ represents execution of the feedback law (3)-(4) given the local coordinates $q|_{\bar{q}}$. Let $\bar{q}(t)$ be a piecewise constant signal specifying a target equilibrium, i.e., for some $\{(\bar{q}_i, t_i)\}_{i=0}^N$, $\bar{q}(t) = \bar{q}_i$ if $t \in [t_i, t_{i+1})$ and $\bar{q}(t) = \bar{q}_N$ if $t \geq t_N$. Let $\{\mathcal{O}_k\}_{k=1}^{N_o}$ represent the set of obstacles. The planning task is to determine an equilibria sequence $\mathcal{P} = \{\bar{q}_i\}_{i=0}^N$ such that there exist a switching time sequence $\{t_i\}_{i=0}^N$ so that the system in (5) satisfies:

1) *Obstacle avoidance: $(x_g(t), y_g(t)) \notin \mathcal{O}_k$ for all $k \in \mathbb{N}_{[1,N_o]}$ and for all $t \geq 0$,*
2) *Convergence to the target: $\lim_{t\to\infty} q_g(t) = \bar{q}_N = \bar{q}_{\text{target}}$ for some target equilibrium $\bar{q}_{\text{target}}$.*

Remark 1. *Dynamic feedback linearization [15] is adopted since the resulting closed-loop PI sets have a simple representation in Cartesian coordinates and the closed-loop exhibits good performance[2]. In contrast, the polar coordinate law in [14] obtains good performance, but does not have simple PI sets in Cartesian coordinates; the backstepping technique in [17] derives a Lyapunov function in Cartesian coordinates, but relies on non-quadratic time-dependent terms that make it difficult to use in an ISMP; while the time-varying control law discussed in [18, Section 1.3] results 2-norm ball PI sets in Cartesian coordinates, but yields very slow asymptotic convergence of the closed-loop system.*

[1]For applications relying on real-time sensing, we assume a preprocessing routine has been performed to bound point cloud data with polyhedral sets. In addition, each obstacle may represent the Minkowski sum of a polyhedral forbidden zone and a polyhedral set accounting for the vehicle size.

[2]We refer the reader to [16] for a comparison of the practical performance of some set-point regulation strategies for unicycle dynamics

In addition, since v is an internal state, the commanded velocity under step changes in equilibria is continuous. This is not the case for the controllers in [14], [17], [18].

The ISMP solves Problem 1 as follows:

Algorithm 1 Invariant Set Motion Planner

1: Generate a grid of equilibrium points $\Sigma = \{\bar{q}_i\}_{i=1}^{M}$ such that $\{(\bar{x}_i, \bar{y}_i)\}$ spans the anticipated workspace and $\{\bar{\phi}_i\}$ spans $(-\pi, \pi]$.
2: For each equilibrium point, construct at least one PI set that is scaled so that its projection onto the (x, y)-plane does not intersect with any obstacle $\mathcal{O}_k$.
3: Build a graph $\mathcal{G} = (\mathcal{I}, \mathcal{E}, \mathcal{W})$ where the vertices $i \in \mathcal{I}$ map to equilibria in Σ, and the edges $(i, j) \in \mathcal{E}$ indicate that the system will enter the obstacle-free invariant set for the equilibrium corresponding to j while tracking the equilibrium corresponding to i.
4: Determine a path in the graph between vertices corresponding to the initial and target equilibria.

The remainder of the paper is devoted to precisely defining how Steps 2, 3, and 4 are conducted.

III. Obstacle-free Invariant Sets for the Unicycle

We begin by deriving the invariant sets used in our approach. Thereafter, we show how these sets can be scaled so that their projections do not intersect with any obstacles.

A. Positive Invariant Sets

Under the conditions in Result 1, for any $Q \in \mathbb{S}^4_{\succ 0}$ and $P \in \mathbb{S}^4_{\succ 0}$ such that $A_K^T P + P A_K = -Q$, where

$$A_K = \begin{bmatrix} 0 & I \\ -K_p & -K_d \end{bmatrix},$$

$K_p = \text{diag}(k_{px}, k_{py})$, and $K_d = \text{diag}(k_{dx}, k_{dy})$, the closed-loop satisfies $\dot{V}(z) = -z^T Q z$ with $V(z) := z^T P z$ and $z := (x, y, \dot{x}, \dot{y})$. We define the following set-valued map,

$$\mathcal{Z}(c) = \{(x, y, \phi, v) \mid V(z) \leq c, z = (x, y, v\cos\phi, v\sin\phi)\},$$

and the sets associated with the restrictions in Result 1,

$$\mathcal{Q}^l_{\text{ext}} = \left\{(q, v) \mid q \in \mathcal{Q}^l, v > 0, v\cos\phi \leq \lambda_{1x} x\right\},$$
$$\mathcal{Q}^r_{\text{ext}} = \left\{(q, v) \mid q \in \mathcal{Q}^r, v < 0, v\cos\phi \geq \lambda_{1x} x\right\}.$$

Then, the following sets are PI for all $c > 0$:

$$\mathcal{Z}^l(c) := \mathcal{Z}(c) \cap \mathcal{Q}^l_{\text{ext}}, \quad \mathcal{Z}^r(c) := \mathcal{Z}(c) \cap \mathcal{Q}^r_{\text{ext}}. \tag{6}$$

B. Obstacle-free Invariant Set Scaling

For a target equilibrium $\bar{q}$, we now describe how to choose the scaling $c > 0$ so that the projection of the PI set, i.e.,

$$\mathcal{Z}^l_{xy}(c) := \{(x, y) \mid \exists(\phi, v) \text{ such that } (x, y, \phi, v) \in \mathcal{Z}^l(c)\},$$

does not intersect any obstacle. We first consider the PI sets for forward motion. Afterwards, we show that the PI sets for backward motion can be scaled using symmetry.

The projection $\mathcal{Z}^l_{xy}(c)$ is made obstacle-free by ensuring that the following slight overapproximation is obstacle-free: $\mathcal{S}^l(c) := \mathcal{Z}_{xy}(c) \cap \{(x, y) \mid x < 0\}$, where

$$\mathcal{Z}_{xy}(c) = \{p \in \mathbb{R}^2 \mid p^T P_{xy} p \leq c\},$$
$$P_{xy} = P_{11} - P_{12} P_{22}^{-1} P_{12}^T, \quad P =: \begin{bmatrix} P_{11} & P_{12} \\ P_{12}^T & P_{22} \end{bmatrix}.$$

Note that $\mathcal{S}^l(c) \supseteq \mathcal{Z}^l_{xy}(c)$ by definition. For an obstacle $\mathcal{O}_k$ following Definition 2, the smallest value for which $\mathcal{S}^l(c) \cap \mathcal{O}_k|_{\bar{q}} \neq \emptyset$ is given by:

$$\min_{p=(x,y)} \; p^T P_{xy} p \tag{7a}$$
$$\text{s.t.} \;\; A_k R(\bar{\phi}) p \leq b_k - A_k \bar{p}, \text{ and } x \leq 0. \tag{7b}$$

Hence, any $c > 0$ smaller than the optimal value of (7) satisfies $\mathcal{S}^l(c) \cap \mathcal{O}_k|_{\bar{q}} = \emptyset$. Note that this procedure can simply be repeated to account for multiple obstacles.

Remark 2. *Simple geometries are usually used to represent obstacles. In such cases, the optimum to* (7) *can be computed efficiently by first checking the first-order optimality conditions at the minimum-cost vertex*[3] *of the polytope defined by* (7b)*. If this is not an optimum, then one can search for a stationary point along each facet of* (7b)*. The worst-case complexity of solving* (7) *in this manner is proportional to the number of facets used to represent the obstacle. Hence, the two-dimensional nature of* (7) *is a notable advantage when compared to the dual problem in [7, Equation 14], whose dimension is equivalent to the number of facets in $\mathcal{O}_k$.*

Similarly, a scaling for backward motion to $\bar{q}$ to be safe for all $\xi(0)|_{\bar{q}} \in \mathcal{Z}^r(c)$ can be determined by solving

$$\min_{p=(x,y)} \; p^T P_{xy} p \tag{8a}$$
$$\text{s.t.} \;\; A_k R(\bar{\phi}) p \leq b_k - A_k \bar{p}, \text{ and } x \geq 0, \tag{8b}$$

which is equivalent to (7) for the opposite orientation.

Proposition 1. *The solution to* (8) *for $\bar{q} = (\bar{x}, \bar{y}, \bar{\phi})$ is equivalent to the solution of* (7) *for $(\bar{x}, \bar{y}, \bar{\phi} + \pi)$.*

Hence, the scaling required for safe backward connections can be determined automatically while determining the scaling for forward connections assuming that the set of sampled equilibrium always includes $(\bar{x}, \bar{y}, \bar{\phi} + \pi)$ for each $(\bar{x}, \bar{y}, \bar{\phi})$.

IV. Directed Graph Construction for the Invariant Set Motion Planner

Let $\Sigma = \{q_i\}_{i=1}^{M}$ represent the set of sampled equilibria for the ISMP graph. Assume that we have solved (7) for each obstacle-equilibrium pair so that each equilibrium is associated with a tuple (c_i^l, c_i^r) such that $\mathcal{Z}^l(c_i^l) \cap \mathcal{O}_k|_{\bar{q}_i} = \emptyset$ and $\mathcal{Z}^r(c_i^r) \cap \mathcal{O}_k|_{\bar{q}_i} = \emptyset$ for all $i \in \mathbb{N}_{[1,M]}$ and $k \in \mathbb{N}_{[1,N_o]}$. Next, we establish conditions for determining if the closed-loop system will enter the PI set for $\bar{q}_j$ while tracking $\bar{q}_i$.

[3]These vertices can be obtained by transforming the vertices of $\mathcal{O}_k$ into local coordinates and determining the additional vertices induced by $x \leq 0$.

A. Connectivity of Equilibria

Define the extended state $\xi|_{\bar{q}_i} := (q|_{\bar{q}_i}, v) = (q_i, v)$. We define connectivity of $\bar{q}_i$ to $\bar{q}_j$ as follows.

Definition 3. *$\bar{q}_i \in \Sigma$ is*

- *forward connected to $\bar{q}_j \in \Sigma$ if under* (3)-(4) *for $\bar{q}_i$ with $\xi(0)|_{\bar{q}_i} \in \mathcal{Q}^l_{\text{ext}}$, there exists $t^* > 0$ such that $\xi(t)|_{\bar{q}_j} \in \mathcal{Z}^l(c^l_j)$ for all $t > t^*$.*
- *backward connected to $\bar{q}_j \in \Sigma$ if under* (3)-(4) *for $\bar{q}_i$ with $\xi(0)|_{\bar{q}_i} \in \mathcal{Q}^r_{\text{ext}}$, there exists $t^* > 0$ such that $\xi(t)|_{\bar{q}_j} \in \mathcal{Z}^r(c^r_j)$ for all $t > t^*$.*
- *connected to $\bar{q}_j$ if it is forward or backward connected.*

Hence, Problem 1 can be solved by finding a sequence of equilibria, $\{\bar{q}_i\}_{i=0}^N$, such that $\bar{q}_i$ is connected to $\bar{q}_{i+1}$ for all $i \in \mathbb{N}_{[1,N-1]}$ and $\bar{q}_{\text{target}} = \bar{q}_N$. Thus, we now establish sufficient conditions for checking connectivity of $\bar{q}_i$ to $\bar{q}_j$. To this end, we express $\bar{q}_i$ in the coordinates of $\bar{q}_j$. That is, let $\bar{q}_i|_{\bar{q}_j} =: (\bar{x}_{ij}, \bar{y}_{ij}, \bar{\phi}_{ij})$, where

$$\begin{bmatrix} \bar{x}_{ij} \\ \bar{y}_{ij} \end{bmatrix} := R(-\bar{\phi}_j) \begin{bmatrix} \bar{x}_i - \bar{x}_j \\ \bar{y}_i - \bar{y}_j \end{bmatrix}, \quad \bar{\phi}_{ij} = \bar{\phi}_i \ominus \bar{\phi}_j. \tag{9}$$

The first requirement for connectivity of $\bar{q}_i$ to $\bar{q}_j$ is that for some $\lambda_c \in (0,1)$,

$$\begin{bmatrix} \bar{x}_{ij} & \bar{y}_{ij} \end{bmatrix} P_{11} \begin{bmatrix} \bar{x}_{ij} \\ \bar{y}_{ij} \end{bmatrix} \leq (1-\lambda_c)c, \tag{10}$$

where $c = c^l_j$ for forward connectivity and $c = c^r_j$ for backward connectivity. The second requirement for forward connectivity is

$$\bar{x}_{ij} \leq -\delta_x, \tag{11a}$$

for some $\delta_x > 0$, and for backward connectivity is

$$\bar{x}_{ij} \geq \delta_x. \tag{11b}$$

Finally, the third requirement for both forms of connectivity is that for some $\delta_\phi \in (0, \pi)$:

$$\bar{\phi}_{ij} \in [-\pi + \delta_\phi, \pi - \delta_\phi]. \tag{12}$$

Proposition 2. *The equilibrium $\bar{q}_i$ is forward (or backward) connected to $\bar{q}_j$ if there exists $\lambda_c \in (0,1)$, $\delta_x > 0$, and $\delta_\phi > 0$ such that* (10)*,* (11a) *(or* (11b)*), and* (12) *are satisfied.*

Proof. We prove the case of forward connectivity and omit the proof for backward connectivity as it follows by symmetric arguments. Let $q|_{\bar{q}_i} =: q_i =: (x_i, y_i, \phi_i)$. The following transformation expresses the vehicle's state in the coordinates of $\bar{q}_j$ (i.e., q_j) as a function of the coordinates of $\bar{q}_i$ (i.e., q_i):

$$\begin{aligned} \begin{bmatrix} x_j \\ y_j \end{bmatrix} &= R(\bar{\phi}_i \ominus \bar{\phi}_j) \begin{bmatrix} x_i \\ y_i \end{bmatrix} + R(-\bar{\phi}_j) \begin{bmatrix} \bar{x}_i - \bar{x}_j \\ \bar{x}_i - \bar{y}_j \end{bmatrix}, \\ \phi_j &= \phi_i \oplus (\bar{\phi}_i \ominus \bar{\phi}_j). \end{aligned} \tag{13}$$

Let $\xi|_{\bar{q}_i}(t) = \xi_i(t) = (x_i, y_i, \phi_i, v)(t)$ evolve under (3)-(4) for $\xi_i(0) \in \mathcal{Q}^l_{\text{ext}}$. We begin by showing that there exists a finite time such that $\xi|_{\bar{q}_j}(t) = \xi_j(t) \in \mathcal{Z}(c^l_j)$. Using the relationship in (13) and the requirement (10), it is straightforward that

$$\xi_j^T P \xi_j \leq (1-\lambda_c)c^l_j + O^{(1)}(\xi_i) + O^{(2)}(\xi_i),$$

where $O^{(1)}(\xi_i)$ and $O^{(2)}(\xi_i)$ represent terms that depend linearly and quadratically on ξ_i. Since $\xi_i(t) \to 0$, there exists $t_1 > 0$ such that $O^{(1)}(\xi_i(t)) + O^{(2)}(\xi_i(t)) < \lambda_c c^l_j$ for all $t > t_1$, which completes the first step of the proof.

Next, we show that there exist a finite time such that $\xi_j(t) \in \mathcal{Q}^l_{\text{ext}}$. Note that $q_i(t) \to 0$ implies that $q_j(t) \to \bar{q}_{ij}$. Hence, by (11a) and $x_j(t) \to \bar{x}_{ij}$, there exist $t_2 > 0$ such that $x_j(t) \leq -\delta_x/2 < 0$ for all $t > t_2$. Next, we prove that there exist $t_3 > 0$ such that $q_j(t) \notin \mathcal{Q}^*$ for all $t > t_3$. The first condition in Q^*, i.e., $(x_j = 0, \cos\phi_j \geq 0)$, is easily ruled out since $x_j(t) < 0$ for all $t > t_2$. The second condition, i.e., $(y_j = 0, \cos\phi_j = -1)$, can be ruled out by noting that $\phi_j \to \bar{\phi}_{ij}$ and that $\bar{\phi}_{ij}$ is bounded away from π by (12). Hence, there exists $t_3 \geq t_2$ such that $q_j(t) \in \mathcal{Q}^l$ for all $t > t_3$. Finally, since $v(t) \to 0$, there exist a $t_4 \geq t_3$ such that $\xi_j(t) \in \mathcal{Q}^l_{\text{ext}}$ for all $t > t_4$. Thus, there exist $t^* = \max\{t_1, t_4\}$ such that $\xi_j(t) \in \mathcal{Z}^l(c^l_j)$ for all $t > t^*$. □

We refer to the procedure of checking whether $\bar{q}_i$ and $\bar{q}_j$ satisfy (10), (11a) (or (11b)), and (12) for some $\lambda_c, \delta_x, \delta_\phi$ as checking if $\bar{q}_j$ and $\bar{q}_j$ are forward (or backward) connected.

Next, we show that forward connectivity is equivalent to backward connectivity with the opposite orientation.

Corollary 1. *Given two equilibria $\bar{q}_j, \bar{q}_j \in \Sigma$, suppose that $q_k, q_\ell \in \Sigma$ are equilibria with opposite orientations, i.e., $\bar{q}_k = (\bar{x}_k, \bar{y}_k, \bar{\phi}_k) = (\bar{x}_i, \bar{y}_i, \bar{\phi}_i \oplus \pi)$ and $\bar{q}_\ell = (\bar{x}_\ell, \bar{y}_\ell, \bar{\phi}_\ell) = (\bar{x}_j, \bar{y}_j, \bar{\phi}_j \oplus \pi)$. Then, $\bar{q}_i$ and $\bar{q}_j$ are forward connected if and only if $\bar{q}_k$ and $\bar{q}_\ell$ are backward connected.*

Proof. We want to show that (10) with $c = c^l_j$, (11a), and (12) are satisfied if and only if

$$\begin{bmatrix} \bar{x}_{k\ell} & \bar{y}_{k\ell} \end{bmatrix} P_{11} \begin{bmatrix} \bar{x}_{k\ell} \\ \bar{y}_{k\ell} \end{bmatrix} \leq (1-\lambda_c)c^r_\ell, \tag{14a}$$

$$\bar{\phi}_{k\ell} \in [-\pi + \delta_\phi, \pi - \delta_\phi], \text{ and } \bar{x}_{k\ell} \geq \delta_x, \tag{14b}$$

where $c^r_\ell = c^l_j$ since $\bar{q}_\ell = (\bar{x}_j, \bar{y}_j, \bar{\phi}_j \oplus \pi)$ (Proposition 1).

By (9) and the definition of $\bar{q}_k$ and $\bar{q}_\ell$, we have that

$$\begin{bmatrix} \bar{x}_{k\ell} \\ \bar{y}_{k\ell} \end{bmatrix} = R(-\bar{\phi}_j \oplus \pi) \begin{bmatrix} \bar{x}_i - \bar{x}_j \\ \bar{y}_i - \bar{y}_j \end{bmatrix} = - \begin{bmatrix} \bar{x}_{ij} \\ \bar{y}_{ij} \end{bmatrix},$$

and $\bar{\phi}_{k\ell} = (\bar{\phi}_i \oplus \pi) \ominus (\bar{\phi}_j \oplus \pi) = -\bar{\phi}_{ij}$. Hence, (14a) is equivalent to (10) with $c = c^l_j = c^r_\ell$. Moreover, we have that $\bar{x}_{ij} \leq -\delta_x \iff \bar{x}_{k\ell} \geq \delta_x$ and $\bar{\phi}_{ij} \in [-\pi + \delta_\phi, \pi - \delta_\phi] \iff \bar{\phi}_{k\ell} \in [-\pi + \delta_\phi, \pi - \delta_\phi]$. □

Hence, if Σ is defined so that $\forall \bar{q}_i \in \Sigma$, $\exists \bar{q}_k \in \Sigma$ such that $\bar{q}_k = (\bar{x}_i, \bar{y}_i, \bar{\phi}_i \oplus \pi)$, then a backward motion graph can be automatically constructed from the forward motion graph.

B. Graph Definition

This subsection describes the structure of the graph $\mathcal{G} = (\mathcal{I}, \mathcal{E}, \mathcal{W})$ discussed in Step 3 of Algorithm 1.

1) Vertices and Edges: The graph $\mathcal{G}$ is defined by three subgraphs. The first subgraph $\mathcal{G}_f = (\mathcal{I}_f, \mathcal{E}_f, \mathcal{W}_f)$ is the graph of forward connected equilibria. Its vertices are labels $i_+ \in \mathcal{I}_f$ that map to an index $i \in \mathbb{N}_{[1,M]}$ of an equilibrium $\bar{q}_i \in \Sigma$. An edge $(i_+, j_+) \in \mathcal{E}_f$ signifies that the equilibria $\bar{q}_i$ is forward connected to $\bar{q}_j$. The second subgraph $\mathcal{G}_b =$

$(\mathcal{I}_b, \mathcal{E}_b)$ is the graph of backward connected equilibria. Its vertices are labels $i_- \in \mathcal{I}_b$ that map to an index $i \in \mathbb{N}_{[1,M]}$ of an equilibrium $\bar{q}_i \in \Sigma$. An edge $(i_-, j_-) \in \mathcal{E}_b$ signifies that the equilibria $\bar{q}_i$ is backward connected to $\bar{q}_j$.

The sets $\mathcal{I}_f$ and $\mathcal{I}_b$ are such that for every $i \in \mathbb{N}_{[1,M]}$, there exist $i_+ \in \mathcal{I}_f$ and $i_- \in \mathcal{I}_b$ that map to i, and $\mathcal{I}_f \cap \mathcal{I}_b = \emptyset$. That is, we assign two separate graph vertices to each equilibrium in Σ to separate the forward and backward connections of each equilibrium.

The third subgraph $\mathcal{G}_{fb} = (\mathcal{I}, \mathcal{E}_{fb}, \mathcal{W}_{fb})$ is the connections of the vertices in $\mathcal{I}_f$ and $\mathcal{I}_b$ that map to the same equilibrium point. That is, for every $i \in \mathbb{N}_{[1,M]}$, we include the edges (i_+, i_-) and (i_-, i_+) in $\mathcal{E}_{fb}$. This allows for transitions between forward and backward motion.

In summary, $\mathcal{G} = (\mathcal{I}, \mathcal{E}, \mathcal{W})$ where $\mathcal{I} = \mathcal{I}_f \cup \mathcal{I}_b$ and $\mathcal{E} = \mathcal{E}_f \cup \mathcal{E}_b \cup \mathcal{E}_{fb}$, and $|\mathcal{I}| = 2M$.

2) Weighting: The weight of an edge $(i_+, j_+) \in \mathcal{E}_f$ that describes the forward connection of $\bar{q}_i$ to $\bar{q}_j$ is

$$w_{ij}^+ = w_c + w_\phi |\bar{\phi}_i \ominus \bar{\phi}_j| + w_\gamma |\bar{\phi}_i \ominus \gamma_{ji}|,$$

where $\gamma_{ji} = \arctan(\bar{y}_j - \bar{y}_i, \bar{x}_j - \bar{x}_i)$, and w_c, w_ϕ, w_γ are non-zero weights. In summary, the weighting rewards paths with few vertices, small orientation changes, and equilibria oriented toward the following equilibria.

For $(k_-, \ell_-) \in \mathcal{E}_b$, the weight is defined as $w_{k\ell}^- = \lambda_b w_{ij}^+$ for a fixed $\lambda_b \geq 1$, where the indices k, ℓ and i, j are such that $\bar{q}_k = (\bar{x}_i, \bar{y}_i, \bar{\phi}_i \oplus \pi)$ and $\bar{q}_\ell = (\bar{x}_j, \bar{y}_j, \bar{\phi}_j \oplus \pi)$. That is, we mirror the weighting for the corresponding forward connection with an optional penalization for the backward motion, as forward motion is often preferred, e.g., due to sensor placement.

The edges in $\mathcal{E}_{fb}$ are assigned a constant weight $w_{fb} > 0$. This ensures that transitioning between forward and backward motion is only used when *somewhat* necessary. For example, the magnitude of w_{fb} may influence whether the planner tends to execute semi-circle or three-point turn maneuvers when turning around.

C. Graph Construction

Finally, we summarize the exact procedure used for construct graph construction:

1) Parameter Selection: Specify K_p and K_d satisfying A1-A2 in Result 1. Choose $Q \in \mathbb{S}^4_{\succ 0}$ and determine $P \in \mathbb{S}^4_{\succ 0}$ by solving the Lyapunov equation. Specify the parameters λ_c, δ_x, and δ_ϕ and the weights $w_c, w_\phi, w_\gamma, \lambda_b$.

Remark 3. *The matrices K_p, K_d, and Q can be selected to shape the set $\mathcal{Z}(c)$ so that equilibria are more easily connected. Meanwhile, λ_c dictates a trade-off between the number of connected equilibria and how much the vehicle may need to slow down to traverse between equilibria.*

2) Invariant Set Scaling: Generate Σ so that there are no sampled equilibria in an obstacle. For each $\bar{q}_i \in \Sigma$, solve (7) for each obstacle $\mathcal{O}_k$. Choose $c_i^l > 0$ as the smallest set scaling that results from this process, and store the set scaling for backward motion to $\bar{q}_j = (\bar{x}_i, \bar{y}_i, \bar{\phi}_i \oplus \pi)$ as $c_j^r = c_i^l$ (Proposition 1).

3) Building the Graph: For each $i \in \mathbb{N}_{[1,M]}$, add the edges (i_+, i_-) and (i_-, i_+) to $\mathcal{E}_{fb}$ and the associated constant weights w_{fb} to $\mathcal{W}_{fb}$. For each $i \in \mathbb{N}_{[1,M]}$ and $j \in \mathbb{N}_{[1,M]}$, if $\bar{q}_i$ is forward connected to $\bar{q}_j$, then add the edge (i_+, j_+) to $\mathcal{E}_f$ and the weight w_{ij}^+ to $\mathcal{W}_f$. Then, find k and ℓ such that $\bar{q}_k = (\bar{x}_i, \bar{y}_i, \bar{\phi}_i \oplus \pi)$ and $\bar{q}_\ell = (\bar{x}_j, \bar{y}_j, \bar{\phi}_j \oplus \pi)$, and add the edge (k_-, ℓ_-) to $\mathcal{E}_b$ (Corollary 1) and the associated weight $w_{k\ell}^-$ to $\mathcal{W}_b$.

The complexity of Step 2 is $O(N_o M)$ since (7) must be solved for each equilibrium-obstacle pair. The complexity of Step 3 is $O(M^2)$. Hence, Step 3 is usually much more expensive since $M \gg N_o$ in most cases. Note that even though the number of obstacles N_o only directly impacts the complexity of Step 2, in practice the number of sampled equilibria M will need to increase in more cluttered environments to ensure the graph is sufficiently connected.

Finally, we remark that all of the computations required in Steps 2 and 3 are easily parallelizable.

D. Graph Search

Once the graph $\mathcal{G}$ is constructed, a standard shortest path algorithm (e.g., Dijkstra's) can be used to find a path between two vertices. However, for a given target equilibrium $\bar{q}_{\text{target}} = \bar{q}_j$ for some $j \in \mathbb{N}_{[1,M]}$, we must decide whether or not we want to find the shortest path to the vertex $j_+ \in \mathcal{I}_f$ or $j_- \in \mathcal{I}_b$. This represents approaching $\bar{q}_j$ with forward or backward motion respectively. Similarly, if the system is starting at rest at an equilibrium $\bar{q}_{\text{init}} = \bar{q}_i$ for some $i \in \mathbb{N}_{[1,M]}$, then we can choose the starting vertex as either $i_+ \in \mathcal{I}_f$ or $i_- \in \mathcal{I}_b$ and select the sign of the initial velocity $v(0)$ accordingly. Hence, for an equilibrium-to-equilibrium maneuver, there are four possible shortest path problems.

There will often be compelling reasons to fix the starting or ending vertex with particular choices (e.g., backing up into a parking space). Otherwise, one can simply select the shortest path of the four options. The graph searches can be performed on millisecond time scales, so performing up to four graph searches is not computationally prohibitive.

V. Online Planning and Control

In this section, we describe the online feedback control required to execute a given path $\mathcal{P}$.

A. Online Control for a Given Path

Let $\{e_k\}_{k=1}^{N_p}$ be the sequence of edges $e_k \in \mathcal{E}$ in the graph defining the path. Consider the case where $e_k = (i_+, j_+) \in \mathcal{E}_f$, i.e., forward motion between $\bar{q}_i$ and $\bar{q}_j$. While tracking $\bar{q}_i$, we periodically check if $\xi(t)|_{\bar{q}_j} \in \mathcal{Z}^l(c_j^l)$ is satisfied. Upon satisfaction, we begin tracking $\bar{q}_j$ and advance the path index to $k+1$. The case where $e_k = (i_-, j_-) \in \mathcal{I}_b$ is handled identically, except for checking membership in $\mathcal{Z}^r(c_j^r)$.

If $e_k = (i_+, i_-) \in \mathcal{E}_{fb}$ where $i_+ \in \mathcal{I}_f$ and $i_- \in \mathcal{I}_b$, we check if $\|q(t)|_{\bar{q}_i}\| \leq \epsilon$ for some small $\epsilon > 0$ and $v(t) < v_{\min}$ for some small $v_{\min} > 0$. Upon satisfaction, we reinitialize the velocity signal to $v(t) = -v_{\text{init}}$ for some small $v_{\text{init}} > 0$ and advance the path index to $k+1$. This induces a small discontinuity in the velocity signal to avoid the singularity in

the control law at $v = 0$. In practice, this transition between forward and backward motion would be handled smoothly at a lower-level control layer for which v is a setpoint. The case where $e_k = (i_-, i_+) \in \mathcal{E}_{fb}$ follows symmetrically by checking if $v(t) > -v_{\min}$ and resetting to $v(t) = v_{\text{init}}$.

B. Non-equilibrium Initial States and $\bar{q}_{\text{target}} \notin \Sigma$

A non-equilibrium initial state $\xi_g(0)$ can be included in the graph by adding a vertex τ to $\mathcal{I}_f$ if $v > 0$ or to $\mathcal{I}_b$ if $v < 0$. If $\tau \in \mathcal{I}_f$, then we add an edge (τ, i_+) for each $i \in \mathbb{N}_{[1,M]}$ such that $\xi(t)|_{\bar{q}_i} \in \mathcal{Z}^l(c_i^l)$. Similarly, if $\tau \in \mathcal{I}_b$, then we add an edge (τ, i_-) for each $i \in \mathbb{N}_{[1,M]}$ such that $\xi(t)|_{\bar{q}_i} \in \mathcal{Z}^r(c_i^r)$. Then, we search for the shortest path between τ and the target vertex. Note that edges ending at τ are unneeded.

A similar procedure can be performed to add a target $\bar{q}_{\text{target}} \notin \Sigma$ to the graph. However, we must first compute the associated PI set scaling(s) for $\bar{q}_{\text{target}}$ by solving (7) for each obstacle. Then, vertices $\nu_+ \in \mathcal{I}_f$ and $\nu_- \in \mathcal{I}_b$ are assigned to $\bar{q}_{\text{target}}$ and the necessary edges $(i_+, \nu_+) \in \mathcal{E}_f$ and $(i_-, \nu_-) \in \mathcal{E}_b$ can be added by checking connections of $\bar{q}_i$ to $\bar{q}_{\text{target}}$. Note that edges starting from ν_+ and ν_- are unneeded.

C. Updating the Graph for New Obstacles

The graph can be updated for a new obstacle by first solving (7) for each $\bar{q}_j \in \Sigma$ and the new obstacle. For each equilibria $\bar{q}_j$ whose forward motion set scaling c_j^l is reduced by the presence of the new obstacle, it is necessary to evaluate whether any edges $(i_+, j_+) \in \mathcal{E}_f$ are still valid. Moreover, for each edge $(i_+, j_+) \in \mathcal{E}_f$ that becomes invalid, a corresponding edge $(k_-, \ell_-) \in \mathcal{E}_b$ that becomes invalid can be determined using Corollary 1.

This procedure is less expensive than the initial offline graph construction since it is only necessary to re-evaluate the validity of existing edges whose target vertices correspond to equilibria whose set scalings were reduced by the new obstacle (e.g., if only one equilibrium is influenced by the new obstacle, then the number of operations required to update the graph is proportional to the number of existing edges with target vertices corresponding to this equilibria).

This strategy can also be used for real-time graph construction in scenarios where obstacles are unknown offline. That is, an obstacle-free graph can be constructed offline to establish the maximum number of edges, then updated online as obstacles are detected in the environment.

VI. Examples

Figure 1 displays the simulation environment. We consider an example where the vehicle must navigate from its starting pose $\bar{q}_\infty^0$ to the first target pose $\bar{q}_\infty^1$, then from $\bar{q}_\infty^1$ to $\bar{q}_\infty^2$, and so on until the vehicle exits at pose $\bar{q}_\infty^5$. The task is executed by planning five separate paths between each equilibria.

A set of expanded obstacles $\bar{\mathcal{O}}_k$ are obtained by expanding the true obstacles $\mathcal{O}_k$ by a box with length and width of 0.6 m to account for the vehicle size (compare Figures 1 and 2). These expanded obstacles are fed to the planner. The sampled equilibria Σ are generated with $\bar{x}_i \in [1.5, 18.5]$ and

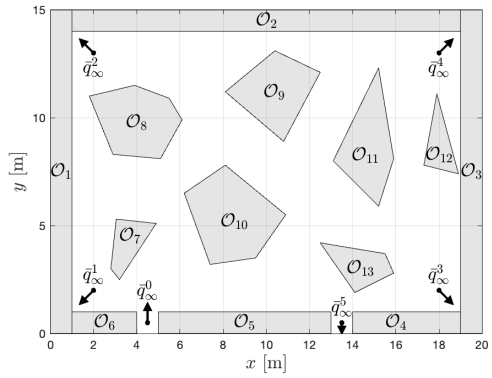


Fig. 1. The simulation environment and target equilibrium points.

$\bar{y}_i \in [0.5, 14.5]$ with uniform spacing of 0.5 m, while the sampled orientations $\bar{\phi}_i$ are

$$\arctan\left\{\begin{bmatrix}1\\0\end{bmatrix}, \begin{bmatrix}2\\1\end{bmatrix}, \begin{bmatrix}1\\1\end{bmatrix}, \begin{bmatrix}1\\2\end{bmatrix}, \begin{bmatrix}0\\1\end{bmatrix}, \begin{bmatrix}-1\\2\end{bmatrix}, \begin{bmatrix}-1\\1\end{bmatrix}, \begin{bmatrix}-2\\1\end{bmatrix}, \begin{bmatrix}-1\\0\end{bmatrix}, \begin{bmatrix}-2\\-1\end{bmatrix}, \begin{bmatrix}-1\\-1\end{bmatrix}, \begin{bmatrix}-1\\-2\end{bmatrix}, \begin{bmatrix}0\\-1\end{bmatrix}, \begin{bmatrix}1\\-2\end{bmatrix}, \begin{bmatrix}1\\-1\end{bmatrix}, \begin{bmatrix}2\\-1\end{bmatrix}\right\},$$

such that each orientation points directly between equilibria positions. This results in a total of $M = 8560$ obstacle-free equilibria, and hence $|\mathcal{I}| = 2M = 17{,}120$.

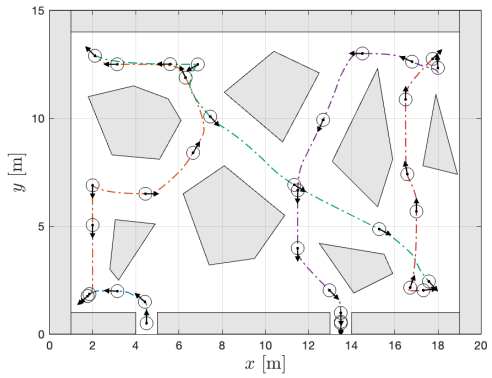
The parameters in Proposition 2 are $\delta_x = 0.5$, $\delta_\phi = 15\pi/180$, $\lambda_c = 0.3$. The weight parameters are $w_c = 1$, $w_\phi = 1$, $w_\gamma = 0.2$, $\lambda_w^- = 1.5$. The control gains are set to the same values used in [16, Section 6.4]: $k_{px} = 2$, $k_{dx} = 3$, $k_{py} = 12$, and $k_{dy} = 7$. The matrix $Q = \text{diag}(k_{px}, 10k_{py}, k_{dx}, 10k_{dy})$ is chosen as per Remark 3.

The graph $\mathcal{G}$ was generated using the procedures described in Section IV-C. The resulting graph contains $|\mathcal{E}| = 443{,}890$ edges. Computing the obstacle-free set scalings using the method described in Remark 2 required a total execution time[4] of 0.05 s, while constructing the graph required 1.4 s. We emphasize that our set scaling procedure requires $26\times$ less computation time (0.05 s vs. 1.3 s) than the method used in [7, Section IV] despite sampling nearly $4\times$ as many equilibria. This is due to the factors noted in Remark 2.

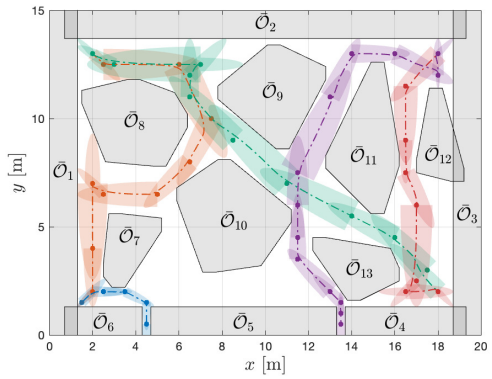
Figure 2 shows the planned paths and the resulting closed-loop trajectories. To demonstrate robustness and practical feasibility, the magnitude of the online control signals a and ω were saturated at 5 m/s^2 and 2 rad/s respectively. The system successfully performs the online control task in spite of this unmodeled saturation. The maneuver is executed in $\sim$95 s. The vehicle reaches a peak positive speed of 2.5 m/s in the path between $\bar{q}_\infty^4$ and $\bar{q}_\infty^5$, and a peak negative speed of -2.3 m/s in the path between $\bar{q}_\infty^1$ and $\bar{q}_\infty^2$.

As discussed in Section IV-D, multiple graph searches are performed for each path segment to automatically determine whether the vehicle should start and end with forward or

[4]The computational experiments were compiled and executed as `mex` code using MATLAB R2024b on a 2023 MacBook Pro (M3 Max) with 64 GB RAM. No special effort was conducted to optimize the code.



(a) Closed-loop trajectories and orientations



(b) Planned sequence of intermediate equilibria and invariant sets

Fig. 2. The planned path and closed-loop trajectories. The top figure shows the vehicle position, orientation, and dimensions at 3 second intervals. The bottom figure shows the planned sequence of invariant set projections compared to the expanded obstacles $\bar{\mathcal{O}}_k$. The path segments between each pair of target equilibria $\bar{q}^i_\infty$ and $\bar{q}^{i+1}_\infty$ are color coded.

backward motion. Specifically, two graph searches (corresponding to the cases with forward starting velocities) were computed to determine the path between $\bar{q}^0_\infty$ and $\bar{q}^1_\infty$, four graph searches were performed to determine each path between $\bar{q}^i_\infty$ and $\bar{q}^{i+1}_\infty$ for $i \in \mathbb{N}_{[1,3]}$, and two graph searches (corresponding to the cases with forward ending velocities) were computed to determine the path between $\bar{q}^4_\infty$ and $\bar{q}^5_\infty$. The cumulative execution time for the 16 graph searches using MATLAB's `shortestpath` function was 0.05 s.

The executed paths are quite efficient, intuitive, and make use of the available space quite well. For example, the sets between $\bar{\mathcal{O}}_9$, $\bar{\mathcal{O}}_{10}$ and $\bar{\mathcal{O}}_{11}$ span several meters in length. Hence, the spacial sample density of 0.5 m could be reduced in these regions. In contrast, the more intricate areas of the path (e.g., near $\bar{q}^1_\infty$ and $\bar{q}^5_\infty$) navigate through much smaller invariant sets. Hence, these regions require the smaller spacial sampling density.

VII. Conclusions

This paper developed an invariant-set motion-planner (ISMP) based on closed-loop feedback linearization for vehicles with unicycle-like dynamics. The ISMP is a graph-based planner that generates dynamically feasible collision-free trajectories by determining a sequence of obstacle-free positive invariant (PI) sets that connect the initial and target state. Efficient symmetry-exploiting methods for PI set scaling and graph construction were discussed. Future work will address robust motion planning under disturbances.

References

[1] A. Weiss, C. Petersen, M. Baldwin, R. S. Erwin, and I. Kolmanovsky, "Safe positively invariant sets for spacecraft obstacle avoidance," *Journal of Guidance, Control, and Dynamics*, vol. 38, no. 4, pp. 720–732, 2015.

[2] A. Weiss, C. Danielson, K. Berntorp, I. Kolmanovsky, and S. Di Cairano, "Motion planning with invariant set trees," in *2017 IEEE Conference on Control Technology and Applications (CCTA)*, pp. 1625–1630, 2017.

[3] K. Berntorp, A. Weiss, C. Danielson, I. V. Kolmanovsky, and S. Di Cairano, "Automated driving: Safe motion planning using positively invariant sets," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–6, 2017.

[4] K. Berntorp, R. Bai, K. F. Erliksson, C. Danielson, A. Weiss, and S. Di Cairano, "Positive invariant sets for safe integrated vehicle motion planning and control," *IEEE Transactions on Intelligent Vehicles*, vol. 5, no. 1, pp. 112–126, 2020.

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

[6] C. Danielson, K. Berntorp, A. Weiss, and S. Di Cairano, "Robust motion planning for uncertain systems with disturbances using the invariant-set motion planner," *IEEE Transactions on Automatic Control*, vol. 65, no. 10, pp. 4456–4463, 2020.

[7] C. Danielson, K. Berntorp, S. Di Cairano, and A. Weiss, "Motion-planning for unicycles using the invariant-set motion-planner," in *2020 American Control Conference (ACC)*, pp. 1235–1240, 2020.

[8] M. Greiff, H. Sinhmar, A. Weiss, K. Berntorp, and S. Di Cairano, "Invariant set planning for quadrotors: Design, analysis, experiments," *IEEE Transactions on Control Systems Technology*, vol. 33, no. 2, pp. 449–462, 2025.

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

[10] J. Leonard, J. How, S. Teller, M. Berger, et al., *A Perception-Driven Autonomous Urban Vehicle*, pp. 163–230. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009.

[11] M. Althoff, D. Althoff, D. Wollherr, and M. Buss, "Safety verification of autonomous vehicles for coordinated evasive maneuvers," in *2010 IEEE Intelligent Vehicles Symposium*, pp. 1078–1083, 2010.

[12] M. Althoff and J. M. Dolan, "Online verification of automated road vehicles using reachability analysis," *IEEE Transactions on Robotics*, vol. 30, no. 4, pp. 903–918, 2014.

[13] R. Tedrake, I. R. Manchester, M. Tobenkin, and J. W. Roberts, "LQR-trees: Feedback motion planning via sums-of-squares verification," *The International Journal of Robotics Research*, vol. 29, no. 8, pp. 1038–1052, 2010.

[14] M. Aicardi, G. Casalino, A. Bicchi, and A. Balestrino, "Closed loop steering of unicycle like vehicles via Lyapunov techniques," *IEEE Robotics Automation Magazine*, vol. 2, no. 1, pp. 27–35, 1995.

[15] A. De Luca, G. Oriolo, and M. Vendittelli, "Stabilization of the unicycle via dynamic feedback linearization," *IFAC Proceedings Volumes*, vol. 33, no. 27, pp. 687–692, 2000. 6th IFAC Symposium on Robot Control (SYROCO 2000), Vienna, Austria, 21-23 September 2000.

[16] A. De Luca, G. Oriolo, and M. Vendittelli, *Control of Wheeled Mobile Robots: An Experimental Overview*, pp. 181–226. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001.

[17] T.-C. Lee, K.-T. Song, C.-H. Lee, and C.-C. Teng, "Tracking control of unicycle-modeled mobile robots using a saturation feedback controller," *IEEE Transactions on Control Systems Technology*, vol. 9, no. 2, pp. 305–318, 2001.

[18] W. Dixon, D. M. Dawson, E. Zergeroglu, and A. Behal, *Nonlinear Control of Wheeled Mobile Robots*, vol. 262 of *Lecture Notes in Control and Information Sciences*. Springer, 2001.