Manifold Learning and Optimal Control for Obstacle Avoidance in Autonomous Driving*

Sanket Diwale[1,2], Tong Duy Son[2], Colin Jones[1‡]

April 29, 2019

Abstract

A novel manifold learning approach is presented to incorporate computationally efficient obstacle avoidance constraints in optimal control algorithms. The method presented provides a significant computational benefit by reducing the number of constraints required to avoid N obstacles from linear complexity $O(N)$ in traditional obstacle avoidance methods to a constant complexity $O(1)$. The application to autonomous driving problems is demonstrated by incorporation of the manifold constraints into optimal trajectory planning and tracking model predictive control algorithms in the presence of static and dynamic obstacles.

1 Introduction

Autonomous driving is an important application domain where obstacle avoidance is required in combination with optimal path planning and control. A wide range of methods including dynamic programming, numerical optimal control, MPC and randomized methods like RRT and A* have been used for motion planning and control of autonomous vehicles in presence of obstacles (e.g. [1–4]).

The approaches to obstacle avoidance can be broadly separated into two classes based on the obstacle/environment representation used. We will call these: (i) an obstacle centric approach and (ii) an environment centric approach. In an obstacle centric approach each obstacle in the environment is represented using a geometric description of its shape and constraints are imposed to ensure that the vehicle geometry does not collide with any obstacle geometry. In the environment centric approach a geometric description is directly extracted for a feasible region of movement from sensor data without reference to individual obstacle shapes. Constraints are then imposed such that the vehicle geometry remains inside the feasible region to avoid any collisions.

Examples of the obstacle centric approach can be found in works like [1–3, 5–14] while the environment centric approach can be found in [4, 15–19].

Polyhedral obstacle and vehicle geometries are used by [1, 5–7] with hyperplane separation constraints to impose obstacle avoidance in a nonlinear model predictive control (NMPC) scheme. [2] uses circular obstacles with dynamic programming while [3] uses an RRT* algorithm with polyhedral obstacles. [12, 13] define a potential field for spherical obstacles while [10, 11, 14] use a mixed integer approach for spatially varying constraints for polyhedral obstacles.

The obstacle centric approach imposes an $O(N)$ complexity in the number of constraints if N obstacles are present. In particular for algorithms planning over a horizon length H, $O(NH)$ constraints are needed. This dependence on N creates a problem for real time applications where N can be large and more importantly, can change dynamically, requiring an online update of the optimal control problem structure.

The environment centric approach seeks to circumvent this dependence on N by constructing directly a representation for the feasible region from sensor data. [15, 16] use a Support Vector Machine to learn a non-convex environment representation and perform RRT within the learned region. [4] uses a circular

**This work has been conducted within the ENABLE-S3 project that has received funding from the ECSEL Joint Undertaking under grant agreement No 692455. This joint undertaking receives support from the European Union's HORIZON 2020 research and innovation program and Austria, Denmark, Germany, Finland, Czech Republic, Italy, Spain, Portugal, Poland, Ireland, Belgium, France, Netherlands, United Kingdom, Slovakia, Norway.

†[1] Automatic Control Laboratory, EPFL, Switzerland

‡[2] Siemens Industry Software NV (SISW), Interleuvenlaan 68, 3001 Leuven, Belgium

region around the vehicle and LIDAR measurements to partition the circle into sectors not containing any obstacles. A multiphase NMPC scheme is then used to plan trajectories within this circle. [17–19] use a deep neural network to learn a mapping from features observed in video data of the road to the steering action applied. The features typically correspond to boundaries or markers for the feasible region (e.g. [20]). The approach however does not combine with optimal planning or control and may not always guarantee obstacle avoidance depending on the quality of training and network architecture used.

We present here a manifold learning algorithm to learn feasible environment representations, for an environment centric approach of obstacle avoidance in optimal control methods. The number of constraints introduced is independent of the number of obstacles ($O(1)$ complexity or $O(H)$ for horizon H), while introducing constraints of comparable computational complexity to the linear hyperplane constraints.

The chapter is structured as follows: Section 2 presents the manifold learning algorithm and its use for obstacle avoidance in a general optimal control method. Section 3 discusses an optimal trajectory planning and path following NMPC scheme for a car parking scenario incorporating the manifold constraints to avoid static and dynamic obstacles in a complex environment. Numerical studies are presented in Section 4. Section 5 concludes the paper with a few remarks on the method presented and future directions.

Notation

Throughout this chapter, let $\mathcal{H}$ be a Hilbert space of 2π-periodic functions mapping $[0, 2\pi)$ to $\mathbb{R}^2$, with orthonormal basis from a subset of $\cup_{k\in\mathbb{N},e_i\in\{(1,0),(0,1)\}}(\{e_i \cos kt\} \cup \{e_i \sin kt\})$. Let $\phi : \mathbb{R}^2 \to [0, 2\pi)$, defined as

$$\phi([x, y]) := \text{arctan2}(y, x)$$

give the angular coordinate of a point in $\mathbb{R}^2$. The angular coordinate for the point $(0, 0)$ is taken to be 0, i.e. $\phi([0, 0]) = 0$. Let $||\cdot||_{\mathcal{H}}$, $\langle\cdot\rangle_{\mathcal{H}}$ and $||\cdot||_{\mathbb{R}^2}$, $\langle\cdot\rangle_{\mathbb{R}^2}$ be the standard 2-norm and inner product on $\mathcal{H}$ and $\mathbb{R}^2$ respectively. Let $\times$ be the cross product in $\mathbb{R}^2$.

2 Manifold learning

We present here our main result for manifold learning, as a means for an environment centric approach to obstacle avoidance in optimal control.

Definition 1. *(Star-shaped Manifold)*
Let c be any point in $\mathbb{R}^2$. Let $f \in \mathcal{H}$ be a 2π periodic function. Let $f_r \in \mathcal{H}$ be defined as $f_r(t) = (r\cos(t), r\sin(t))$ for some fixed $r > 0$ and $\mathcal{M}_c := \{F(t) := (f(t) + f_r(t) + c) : t \in [0, 2\pi)\}$ be a closed curve of points in $\mathbb{R}^2$. Then, for all $t \in [0, 2\pi)$ if $\phi(F(t) - c) = t$ then we say $\mathcal{M}_c$ is a star shaped $\mathcal{S}^1$ isomorphic manifold centered at c.

Note that this definition for star shaped manifold is non-standard and refers to the idea that the interior of such a closed curve will be a star shaped set. For a star shaped manifold $\mathcal{M}_c$, let the interior be defined as $\text{int}(\mathcal{M}_c) := \{p \in \mathbb{R}^2 : \forall m \in [0, 1],\ mp + (1 - m)c \notin \mathcal{M}_c\}$, i.e. the set of points p in $\mathbb{R}^2$ such that a straight line connecting p to c has no intersection with the closed curve of the manifold $\mathcal{M}_c$.

Note that not all $f \in \mathcal{H}$ will represent a star shaped manifold and that the shape of the curve changes as we change the function f. Define

$$L_t f := F(t) = f(t) + f_r(t) + c \tag{1}$$

as the affine operator from $\mathcal{H} \to \mathbb{R}^2$ for each fixed t. Similarly define

$$T_t f = \partial_t f(t) + \partial_t f_r(t) \tag{2}$$

and

$$N_t f := \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} T_t f \tag{3}$$

giving a tangent and normal respectively to the manifold at t.

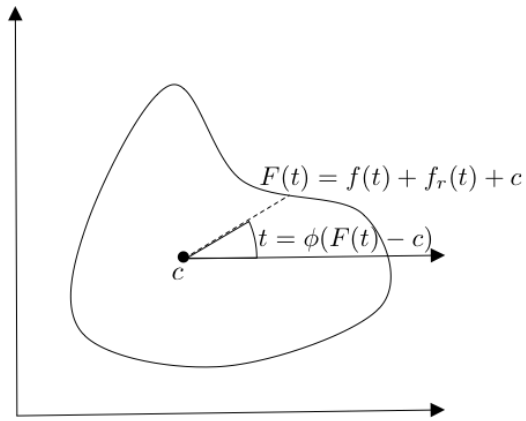


Figure 1: A star shaped manifold $\mathcal{M}_c$ centered at $c \in \mathbb{R}^2$. The manifold is parameterized by $t \in [0, 2\pi)$ and satisfies the constraint $t = \phi(F(t) - c)$ for all $t \in [0, 2\pi)$.

2.1 Learning the manifold

Given a point cloud $\mathcal{P}$ of data in $\mathbb{R}^2$, Theorem 1 below describes the means to learning a star shaped manifold $\mathcal{M}_c$ such that all obstacle points lie outside the area enclosed by $\mathcal{M}_c$, i.e. $\text{int}(\mathcal{M}_c) \cap \mathcal{P} = \emptyset$.

Theorem 1. *Let $\mathcal{P} := \{p_i : i = 1, \dots, M, p_i \in \mathbb{R}^2\}$ be a point cloud of data coordinates and for any $c \in \mathbb{R}^2 \backslash \mathcal{P}$ be some point not included in $\mathcal{P}$. Then a minimizer to the variational problem*

$$f_{opt} = \underset{f \in \mathcal{H}}{\text{argmin}} ||f||^2_{\mathcal{H}} \tag{4}$$

$$s.t.\ \forall i \in \{1, \dots, M\},\ t_i = \phi(p_i - c)$$

$$(L_{t_i} f - c) \times N_{t_i} f_r = 0 \tag{4a}$$

$$\langle p_i - L_{t_i} f, N_{t_i} f_r \rangle_{\mathbb{R}^2} \geq 0 \tag{4b}$$

defines a star shaped $\mathcal{S}^1$ isomorphic manifold centered at $c \in \mathbb{R}^2$,

$$\mathcal{M}_c := \{F_{opt}(t) := f_{opt}(t) + f_r(t) + c : t \in [0, 2\pi)\}$$

with $\text{int}(\mathcal{M}_c) \cap \mathcal{P} = \emptyset$.

The proof for the theorem relies on the following two lemmas.

Lemma 1. *Let $\mathcal{H}_* := \{f \in \mathcal{H} : \forall t \in [0, 2\pi), \phi(L_t f - c) = t\}$ be the subset of curves in $\mathcal{H}$ that lead to a star shaped manifold. Define*

$$\pi_f = \text{argmin}_{\hat{f} \in \mathcal{H}_*} ||f - \hat{f}||_{\mathcal{H}}$$

as the projection of f to $\mathcal{H}_*$. Then

(i) $\mathcal{H}_*$ is a closed convex set in $\mathcal{H}$ and

(ii) $\forall f \in \mathcal{H} \backslash \mathcal{H}_*$, $||\pi_f||_{\mathcal{H}} < ||f||_{\mathcal{H}}$

Proof. Firslty, note that any $f \in \mathcal{H}_*$ must be of the form $f(t) = a(t)(\cos(t), \sin(t))$, for some 2π-periodic function $a : [0, 2\pi) \to [0, \infty)$ in $L^2([0, 2\pi))$. $f(t)$ must take this form, because the angle $\phi(L_t f - c) = t$ is given for any $f \in \mathcal{H}_*$. Also the function $a(\cdot)$ must be in $L^2([0, 2\pi))$ to ensure that $||f||_{\mathcal{H}} = ||a(\cdot)(\cos(\cdot), \sin(\cdot))||_{\mathcal{H}} < \infty$. Thus for all $f_1, f_2 \in \mathcal{H}_*$, there exist functions $a_1 : [0, 2\pi) \to [0, \infty)$ and $a_2 : [0, 2\pi) \to [0, \infty)$ such that $f_1(t) = a_1(t)(\cos t, \sin(t))$ and $f_2(t) = a_2(t)(\cos t, \sin(t))$.

Then for all $\alpha, \beta \in [0, \infty)$, $\alpha f_1 + \beta f_2 = (a_1(t) + a_2(t))(\cos(t), \sin(t)) \in \mathcal{H}_*$. As a result for any $\alpha \in [0, 1]$, $\beta = (1 - \alpha)$ and $f_1, f_2 \in \mathcal{H}_*$, we have $\alpha f_1 + (1 - \alpha) f_2 \in \mathcal{H}_*$, implying $\mathcal{H}_*$ is a convex set.

Further for any converging sequence $f_n \in \mathcal{H}_*$, we have a corresponding converging sequence $a_n \in L^2([0, 2\pi))$. Since $L^2([0, 2\pi))$ is a closed Hilbert space (by the Riesz-Fischer theorem), a_n must converge to a point $a \in L^2([0, 2\pi))$ and thus f_n converges to a f in $\mathcal{H}_*$, implying $\mathcal{H}_*$ is closed.

Thus we have shown the first statement ($\mathcal{H}_*$ is a closed convex set).

Then by the Hilbert Projection Theorem [21, Theorem 1.2], $\pi_f \in \mathcal{H}_*$ and $(f - \pi_f) \in \mathcal{H}_*^{\perp}$. Thus $||\pi_f||^2_{\mathcal{H}} + ||f - \pi_f||^2_{\mathcal{H}} = ||f||^2_{\mathcal{H}} \implies$ (ii). □

Lemma 2. *Let $f \in \mathcal{H}$ be any feasible solution to* (4)*. Then $\pi_f = \text{argmin}_{\hat{f} \in \mathcal{H}_*} ||f - \hat{f}||$ is also feasible for* (4)*.*

Proof. Let $\mathcal{T} = \{t : t = \phi(p_i - c), p_i \in \mathcal{P}\}$ be all the t_is at which constraints in (4) are imposed. Note that normal to the circle f_r at any $t \in [0, 2\pi)$, is given by $N_t f_r = (r \cos t, r \sin t)$. Also, as argued in Lemma 1, $\pi_f \in \mathcal{H}_*$ must be of the form $\pi_f(t) = a(t)(\cos(t), \sin(t))$ for some 2π-periodic function $a : [0, 2\pi) \to [0, \infty)$ in $L^2([0, 2\pi))$. Thus π_f trivially satisfies (4a) for all $t_i \in \mathcal{T}$. Also since both $f(t_i)$ and $\pi_f(t_i)$ satisfy (4a), both are in the span of $N_{t_i} f_r$. We can then claim $f(t_i) = \pi_f(t_i)$ for all $t_i \in \mathcal{T}$ as follows.

Suppose $f(t_i) \neq \pi_f(t_i)$ for some $t_i \in \mathcal{T}$, then there must exist a $g \neq 0$ in $\mathcal{H}_*$ and a $g^{\perp} \in \mathcal{H}_*^{\perp}$ such that $f = \pi_f + g + g^{\perp}$. Also since for all $t_i \in \mathcal{T}$, $f(t_i), \pi_f(t_i), g(t_i)$ are all co-linear (satisfying (4a)), $g^{\perp}(t_i) = 0$ for all $t_i \in \mathcal{T}$. Then $\pi_f = \text{argmin}_{\hat{f} \in \mathcal{H}_*} ||f - \hat{f}||_{\mathcal{H}} = \text{argmin}_{\hat{f} \in \mathcal{H}_*} ||\pi_f + g + g^{\perp} - \hat{f}||_{\mathcal{H}} = \pi_f + g$. But this implies $g = 0$ and hence a contradicts the assumption $f(t_i) \neq \pi_f(t_i)$ for some $t_i \in \mathcal{T}$. Thus for all $t_i \in \mathcal{T}$, we must have $f(t_i) = \pi_f(t_i)$ and thus $L_{t_i} \pi_f = L_{t_i} f$ and π_f satisfies (4b) as well. □

The proof for Theorem 1 then follows,

Proof. [Theorem 1] Note that by Lemma 2 for any feasible solution $f \in \mathcal{H} \backslash \mathcal{H}_*$, there exists the projection $\pi_f \in \mathcal{H}_*$ as a feasible star shaped solution. Further by Lemma 1, $||\pi_f||_{\mathcal{H}} < ||f||_{\mathcal{H}}$. Thus the minimum norm solution $\mathcal{M}_c$ must be star shaped. To show $\text{int}(\mathcal{M}_c) \cap \mathcal{P} = \emptyset$, note that (4a) enforces $L_{t_i} f_{opt} - c$ to be in the span of the outward pointing normal $N_{t_i} f_r$, while (4b) enforces that the vector pointing from $L_{t_i} f_{opt}$ to p_i is also outward pointing. This ensures that $L_{t_i} f_{opt}$ lies inside the line segment joining c and p_i and since $\mathcal{M}_c$ is star shaped, $\text{int}(\mathcal{M}_c) \cap \mathcal{P} = \emptyset$. Finally, note that for all $c \in \mathbb{R}^2 \backslash \mathcal{P}$ there exists an open neighborhood of c, call it $\text{int}\,(\mathcal{M}_{feas})$, such that $\text{int}\,(\mathcal{M}_{feas}) \cap \mathcal{P} = \emptyset$ (by virtue of Hausdorff separability of $\mathbb{R}^2$). Thus for all $c \notin \mathcal{P}$, there exists a feasible solution to (4). □

Note that the minimization problem in (4) can be an infinite dimensional one and that the proofs did not rely on any particular definition for the inner product (thus the theorems hold for any inner product definition on $\mathcal{H}$). The infinite dimensional problem is reduced to a finite dimensional one by limiting $\mathcal{H}$ to a space generated by finitely many *sin* and *cosine* basis. Then $f \in \mathcal{H}$ takes the form $f(t) = \sum_{k=0}^{K} a_k \cos kt + b_k \sin kt$ with a finite K and $a_k, b_k \in \mathbb{R}^2$ become the decision variables for (4). Another approach to reducing (4) to a finite dimensional problem while not reducing $\mathcal{H}$ to a finite basis is to use a reproducing kernel hilbert space $\mathcal{H}$ with a periodic kernel. We avoid this approach in the present work as it would incur a $O(M)$ complexity (M being the size of the point cloud) in evaluating $L_t f_{opt}$ (instead of $O(K)$, K being the size of the basis) making obstacle checking more expensive in Theorem 2.

2.2 Using the manifold for planning and control

Theorem 2. *Let $\mathcal{M}_c$ be the optimal manifold given by Theorem 1 with a center $c \in \mathbb{R}^2$. Then a point $p \in \mathbb{R}^2$ is contained in its interior,* $\text{int}(\mathcal{M}_c)$*, if and only if* (5) *is satisfied, i.e.,*

$$p \in \text{int}(\mathcal{M}_c) \iff \langle p - L_v f_{opt}, N_v f_r \rangle_{\mathbb{R}^2} \leq 0 \quad \textit{for} \quad v = \phi(p - c) \tag{5}$$

Theorem 2 provides a simple inequality check for any point being contained in a star shaped manifold. Thus for any point $p \in \mathbb{R}^2$ in order to check for containment in $\text{int}(\mathcal{M}_c)$ a single inequality suffices. To include such a constraint for obstacle avoidance in an optimal control problem simply include (5) for each p that needs to be checked. Thus for a horizon H optimal control algorithm where the state for each of the H steps is enforced to be feasible the number of constraints included are of the order $O(H)$. Section 3 describes this process in more detail.

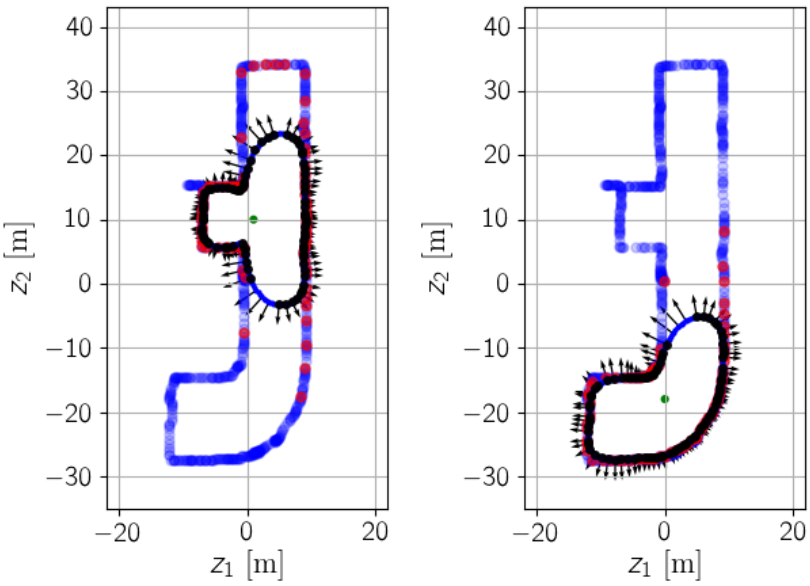


Figure 2: Learned Manifolds and Normal Fields. (Green point - center of manifold, red points - point cloud visible from manifold center, blue points - points invisible/out of sensor range from center, black - surface of the manifold, arrows - outward pointing normals on the manifold surface)

3 Optimal control and manifold constraints

Three different optimal planning and control algorithms are presented below that are used to accomplish different tasks for an autonomous car parking scenario, using the manifold constraints from Theorem 2. Section 3.1 describes a corridor planning algorithm over a graph of manifolds using a dynamic programming approach. The corridor plan is then used in Section 3.2 to solve a N-phase free end time, numerical optimal control problem for planning a trajectory for the vehicle to move within the free space described by the corridor, while accounting for the vehicle dynamics. The planned trajectory is used as a reference path and a path-following model predictive controller is described in Section 3.3 to account for dynamic obstacles and real-time control requirements.

3.1 Corridor planning

Given a set of point cloud information pertaining to locations of obstacles in an environment, a single star shaped manifold may not be enough to adequately represent the entire free space configuration in which the vehicle can move. Figure 3 shows an example of such an environment (with the point cloud shown in blue). A collection of manifolds (shown in faded black) are then learned with different centers in order to cover the entire free space of interest for the vehicle movement. The collection of manifolds is constructed to ensure that no manifold has a disjoint interior from the rest of the collection and as such there is a path connecting any two manifolds in the collection going through manifolds within the collection.

A undirected graph $\mathcal{G}$ of manifolds is then given by such a collection with each nodes in the graph representing a manifold from the collection and an edge between two nodes indicating that the interiors of the manifolds has non-empty intersection. The complete point cloud of static obstacle points from which the manifolds are learned is denoted as $\mathcal{O}_{static}$. Figure 3 shows an example of such an $\mathcal{O}_{static}$ and $\mathcal{G}$. A unit weight is assigned to each edge for simplicity (although other weighting schemes are also possible). Further all manifolds are learned with Theorem 1 so that $(\cup_{\mathcal{M}\in\mathcal{G}} \text{int}(\mathcal{M})) \cap \mathcal{O}_{static} = \emptyset$.

Then, given a desired starting and end point, p_{start} and p_{end} respectively, for the vehicle in $\mathbb{R}^2$, we can find the shortest sequence of manifolds in $\mathcal{G}$ connecting p_{start} to p_{end} using a dynamic programming

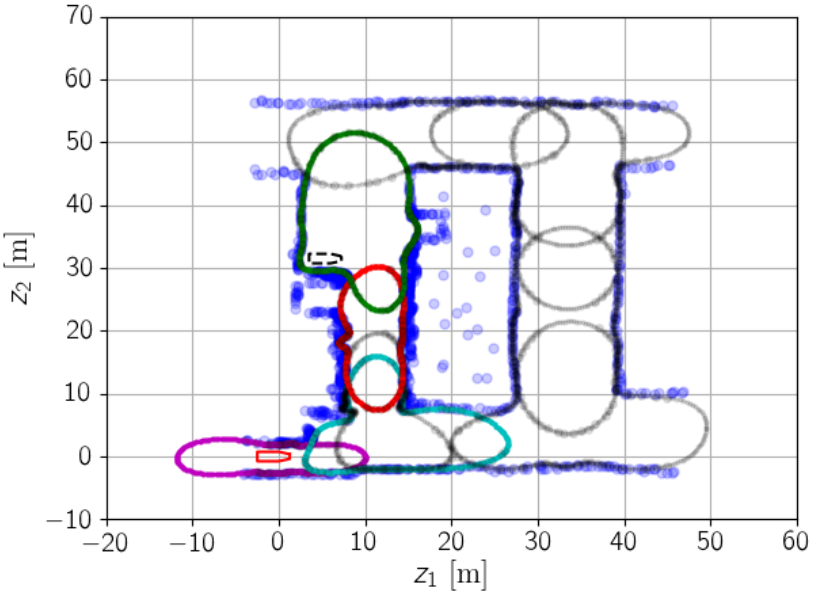


Figure 3: Corridor Planning over Manifolds. The shortest sequence of manifolds R_{seq} colored in magenta, cyan, red and green from start to goal. Unused manifolds from $\mathcal{G}$ are in faded black. The car to be controlled is plotted in red and the desired target state is shown with a dashed black profile.

algorithm. The process for constructing the shortest sequence is a standard dynamic programming algorithm is as shown in Algorithm 1. The containment check for any point p in a manifold in Algorithm 1 can be done using Theorem 2.

A path connect subset $\mathcal{C} \subseteq \mathbb{R}^2$ such that $\mathcal{C} \cap \mathcal{O}_{static}$ (i.e. not containing any static obstacle points) is called a corridor in the context of autonomous driving. Such a corridor is given by $\mathcal{M}_{c_1} \cup \mathcal{M}_{c_1} \cup \cdots \cup \mathcal{M}_{c_N}$, because each manifold satisfies $\mathcal{M}_{c_l} \cap \mathcal{O}_{static} = \emptyset$, by virtue of Theorem 1.

Algorithm 1 thus provides a fast corridor planning algorithm to find the shortest sequence of manifolds, $R_{seq} = \{\mathcal{M}_{c_1}, \ldots, \mathcal{M}_{c_N}\}$, that must be traversed to reach the end position. A multiphase optimal trajectory to traverse R_{seq} is then constructed in section 3.2, taking the vehicle dynamics into account.

3.2 Optimal trajectory planning

Let $R_{seq} = \{\mathcal{M}_{c_1}, \ldots, \mathcal{M}_{c_N}\}$ be the sequence of manifolds given by Algorithm 1 for start and end points, p_{start} and p_{end} respectively. Also let the center of $\mathcal{M}_{c_i}$ be the point $c_i \in \mathbb{R}^2$ for all $i = 1, \ldots, N$.

Consider for simplicity, a slip free Dubin's car model (6) to describe the non-holonomic vehicle dynamics, with the state $q = (z_1, z_2, \psi, v)$ comprising of (z_1, z_2) giving a coordinate position for the vehicle in $\mathbb{R}^2$, ψ giving a yaw orientation and v giving the car's forward speed. The controls used are a steering input δ and acceleration a. k_δ, k_{acc} are known constants corresponding to the steering and acceleration input gains.

$$\dot{q} = \left(v\cos\psi, v\sin\psi, k_\delta \cdot v \cdot \delta, k_{acc} \cdot a\right)^T \tag{6}$$

Assume, now, that the initial state of the vehicle dynamics is given as q_{start}, such that the corresponding position of the vehicle in $\mathbb{R}^2$ is p_{start} and the desired end state q_{end} for the vehicle is such that the corresponding position is p_{end}, as were used to plan the corridor sequence R_{seq}.

The algorithm presented next is agnostic of the exact form and details of the dynamic model used and we will simply denote the state of the vehicle dynamics by q, the inputs to the vehicle as u and the dynamics to be given by an ordinary differential equation,

$$\dot{q} = Q(q, u) \tag{7}$$

Algorithm 1 Dynamic programming over a graph for corridor planning

Input: graph of manifolds: $\mathcal{G}$, start point: p_{start}, end point: p_{end}

Algorithm:

First setup the costs for traversing the graph from any point to p_{end} as follows:

(i) Let all manifolds in $\mathcal{G}$ be assigned an infinite cost.

(ii) Find all manifolds in $\mathcal{G}$ containing p_{end}, call the set of such manifolds $\mathcal{A}_0$ and set their cost to 0. Set the iteration counter $l = 0$.

(iii) Let, the set of immediate neighbors to $\mathcal{A}_l$ with cost equal to ∞ be called $\mathcal{A}_{l+1}$. If $\mathcal{A}_l$ is an empty set, then terminate. Else, set the cost of all manifolds in $\mathcal{A}_l = l + 1$. Set the iteration counter l to $l + 1$.

(iv) Repeat (iii) till termination. (Note that the steps terminate since we have finitely many nodes in $\mathcal{G}$)

Next find a shortest sequence of manifolds going from p_{start} to p_{end}

(i) Find all nodes in $\mathcal{G}$ containing the point p_{start}, call the set $\mathcal{B}_0$. Set $\mathcal{M}_{c_1}$ as the manifold in $\mathcal{B}_0$ with the lowest assigned cost. Set the iteration counter to $l = 1$.

(ii) Find an immediate neighbor of $\mathcal{M}_{c_l}$ with minimum cost and set $\mathcal{M}_{c_{l+1}}$ to that neighbor. If the cost of $\mathcal{M}_{c_{l+1}}$ is 0, terminate. Else, set the iteration counter l to $l + 1$.

(iii) Repeat (ii) till termination.

Assuming the iteration terminates of the N^{th} step, we have the sequence of manifolds $\{\mathcal{M}_{c_1}, \ldots, \mathcal{M}_{c_N}\}$ giving the minimum cost for traversing the graph from p_{start} to p_{end}

Output: $\{\mathcal{M}_{c_1}, \ldots, \mathcal{M}_{c_N}\}$

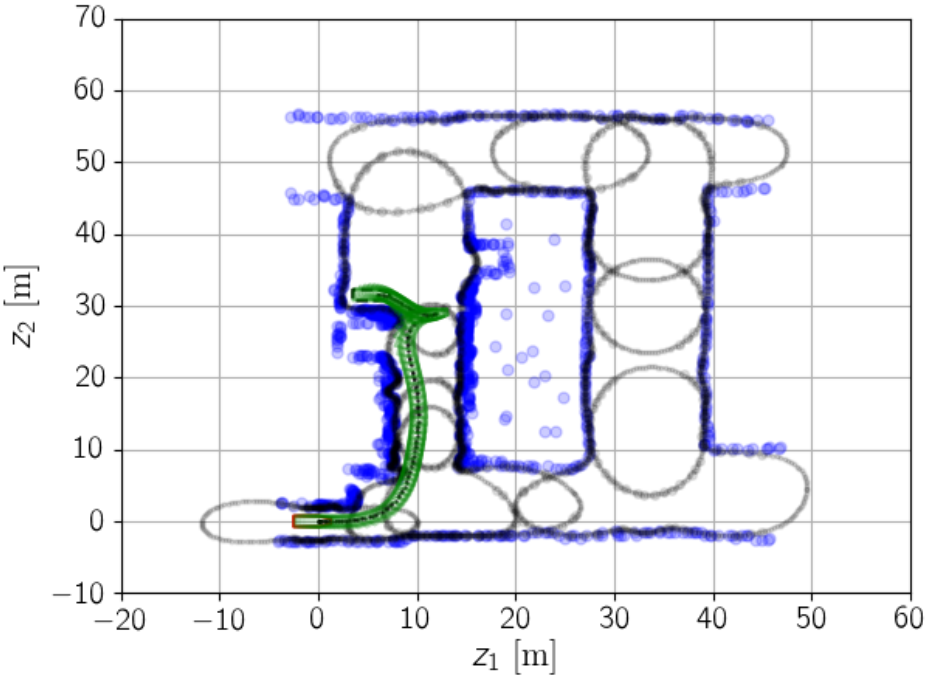


Figure 4: Optimal Trajectory Planning over Manifolds. The shortest time trajectory avoiding the corridor plan manifold constraints and adhering to the state dynamics and state-input constraints. The evolution of the car position and orientation is plotted in green over the period of the optimal trajectory.

Then, a N-phase optimal trajectory satisfying the non-holonomic vehicle dynamics and obstacle avoidance constraints can be generated as follows.

Let $\mathcal{M}_{c_1}$ to $\mathcal{M}_{c_N}$ denote the N manifolds in R_{seq}. Let $q(t) \in \mathbb{R}^n, u(t) \in \mathbb{R}^m$ be the state and input at time t for the vehicle and $\dot{q}(t) = Q(q(t), u(t))$ be the vehicle dynamics. q_{start} is given as the initial state of the vehicle at time $t = 0$ and q_{end} is the desired end state.

Let $p_j(t) = \Omega_j(q(t))$, $j = 1, \ldots, k$ for some finite k denote a collection points on the vehicle geometry for which to enforce obstacle avoidance, given by selection functions $\Omega_j : \mathbb{R}^n \to \mathbb{R}^2$ (see Definition 2 for an example of Ω_j). Also for purposes of brevity we will denote the fact that

$$p_j(t) = \Omega_j(q(t)) \in \text{int}(\mathcal{M}), \quad \forall j = 1, \ldots, k \iff q(t) \in \text{int}(\mathcal{M})$$

by the abuse of notation $q(t) \in \text{int}(\mathcal{M})$.

By construction, R_{seq} is such that $q_{start} \in \text{int}(\mathcal{M}_{c_1})$ and $q_{end} \in \text{int}\,(\mathcal{M}_{c_N})$. Let the state and input be bounded in box constraints $\mathcal{X}_{box}, \mathcal{U}_{box}$ respectively. Let $t_f \in [0, \infty)$ be a free end time for the trajectory and let $t_f^i \in [0, \infty)$ be the time for first exit from $\text{int}(\mathcal{M}_{c_i})$ for $i \in \{1, \ldots, N-1\}$. For notational convenience, let $t_f^0 = 0$ and $t_f^N = t_f$. A general N-phase optimal control problem is described in Algorithm 2 below, variants of which lead to the optimal trajectory generation and MPC path following algorithms to be presented later.

Algorithm 2 N-phase Optimal Control

Input: initial state: q_0, goal state: q_f, manifolds: $\{\mathcal{M}_{c_1}, \ldots, \mathcal{M}_{c_N}\}$ and smooth, strongly convex cost functionals: $\ell : \mathbb{R}^n \times \mathbb{R}^m \to [0, \infty)$, $G : \mathbb{R}^n \to [0, \infty)$

OCP: $q_{opt}, u_{opt}, t^1_{f\,opt}, \ldots, t^N_{f\,opt} :=$

$$\underset{\substack{\hat{q}(\cdot) \in L^2([0,\infty);\mathbb{R}^n) \\ \hat{u}(\cdot) \in L^2([0,\infty);\mathbb{R}^m) \\ t_f^1, \ldots, t_f^N \in [0,\infty)}}{\text{argmin}} G(\hat{q}(t_f)) + \int_0^{t_f} \ell(\hat{q}(s), \hat{u}(s))ds + \sum_{i=1}^{N} (t_f^i)^2 \tag{8a}$$

$$\text{s.t.} \quad \forall s \in (t_f^{i-1}, t_f^i], i \in \{1, \ldots, N\}, j \in \{1, \ldots, k\} \tag{8b}$$

$$\hat{q}(0) = q_0,\ \hat{q}(t_f) = q_f,\ t_f^0 = 0 \tag{8c}$$

$$\hat{q}(s) \in \mathcal{X}_{box}, \hat{u}(s) \in \mathcal{U}_{box},\ t_f^i - t_f^{i-1} \geq 0 \tag{8d}$$

$$\dot{\hat{q}}(s) = Q(\hat{q}(s), \hat{u}(s)) \tag{8e}$$

$$\hat{p}_j(s) := \Omega_j(\hat{q}(s)) \in \text{int}\,(\mathcal{M}_{c_i}) \tag{8f}$$

Output: q_{opt}, $u_{opt}, t^1_{f\,opt}, \ldots, t^N_{f\,opt}$

(8) describes a general free-end time $N-$phase optimal control problem where the variable t_f^i represents the end time for phase i, $i = 1, \ldots, N$. In each phase of the problem one manifold constraint is active, i.e. $\mathcal{M}_{c_i}$ is active for phase i.

(8c) enforces the initial and terminal boundary conditions for the vehicle state and sets the initial $t_f^0 = 0$ for notational convenience of (8d). (8d) enforces the input and state constraints to be satisfied and states that the switching times should be ordered such that the time at which the manifold is switched from $\mathcal{M}_{c_i}$ to $\mathcal{M}_{c_{i+1}}$ (given by t_f^i) is greater than the switching time t_f^{i-1} when the constraint for $\mathcal{M}_{c_i}$ was first made active. (8e) enforces that the solution q_{opt}, u_{opt} satisfy the differential equation for the dynamics considered. (8f) enforces that for all times $s \in (t_f^{i-1}, t_f^i]$ the selected points on the vehicle geometry are in the interior of the active manifold $\mathcal{M}_{c_i}$, thus avoiding all obstacles. (8f) is equivalent to the inequality constraint given by (5).

The optimal reference trajectory q_{ref} and control u_{ref} from q_{start} to q_{end}, given R_{seq} can then be found using Algorithm 3. Note that we are subscripting the optimal solutions as q_{ref} and u_{ref} as these solutions will be used as reference trajectories for a path following model predictive controller in Section 3.3.

Algorithm 3 Optimal Trajectory Generation

Input: $q_0 = q_{start}$, $q_f = q_{end}$, manifolds: R_{seq}, $\ell(\hat{q}(s), \hat{u}(s)) = \gamma||\hat{u}(s)||^2$ for some $\gamma > 0$ and $G(\hat{q}(t_f)) = 0$
Solve: OCP (8) and get q_{opt}, u_{opt} and $t^i_{f\,opt}$, $i = 1, \ldots, N$
Output: $t_f^{ref} = t^N_{f\,opt}$, $q_{ref} : [0, t_f^{ref}] \to \mathbb{R}^n := q_{opt}$

Theorem 3. *Let R_{seq}, $\mathcal{X}_{box}$ and $\mathcal{U}_{box}$ be such that the optimization (8) is feasible in Algorithm 3. Then the optimal trajectory, q_{ref} is such that for all $t \in [0, t_f^{ref}]$ and all $j \in \{1, \ldots, k\}$, $\Omega_j(q_{ref}(t)) \cap \mathcal{O}_{static} = \emptyset$ and $q_{ref}(t_f^{ref}) = q_{end}$.*

The proof for Theorem 3 follows directly from the enforced terminal constraint (8c) and manifold constraints (8f), which by Theorem 1 implies $\Omega_j(q_{ref}(t)) \cap \mathcal{O}_{static} = \emptyset$ in each phase. Thus Theorem 3 guarantees that the optimal reference trajectory is such that for all points on the trajectory q_{ref}, selected points of the vehicle geometry are contained in the interior of manifolds in R_{seq}, thus avoiding all known static obstacles in the map.

The next section, describes a real-time path following model predictive controller, using q_{ref} as a reference path in order to address the concerns of fast real time control (as generating an optimal trajectory by (8) for ever changing values of N, large values of N can be computationally expensive) and to address the issue that $\mathcal{O}_{static}$ being an offline data set of static obstacle information may not accurately describe the obstacles in the environment. To address the issue of apriori unknown and possibly moving obstacles, we introduce a new dynamic set $\mathcal{O}_{dyn}$ of point cloud data (acquired in real time with a LIDAR like sensor) in the next section and learn a dynamically changing manifold $\mathcal{M}_{c_1(t)}$ centered around a point on the vehicle.

3.3 Dynamic obstacles and model predictive control

Let $\mathcal{O}_{dyn}$ be a point cloud of dynamic obstacles not accounted for in $\mathcal{G}$ and let $\mathcal{O} = \mathcal{O}_{static} \cup \mathcal{O}_{dyn}$. While Theorem 3 provides an effective method to plan trajectories in presence of static obstacles; for dynamic obstacles we formulate an MPC path following scheme tracking the planned q_{ref} with a reference geometric path to follow. Treating q_{ref} as simply a parametrized geometric path and not a time-bound trajectory allows the controller to move to q_{ref} and follow along q_{ref} at a speed that is feasible for the real vehicle dynamics (as there may be a difference between the real vehicle dynamics and the model used for planning) and for constraints placed by the new dynamic obstacles.

Also since path-following under dynamic obstacles can lead to unforeseeable situations, we address here only the problem in a semi-cooperative setting. Under such a setting, we assume that path q_{ref} is not permanently made infeasible, i.e. we can move to any point to q_{ref} without being hindered by an obstacle permanently (a point may be unreachable for a finite amount of time, but the obstacles will move away in a finite time span, to make the point reachable). In such a setting, we also do not address adversarial obstacles, that are either actively trying to collide with the vehicle or accidentally in a state such that no control action by the MPC controller can avoid collision.

At any time t, let $q(t)$ be the vehicle state. Let $c_1(t) = s_0(q(t))$ be the position of a sensor on the vehicle in $\mathbb{R}^2$, given the state of the car, $q(t)$. Let $\mathcal{M}_{c_1(t)}$ be a dynamic manifold learned using Theorem 1 at each time t using new data from the sensor, allowing detection and avoidance of dynamic obstacles in $\mathcal{O}$. Let t_f^{ref} and $q_{ref} : [0, t_f^{ref}] \to \mathbb{R}^n$ be the reference end time and reference geometric curve mapping into the vehicle's state space as given by Algorithm 3.

Let $J : R_{seq} \to \mathbb{R}$ be a map from a manifold in R_{seq} to its cost to go, assigned in the dynamic program from Algorithm 1 to reach the end of the sequence. Then choose a manifold,

$$\mathcal{M}_{c_2(t)} = \text{argmin}_{\mathcal{M} \in R_{seq}} J(\mathcal{M}) \text{ s.t. } c_1(t) \in \text{int}(\mathcal{M})$$

i.e. $\mathcal{M}_{c_2(t)}$ is the manifold in R_{seq} with the minimum cost to go such that the current sensor position is contained in its interior. Recall that $\mathcal{M}_{c_1(t)}$ is the dynamically learned manifold, centered around the sensor position and thus $\mathcal{M}_{c_2(t)} \cap \mathcal{M}_{c_1(t)} \neq \emptyset$. It is assumed that the initial state of the vehicle in such that $\mathcal{M}_{c_2(t_0)} \cap \mathcal{M}_{c_1(t_0)} \neq \emptyset$ and the recursive feasibility of the MPC shown later will ensure that this remains true for all $t \geq t_0$.

Then for the path following MPC problem considered over a finite horizon T, a heuristic reference terminal state for the horizon $[t, t+T]$ is then chosen as follows.

Let $\mathcal{N}(\mathcal{M}_{c_2(t)})$ be the set of immediate neighbors of $\mathcal{M}_{c_2(t)}$ in R_{seq} including $\mathcal{M}_{c_2(t)}$ itself. Let $t_w(w,t) > 0$ be a positive offset parameter at time and path parameter (t,w), given by

$$t_w(w,t) = \text{argmax}_{\hat{t}_w \in [0, t_f^{ref}]} \hat{t}_w \text{ s.t. } q_{ref}(w+\hat{t}_w) \in \text{int}(M) \text{ and } M \in \mathcal{N}(\mathcal{M}_{c_2(t)})$$

and let $w_f(w,t) := \min(w + t_w(w,t), t_f^{ref})$ be a forward shift for any $w \in [0, t_f^{ref}]$. Then choose

$$w^*(t) = \text{argmin}_{w \in [0, t_f^{ref}]} ||q_{ref}(w) - q(t)|| \text{ s.t. } q_{ref}(w_f(w,t)) \in \cup_{\mathcal{M} \in \mathcal{N}(\mathcal{M}_{c_2(t)})} \text{int}(\mathcal{M})$$

This ensures $q_{ref}(w^*(t))$ is the closest point on q_{ref} to the current vehicle state $q(t)$ such that a future point $q_{ref}(w_f(w^*(t),t))$ lies within the neighborhood $\mathcal{N}(\mathcal{M}_{c_2(t)})$ and that $q_{ref}(w_f(w^*(t),t))$ is the closest possible point to the end goal that can be selected within the neighborhood $\mathcal{N}(\mathcal{M}_{c_2(t)})$. Recall that for any state $q \in \mathbb{R}^n$, we mean by $q \in \text{int}(\mathcal{M})$, that the corresponding selection of vehicle points $p_i \in \mathbb{R}^2$ is in $\text{int}(\mathcal{M})$. Note that this minimization is always feasible, since $\mathcal{M}_{c_2(t)}$ belongs to $\mathcal{N}(\mathcal{M}_{c_2(t)})$ and it is assumed the current vehicle state is in $\mathcal{M}_{c_2(t)}$ and thus one can always trivially choose $q_{ref}(w^*(t))$ and $q_{ref}(w_f(w^*(t),t))$ to be points in $\mathcal{M}_{c_2(t)}$ (since $\mathcal{M}_{c_2(t)}$ is a manifold chosen at time t from R_{seq} and q_{ref} passes through every manifold in R_{seq} by design).

Now let the heuristic end goal for the MPC over the horizon $[t, t+T]$ be given as

$$q_{end}^{ref}(t) = q_{ref}(w_f(w^*(t),t))$$

and let

$$\mathcal{M}_{c_3(t)} = \text{argmin}_{\mathcal{M} \in \mathcal{N}(\mathcal{M}_{c_2(t)})} J(\mathcal{M}) \text{ s.t. } q_{end}^{ref}(t) \in \text{int}(\mathcal{M})$$

be the manifold in the neighborhood $\mathcal{N}(\mathcal{M}_{c_2(t)})$ with minimum cost to go, containing the end goal $q_{end}^{ref}(t)$.

Thus we have three manifolds at each time t; $\mathcal{M}_{c_1(t)}$ accounting for new or dynamic obstacles in $\mathcal{O}$, and $\mathcal{M}_{c_2(t)}, \mathcal{M}_{c_3(t)} \in R_{seq}$, accounting for only static obstacles in $\mathcal{O}_{static}$ for manifolds leading from $q_{ref}(w^*(t))$ to $q_{end}^{ref}(t)$. The following optimal control problem can then be solved at each time t to get a path following NMPC controller.

Algorithm 4 Path Following MPC

Input: $N = 3$, $q_0 = q(t)$, $q_f = q_{end}^{ref}(t)$, manifolds: $\{\mathcal{M}_{c_1(t)}, \mathcal{M}_{c_2(t)}, \mathcal{M}_{c_3(t)}\}$, a safety time margin: $t_{safe} > 0$ and a maximum blocking time: $T_{blocking}$. The cost functionals:

$$\ell(\hat{q}(s), \hat{u}(s)) = \gamma_1 ||\hat{q}(s) - q_{ref}(\min\{w^*(t)+s, w_f(w^*(t),t)\})||^2 + \gamma_2 ||\hat{u}(s)||^2$$

$$G(\hat{q}(t_f)) = \gamma_3 ||\hat{q}(t_f) - q_{end}^{ref}(t)||^2$$

for some constants $\gamma_1, \gamma_2, \gamma_3 > 0$

Solve: OCP (8) subject to an additional constraint $T_{blocking} \geq t_f^1 \geq t_{safe}$ and get the optimal solution u_{opt}

Output: Applied control action: $u(t) = u_{opt}(0)$

Algorithm 4 thus proposes a moving horizon version of the free end time, $N-$phase optimal control problem in (8) with $N = 3$, tracking a geometric curve q_{ref}, thus giving a path following MPC controller. The MPC controller computes the control signal to follow the geometric reference path as closely as possible starting at $q(t)$ while avoiding the manifold constraints in order to reach a end state $q_{end}^{ref}(t)$ such that $q_{opt}(t_{f\ opt}^N) = q_{end}^{ref}(t)$ (by the imposed terminal constraint in (8)). The computed control signal at time t, $u(t)$ is applied to the system, to reach a new state and the optimization problem for Algorithm 4 is resolved again at the new state. Section 4 gives more details on the discrete time implementation of such a scheme. With the initial state in (8), set as the current state $q(t)$ and the end goal q_f in (8) set to the heuristically selected end goal $q_{end}^{ref}(t)$, note that when the true state q_{end} is in $\mathcal{N}(\mathcal{M}_{c_2(t)})$, the heuristic end goal as given above will always be $q_{end}^{ref}(t) = q_{end}$, as it is the closest point in the neighborhood to the end goal. Thus the

heuristic end goal over the horizon moves forward towards q_{end} as soon as such a movement is permitted by the obstacle environment, enabling the path following MPC to progress towards the end goal.

The manifolds $\mathcal{M}_{c_1(t)}$ is enforced to ensure that the dynamic obstacles are avoided for all time $[t, t+t_f^1]$. Manifolds $\mathcal{M}_{c_2(t)}$ and $\mathcal{M}_{c_3(t)}$ are used to plan future motion towards the goal once a blocking obstacle has moved away. The switching time $t_f^1 > t_{safe}$ ensures that a safety time margin is permitted for the vehicle to come to a halt or reverse its motion to avoid a moving obstacle, during the next iteration of the MPC algorithm. The amount of time the vehicle has to spend within $\mathcal{M}_{c_1(t)}$ before it can proceed with motion through the originally planned manifolds $\mathcal{M}_{c_2(t)}$ and $\mathcal{M}_{c_3(t)}$ is given by t_f^1. If a obstacle is blocking the path of motion for the vehicle then the problem in (8) is infeasible as we require for a $t^1_{f\,opt} \leq t^N_{f\,opt} < \infty$, $q_{opt}(t^N_{f\,opt}) = q^{ref}_{end}(t)$. Thus, an assumption is made on the obstacles, such that the obstacle will move away in a maximum time $T_{blocking}$ and thus we have $t_1^f = T_{blocking}$ when the motion is blocked and we plan the motion through $\mathcal{M}_{c_2(t)}$ and $\mathcal{M}_{c_3(t)}$ for the time $[t + T_{blocking}, infty)$. Note however that if the obstacle does not move away in time $T_{blocking}$, the re-solving of the MPC at the next time iteration again sets $t_1^f = T_{blocking}$ and thus the MPC can be permanently blocked from making progress by an obstacle and also the controller will not collide with such an obstacle for any time $[0, \infty)$, since there is a safety margin of t_{safe} seconds left from the previous iteration in which the vehicle can be brought to a halt.

The following assumptions on the obstacle environment are made to give formal guarantees on the convergence and recursive feasibility properties for the MPC scheme.

Assumption 1.

(i) The dynamic obstacles follow a semi-cooperative policy for their motion such that at any time $t \in [0, \infty]$, the obstacle will remain outside $\mathcal{M}_{c_1(t)}$ for the time interval $[t, t + t_{safe}]$ seconds. Note that this is not exploited by the MPC to behave in a adversarial manner and push obstacles around, since t_{safe} is only a lower bound for t_f^1. This assumption simply means that there is a non-zero safety time margin t_{safe} for the MPC to take a control action such that a collision can be avoided, given the current state of the vehicle.

(ii) Obstacles do not permanently make $q^{ref}_{end}(t)$ unreachable in R_{seq} (i.e. blocking obstacles will eventually move away). (This time may be different from $T_{blocking}$, the assumption is just required to ensure that we do not have infinite iterations of the MPC with $t_f^1 = T_{blocking}$ and thus the MPC is prevented from making any progress towards the goal)

(iii) The vehicle dynamics and input constraints are such that the vehicle has a maximum velocity and acceleration/braking such that in t_{safe} seconds it can go from the maximum velocity to zero velocity in $t_{safe}/2$ seconds and the reference trajectory can be tracked with zero position and orientation error for any velocity profile within the limits set by the state and input constraints, given a zero error at the initial state.

Given such assumptions, the following theorem can be established for an MPC scheme for following q_{ref} as a reference path in the presence of dynamic obstacles.

Theorem 4. *Given assumption 1-(i) and (iii), the closed loop solution $q(t)$ obtained by applying a control signal $u(t) = u_{opt}(0)$ using Algorithm 4 is such that for all $t \in [0, \infty)$ and all $j \in \{1, \ldots, k\}$, $\Omega_j(q(t)) \cap \mathcal{O} = \emptyset$, i.e. selected points on the vehicle geometry avoid all obstacles (dynamic and static) from $\mathcal{O} = \mathcal{O}_{dyn} \cup \mathcal{O}_{static}$.*

Proof. At some time t if (8) is feasible then $t^1_{f\,opt}$ is strictly greater than t_{safe} (by the imposed constraint). This implies there is a strictly positive time $t^1_{f\,opt} > t_{safe}$ before any $\Omega_j(q(t))$ exits $\text{int}(\mathcal{M}_{c_1(t)})$. Further by assumption 1-(i), all obstacles are guaranteed to remain outside $\text{int}(\mathcal{M}_{c_1(t)})$ for $[t, t + t_{safe}]$. Thus for all $t' \in [t, t + t_{safe}]$, $\Omega_j(q(t)) \cap \mathcal{O} = \emptyset$. Given the optimal solution at time t, going from $q(t)$ to $q^{end}_{ref}(t)$, for any time $t' \in [t, t + t_{safe}/2]$, Algorithm 4 can be re-solved for which a feasible solution from $q(t')$ to $q^{ref}_{end}(t')$ can be obtained as a motion from $q(t')$ to $q^{end}_{ref}(t)$ (albeit with a different time profile) followed by motion along q_{ref} from $q^{end}_{ref}(t)$ to $q^{end}_{ref}(t')$. (Such a solution exists by virtue of the assumption 1-(iii), which state that a

given state trajectory is can be tracked with zero error in position and yaw for any velocity profile). Thus (8) remains feasible for all $t' \in [t, t + t_{safe}/2]$. By recursively applying this argument for any $t \in [0, \infty)$, (8) remains feasible for $[t, t + t_{safe}/2]$ for each $t \in [0, \infty)$. Thus (8) remains feasible for all $t \in [0, \infty)$ if (8) is feasible at $t = 0$ and $\Omega_j(q(t)) \cap \mathcal{O} = \emptyset$. □

Theorem 5. *Given assumption 1-(ii), there exists a finite time t_{end} such that $q(t_{end}) = q_{end}$*

Proof. By assumption 1-(ii), for any time t, $q^{end}_{ref}(t)$ is reachable in some finite time, i.e. there exists a finite time $t' \in [0, \infty)$ such that $q(t') = q^{end}_{ref}(t)$ (note that here by reachable we mean the actual state q reaching the goal $q^{end}_{ref}(t)$ and not just the MPC prediction q_{opt}). Thus there exists a finite time $t' > t$ such that $w^*(t') > w^*(t)$. Thus $q^{ref}_{end}(t')$ is closer to q_{end} along q_{ref} than $q^{ref}_{end}(t)$. Since $w^*(t')$ is upper bound by t^{ref}_f, there must exist a finite t' such that $w^*(t') = t^{ref}_f$, i.e., $q^{ref}_{end}(t') = q_{end}$. Then a finite time $t_{end} > t'$ exists such that $q(t_{end}) = q_{end}$. □

4 Numerical results

Figure 2 shows the result of learning individual manifolds around different centers using Theorem 1. We use $r = 30$ m in f_r and a $\mathcal{H}$ space generated by a finite basis of sine and cosine with $K = 10$. In order to avoid the dependence on a dynamically changing and large M in Theorem 1, we preprocess the sensor data to return a single closest point in a sector of resolution θ_{res} by dividing the $[0, 2\pi]$ interval into $N_{part} = 90$ intervals. With N_{part} large enough we are assured that the sensor data accurately enough represents the obstacles. Thus M in Theorem 1 is fixed to be N_{part} for fast online optimization. The preprocessed visible points are shown in red in Figure 2.

Figure 3 shows R_{seq} for a parking scenario with $q_{start} = (0, 0, 0, 0)$ and $q_{end} = (6, 31.5, 0, 0)$. Using this R_{seq} and Theorem 3, Figure 4 shows the optimal trajectory plan q_{ref} from q_{start} to q_{end}. Figure 5 shows the closed loop behavior of the MPC scheme in presence of a dynamic obstacle. A second car (displayed as a green polytope) is added to the environment (not accounted for in $\mathcal{O}_{static}$) and drives out in the opposite direction of the controlled car (displayed as a red polytope). The planned motion at time t ($q_{opt}(\cdot)$) is shown in cyan and the goal state at time t ($q^{ref}_{end}(t)$) is shown as a pink dashed polytope. The final goal state q_{end} is shown as a black dashed polytope. When the new obstacle is encountered, as shown in Figure 5b, the path for the vehicle to move forward is blocked. In order to avoid the obstacle (moving in the opposite direction), the car reverses its motion and moves in reverse from Figure 5b to 5c (the front edge of the car can be seen moving from $z_2 = 20$ m in 5b to $z_2 = 17$ m in 5c). Eventually space is freed by the moving obstacle (Figure 5d) and the car drives forward again to eventually reach the parking state q_{end}.

The slip free Dubin's car model from (6) is used for the non-holonomic vehicle with the state $q = (z_1, z_2, \phi, v)$ comprising of the z_1, z_2 coordinate position in the ground plane, yaw orientation ψ and car's forward speed v. The controls used are a steering input δ and acceleration a.

Definition 2. *(Vehicle Geometry)*
For describing the vehicle geometry we use an elongated hexagon for the car shape projected on the $z_1 - z_2$ ground plane. Nine vertices are placed on the hexagon corners and side and backward face bisectors. The sensor for the car is placed at its center. The corresponding selection functions Ω_j ($j = 0, \ldots, 9$) are defined as $\Omega_j(q(t)) = (p^j_1 \cos\phi - p^j_2 \sin\phi + z_1(t), p^j_1 \sin\phi + p^j_2 \sin\phi + z_2(t))^T$ if (p^j_1, p^j_2) are coordinates of the point when the car state is $(0, 0, 0, 0)^T$.

For solving the free end time optimal control problem in (8), we use a time scaling input as a decision variable along with time scaled vehicle dynamics. The continuous time problem is converted to discrete time using a multiple shooting approach with RK4 integration of step-size: 0.1.

The control and state bounds imposed were $\mathcal{U}_{box} := \{(-1, -1)^T \leq u \leq (1, 1)^T\}$, $\mathcal{X}_{box} := \{(-\infty, -\infty, -\infty, -1) \leq x \leq (+\infty, +\infty, +\infty, 4)\}$ and $t_{safe} = 0.05$ seconds with $k_\delta = 0.4$, $k_{acc} = 5$.

On an Intel Core i7, 2.8 GHz processor using an interior point solver (ipopt) the average solve times for the algorithms were as follows: Manifold Learning: 200 ms, free end 4−phase time optimal trajectory generation: 34.9 sec and the free end time 3−phase path following MPC: 754 ms. Note also that the longer solve times for the MPC and optimal trajectory generation are to be expected as we are solving a

multiphase, free end time optimal control problem, which is typically computationally expensive compared to a trajectory tracking like approach. Faster implementation schemes thus need to be explored to make the MPC controller compatible for real time implementation.

5 Conclusion

A novel manifold learning approach was presented to learn representations of complex and dynamic obstacle environments and to provide computationally tractable constraints for optimal control algorithms. The use of the manifold constraints for obstacle detection and avoidance was demonstrated with three variants of optimal control problems; a dynamic programming approach for corridor planning, an optimal trajectory generation problem and a nonlinear MPC problem for path following. The three variants were deployed to drive a vehicle in a car parking scenario in presence of static and dynamic obstacles. Recursive feasibility of the MPC under semi-cooperative obstacle movements was shown. MPC schemes taking into account obstacle speed and movement plan or adversarial obstacles remains a subject for future investigation and was not covered in this work.

References

[1] A. Liniger, A. Domahidi, and M. Morari, "Optimization-Based Autonomous Racing of 1:43 Scale RC Cars," *Optimal Control Applications and Methods*, vol. 36, no. 5, p. 628 – 647, Jul. 2014.

[2] A. T. Rashid, A. A. Ali, M. Frasca, and L. Fortuna, "Path planning with obstacle avoidance based on visibility binary tree algorithm," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1440 – 1449, 2013.

[3] D. Connell and H. M. La, "Dynamic path planning and replanning for mobile robots using rrt," *arXiv preprint arXiv:1704.04585*, 2017.

[4] J. Liu, P. Jayakumar, J. L. Stein, and T. Ersal, "A nonlinear model predictive control algorithm for obstacle avoidance in autonomous ground vehicles within unknown environments," Army Tank Automotive Research Development and Engineering Center Warren MI, Tech. Rep., 2015.

[5] M. G. Plessen, D. Bernardini, H. Esen, and A. Bemporad, "Spatial-based predictive control and geometric corridor planning for adaptive cruise control coupled with obstacle avoidance," *IEEE Trans. on Control Systems Technology*, vol. 26, no. 1, pp. 38–50, Jan 2018.

[6] J. V. Frasch, A. Gray, M. Zanon, H. J. Ferreau, S. Sager, F. Borrelli, and M. Diehl, "An auto-generated nonlinear mpc algorithm for real-time obstacle avoidance of ground vehicles," in *2013 European Control Conf. (ECC)*, July 2013, pp. 4136–4141.

[7] T. Mercy, W. V. Loock, and G. Pipeleers, "Real-time motion planning in the presence of moving obstacles," in *2016 European Control Conf. (ECC)*, June 2016, pp. 1586–1591.

[8] Salmah, Sutrisno, E. Joelianto, A. Budiyono, I. E. Wijayanti, and N. Y. Megawati, "Model predictive control for obstacle avoidance as hybrid systems of small scale helicopter," in *2013 3rd International Conf. on Instrumentation Control and Automation (ICA)*, Aug 2013, pp. 127–132.

[9] S. M. Erlien, S. Fujita, and J. C. Gerdes, "Shared steering control using safe envelopes for obstacle avoidance and vehicle stability," *IEEE Trans. on Intelligent Transportation Systems*, vol. 17, no. 2, pp. 441–451, Feb 2016.

[10] A. Bemporad and C. Rocchi, "Decentralized hybrid model predictive control of a formation of unmanned aerial vehicles," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 11 900 – 11 906, 2011, 18th IFAC World Congress.

[11] A. Bemporad, C. Pascucci, and C. Rocchi, "Hierarchical and hybrid model predictive control of quadcopter air vehicles," *IFAC Proceedings Volumes*, vol. 42, no. 17, pp. 14 – 19, 2009, 3rd IFAC Conf. on Analysis and Design of Hybrid Systems.

[12] J.-H. Chuang, "Potential-based modeling of three-dimensional workspace for obstacle avoidance," *IEEE Trans. on Robotics and Automation*, vol. 14, no. 5, pp. 778–785, Oct 1998.

[13] T. Paul, T. R. Krogstad, and J. T. Gravdahl, "Uav formation flight using 3d potential field," in *2008 16th Mediterranean Conf. on Control and Automation*, June 2008, pp. 1240–1245.

[14] T. Schouwenaars, B. D. Moor, E. Feron, and J. How, "Mixed integer programming for multi-vehicle path planning," in *2001 European Control Conf. (ECC)*, Sept 2001, pp. 2603–2608.

[15] J. Miura, "Support vector path planning," in *2006 IEEE/RSJ International Conf. on Intelligent Robots and Systems*, Oct 2006, pp. 2894–2899.

[16] N. Morales, J. Toledo, and L. Acosta, "Path planning using a multiclass support vector machine," *Applied Soft Computing*, vol. 43, pp. 498 – 509, 2016.

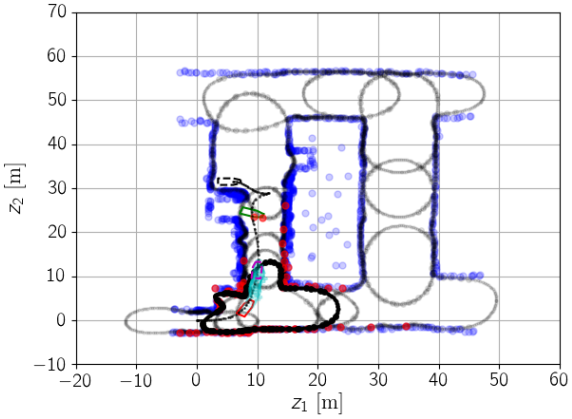
[17] H. Xu, Y. Gao, F. Yu, and T. Darrell, "End-to-end learning of driving models from large-scale video datasets," *CoRR*, vol. abs/1612.01079, 2016.

[18] M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba, "End to end learning for self-driving cars," *CoRR*, vol. abs/1604.07316, 2016.

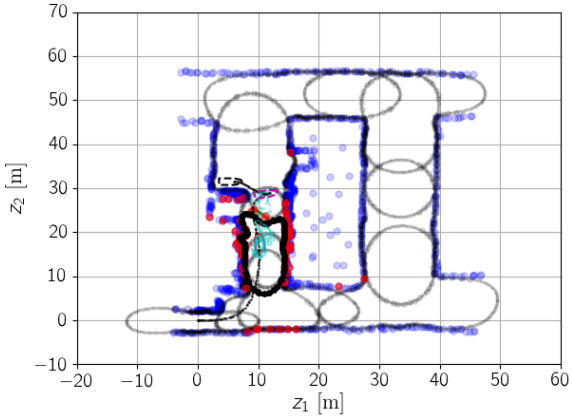
[19] L. Caltagirone, M. Bellone, L. Svensson, and M. Wahde, "Simultaneous perception and path generation using fully convolutional neural networks," *CoRR*, vol. abs/1703.08987, 2017.

[20] M. Bojarski, P. Yeres, A. Choromanska, K. Choromanski, B. Firner, L. D. Jackel, and U. Muller, "Explaining how a deep neural network trained with end-to-end learning steers a car," *CoRR*, vol. abs/1704.07911, 2017.

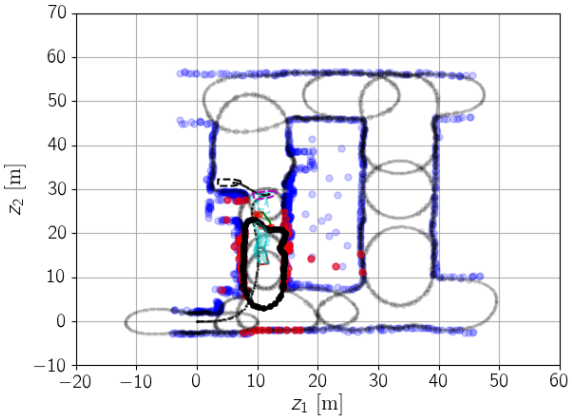
[21] A. Domokos, J. M. Ingram, and M. M. Marsh, "Projections onto closed convex sets in hilbert spaces," *Acta Mathematica Hungarica*, vol. 152, no. 1, pp. 114–129, Jun 2017.



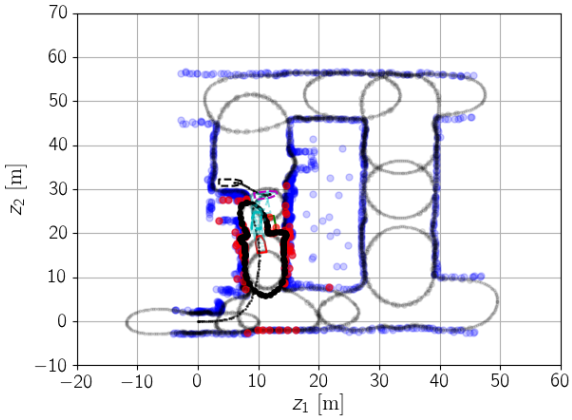
(a) Tracking on a turn



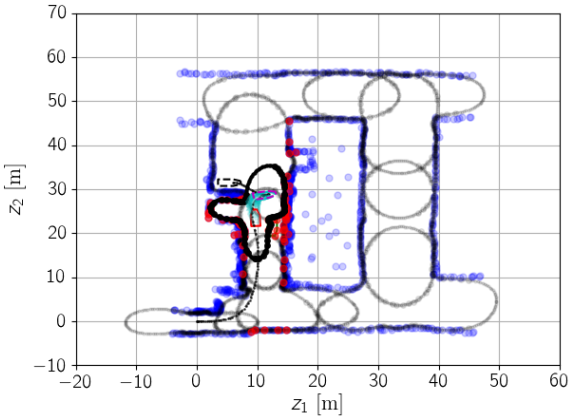
(b) Dynamic obstacle encountered



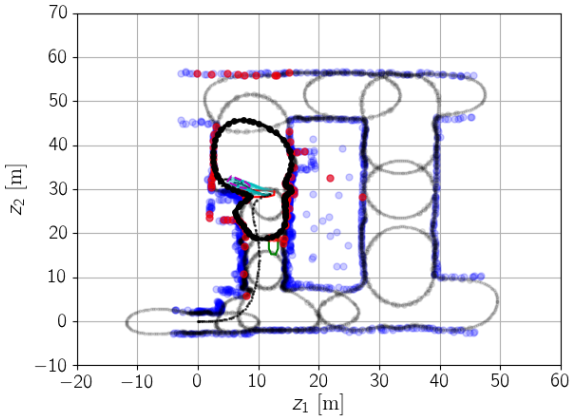
(c) Car reversing to avoid obstacle



(d) Space found after obstacle moved forward



(e) Transition into the reverse parking position



(f) Executing the reverse parking

Figure 5: Path Following MPC with Dynamic Obstacles ($\mathcal{M}_{c_1(t)}$ in bold black, unused manifolds in faded black, sensor data as red point cloud, q_{opt} in cyan, $q_{end}^{ref}(t)$ in dashed pink, q_{end} in dashed black, $q(t)$ in red, q_{ref} as black dotted line)