Robotica (2017) volume 35, pp. 1157–1175. © Cambridge University Press 2015
doi:10.1017/S026357471500106X

Dual objective motion planning subject to state constraints

Nader Sadegh*

The George W. Woodruff School of Mechanical Engineering, Georgia Institute of Technology, Atlanta, Georgia 30332–0405, USA

(Accepted November 13, 2015. First published online: December 14, 2015)

SUMMARY

This paper presents a novel motion planning approach inspired by the Dynamic Programming (DP) applicable to multi degree of freedom robots (mobile or stationary) and autonomous vehicles. The proposed discrete–time algorithm enables a robot to reach its destination through an arbitrary obstacle field in the fewest number of time steps possible while minimizing a secondary objective function. Furthermore, the resulting optimal trajectory is guaranteed to be globally optimal while incorporating state constraints such as velocity, acceleration, and jerk limits. The optimal trajectories furnished by the algorithm may be further updated in real time to accommodate changes in the obstacle field and/or cost function. The algorithm is proven to terminate in a finite number of steps without its computational complexity increasing with the type or number of obstacles. The effectiveness of the global and replanning algorithms are demonstrated on a planar mobile robot with three degrees of freedom subject to velocity and acceleration limits. The computational complexity of the two algorithms are also compared to that of an A*–type search.

KEYWORDS: Time-Optimal, Motion Planning Obstacle Avoidance, Dynamic Programming, Global Optimization, Spline, State Constraints, Multi-objective.

1. Introduction

Motion planning through an obstacle field is a critical task in controlling autonomous vehicles such as mobile robots, Unmanned Air Vehicles (UAV's), underwater vehicles, and exploratory rovers. No autonomous vehicle or robot can successfully operate in a constrained environment without a systematic mechanism for planning its motion path. The basic objective of most motion planning algorithms is to generate a feasible path starting from an arbitrary initial location to the final target while avoiding the obstacles. Moreover, the resulting trajectory is often preferred to be optimal with respect to a certain criterion such as time and subject to state constraints such as velocity and acceleration limits as well as the actuation inputs.

The optimization problem associated with most motion planning tasks is inherently non-convex[1] and difficult to solve.[2] The non-convexity of the problem is a direct consequence of the obstacle avoidance requirement and is further complicated by inclusion of the state and actuation constraints. Furthermore, changing and/or uncertain obstacle fields, which arise in most real–world applications, pose additional challenges in achieving the required solution to this problem.

The basic motion planning problem has been approached from several different angles. The classical Dijkstra's Algorithm[3] and its variation known as A*search [4] are among the most widely used motion planning algorithms. They both use graph search techniques to construct the shortest path between the source and destination (goal or target) nodes. The A* search algorithm employs a heuristic distance function to improve the efficiency of Dijkstra's Algorithm. Both algorithms require a discrete search domain (nodes) whose complexity may grow at least quadratically with the number of obstacles.[5] Furthermore, the generated optimal path may not conform to the robot's other state constraints such as velocity or acceleration. To address some of the weaknesses of the original graph

* Corresponding author. E-mail: sadegh@gatech.edu

https://doi.org/10.1017/S026357471500106X Published online by Cambridge University Press

search algorithms, a sampling–based motion planner has emerged in recent years[6,7] that uses the dynamic model of the robot to search for optimal trajectories. In particular, Dunlap *et al.*[6] employs a Sampling Based Model Predictive Controller (SBMPC) and Optimizer (SBMPO) to search for the robot's optimal control policy and the resulting trajectories using a Lifelong Planning A* (LPA*)[8] graph search algorithm. Moreover, as in this paper, it avoids explicit discretization of the state–space by sampling the space of all possible inputs.

The Potential Field Theory (PFT) is another widely used approach for robotic motion planning.[9–11] Rimon and Koditschek[9] developed a special artificial potential function, called a navigation function, that guarantees collision-free motion and convergence to the destination from almost all initial free configurations. Eichhorn[10] applied the PFT for trajectory planning of an underwater vehicle. Bruijnen *et al.*[11] proposed an extension of the PFT that can handle a changing obstacle field and generates a sub-optimal smooth path with bounds on the allowed velocity, acceleration and jerk. While the PFT is well suited for a varying obstacle fields, it does not necessarily generate a globally optimal path and in many instances it may render only a local optimum away from the destination. A recent work[12] partially addresses this deficiency by adding an extra control force to that of the PFT but falls short of guaranteeing convergence to the global optimal solution.

The third approach aimed at generating a global solution is based on Linear Programming (LP) and its variations.[13–15] The main idea behind this approach is to formulate the path planning problem utilizing linear objective function and constraints and solve the resulting LP problem. Even though this approach is well suited for obstacle free situations, the addition of obstacle avoidance constraints makes the optimization problem non-convex and dramatically increases its complexity.

This paper presents a *dual* objective motion planning algorithm based on the DP approach.[16] A key advantage of the DP compared to graph search algorithms is that it not only generates the optimal path but also the optimal control policy (e.g., the next point) for each location within the search space.[17] Therefore, the optimal policy may still be enforced even if the original optimal path is deviated from. The existing works on the applications of DP to motion planning has been the subject of numerous papers in the literature.[17–21] Sadegh[18] presented an earlier version of the current paper with only a single objective function and static obstacles. Nagabhushan and Manohara[19] incorporated a framed free space approach to reduce the computational complexity of the previous DP approaches. However, none of the aforementioned works guarantees time optimality combined with a secondary objective function. Most of them also do not consider state or input constraints and are limited to piecewise linear paths.

The motion planning algorithms presented here can be used to navigate a mobile robot through an arbitrary obstacle field. The primary objective is to generate a smooth spline path of the shortest duration that reaches the goal from an arbitrary state within its domain of operation while avoiding the obstacles. The secondary objective is to choose from among the time–optimal trajectories that also minimize a secondary cost function such as the path length, energy cost, etc. The degrees of freedom along with the level of spline smoothness can be chosen arbitrarily. In addition, the motion state (velocity, acceleration, etc.) and input constraints are naturally incorporated. Of the two algorithms presented in this paper, the first one generates an optimal control policy and a continuous secondary cost-to-go function for every admissible initial location (i.e., locations that can reach the goal in a finite number of steps) based on a continuous input space. It does not require discretization of either the input or the state–space but requires elementary set operations such as set union, intersection, and subtraction. The second half of the paper is devoted to a discrete implementation of the continuous algorithms drawing its control policy from a discrete set of inputs. Unlike the SBMPC, which samples the entire input space, it is shown here that only three input values (e.g., 0 and ± 1) suffice for each degree of freedom in $\mathbb{R}^n$ to accomplish the motion planning objective with a *prescribed* degree of accuracy. Both algorithms are proven to terminate in a finite number of steps and their complexities are shown not to increase with the number of obstacles. While the global DP planner allows for the robot to always return to the optimal trajectory even if it momentarily deviates from it, it assumes a fixed known environment (i.e., obstacles), which is often impractical. To relax this requirement, a more efficient replanning algorithm (similar to a backward A* and also D*–Lite[8] for the discrete version) is proposed to update the original optimal trajectory dynamically while the robot is searching for its destination allowing for variations in the obstacle field and other environmental effects. A three degree-of-freedom planar (translation plus rotation) robot subject to velocity and acceleration limits moving in a field of randomly placed rectangular and circular obstacles is used to

https://doi.org/10.1017/S026357471500106X Published online by Cambridge University Press

illustrate the effectiveness of this approach. The numerical study also demonstrates the effectiveness of the dynamic updating of the optimal trajectory in the presence of changing or moving obstacles.

This paper is organized as follows: Sections 2 and 3 present the Problem Statement and the basic Spline Trajectory Model used in this paper. Section 4 derives the key results on obstacle free spline trajectories needed in the main algorithm presented in Section 5. The discrete implementation of the main algorithm is given in Section 6 and its application to a three degree-of-freedom planar robot is presented in Section 7. Appendix Appendix A contains the technical proofs of the propositions and lemmas in the paper.

2. Problem Statement

Consider an n–dimensional open subset $\mathcal{X} \subset \mathbb{R}^n$ representing the geometric constraint associated with all the possible paths. Each path is specified as an mth order *spline* defined according to

$$\mathcal{S}_m = \left\{ \boldsymbol{x} : \bigcup_{k=0}^{N-1} [kT, (k+1)T] \to \mathcal{X},\ N = 1, 2, \ldots \right\} \tag{1}$$

where T represents the spline step–size (also referred to as the *sampling period*) and $[kT, (k+1)T]$ is the domain of the ith piece of the spline (an mth order polynomial). The length or duration of each $\boldsymbol{x} \in \mathcal{S}_m$ is the number of polynomials (N) associated with $\boldsymbol{x}$. The objective of the proposed optimization algorithm is twofold: (1) Identify the set of all possible splines $\boldsymbol{x}(\cdot) \in \mathcal{S}_m$ that connect an arbitrary initial position $\boldsymbol{x}(0) \in \mathcal{X}$ to a given target position $\boldsymbol{x}(N) = \boldsymbol{x}^* \in \mathcal{X}$ in the fewest number of steps(N) with bounded time derivatives: $|x_i^j| \leq x_{i,\max}^j$, $i = 1, \ldots n$, $j = 1, \ldots, m$, where $\boldsymbol{x}^j := \frac{d^j \boldsymbol{x}}{dt^j}$. (2) Select the spline(s) from those identified in (1) that minimizes the secondary objective function

$$J = \sum_{k=0}^{N-1} \psi(\boldsymbol{x}_k, \dot{\boldsymbol{x}}_k, \ldots, \boldsymbol{x}_k^m) \tag{2}$$

where $\psi : \mathbb{R}^n \times \cdots \times \mathbb{R}^n \to \mathbb{R}$ is a non-negative continuous function and $\boldsymbol{x}_k^j = \boldsymbol{x}^j(kT)$. We further assume that

- The set $\mathcal{X} \subset \mathbb{R}^n$ containing all possible paths is open, convex, and bounded.
- $\boldsymbol{x}_k \notin \mathcal{O} \subset \mathcal{X}$, where $\mathcal{O}$ is a closed subset of $\mathcal{X}$ representing the obstacles.

3. Spline Path Model

Each $\boldsymbol{x}(\cdot) \in \mathcal{S}_m$ spline can be represented by the differential equation $x_i^m(t) = u_i(t)$ where each $u_i(t) \in [-x_{i,\max}^m, x_{i,\max}^m]$ is a piecewise constant function such that $u_i(t) = u_{i,k}$ for $t_k \leq t \leq t_{k+1}$. Defining the ith partial state vector $\boldsymbol{\xi}_i = \left(x_i, x_i^1, \ldots, x_i^{m-1}\right)$, $i = 1, \ldots, n$, and integrating the resulting state equations for $t_k \leq t \leq t_{k+1}$ yields $\boldsymbol{\xi}_i(t) = \mathbf{A}_{t-t_k} \boldsymbol{\xi}_{i,k} + \mathbf{B}_{t-t_k} u_{i,k}$ where $\boldsymbol{\xi}_{i,k} = \boldsymbol{\xi}_i(t_k)$, $\mathbf{A}_\tau$ is an $m \times m$ upper triangular matrix whose ijth element, $A_{\tau,ij} = \frac{\tau^{(j-i)}}{(j-i)!}$, $j \geq i$, and $\mathbf{B}_\tau$ is an $m \times 1$ vector with its ith component $\mathbf{B}_{\tau,i} = \frac{\tau^{(m-i+1)}}{(m-i+1)!}$. For $t = t_{k+1}$, we have

$$\boldsymbol{\xi}_{i,k+1} = \mathbf{A}_T \boldsymbol{\xi}_{i,k} + \mathbf{B}_T u_{i,k}. \tag{3}$$

Rather than using $\boldsymbol{\xi} = (\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_n)$ as the full state vector, we employ a permutation of it, $\boldsymbol{z} := \left(\boldsymbol{x}, \boldsymbol{x}^1, \ldots, \boldsymbol{x}^{m-1}\right)$, which can incorporate the geometric and dynamic constraints more naturally. It can be seen that the transition from $\boldsymbol{z}_k = \boldsymbol{z}(t_k)$ to $\boldsymbol{z}_{k+1} = \boldsymbol{z}(t_{k+1})$ is given by $\boldsymbol{z}_{k+1} = \boldsymbol{\varphi}_+(\boldsymbol{z}_k, \boldsymbol{u}_k)$where $\boldsymbol{\varphi}_+(\boldsymbol{z}, \boldsymbol{u}) = (\mathbf{A}_T \otimes I_{n\times n})\boldsymbol{z} + (\mathbf{B}_T \otimes I_{n\times n})\boldsymbol{u}$, $\otimes$ is the Kronecker product, and $I_{n\times n}$ the $n \times n$ identity matrix. The backward state transition, $\boldsymbol{\varphi}_- : \mathbb{R}^{mn} \times \mathbb{R}^m \to \mathbb{R}^{mn}$, which maps $(\boldsymbol{z}_{k+1}, \boldsymbol{u}_k)$ to $\boldsymbol{z}_k$, is given by $\boldsymbol{\varphi}_-(\boldsymbol{z}, \boldsymbol{u}) = (\mathbf{A}_{-T} \otimes I_{n\times n})\boldsymbol{z} + (\mathbf{B}_{-T} \otimes I_{n\times n})\boldsymbol{u}$. We shall use the symbol $\mathcal{Z}$ to denote the subset of the state–space that meets the geometric ($\boldsymbol{x} \in \mathcal{X}$) and dynamic constraints: $\mathcal{Z} = \mathcal{X} \times \mathcal{X}^1 \times \cdots \times \mathcal{X}^{m-1}$, where $\mathcal{X}^j = \prod_{i=1}^n (-x_{i,\max}^j, x_{i,\max}^j)$. The space of all admissible inputs, $\boldsymbol{u}_k$'s, is the closure of $\mathcal{U} := \prod_{i=1}^n (-x_{i,\max}^m, x_{i,\max}^m)$ denoted by $\overline{\mathcal{U}}$.

https://doi.org/10.1017/S026357471500106X Published online by Cambridge University Press

The path planning algorithms to be presented will guarantee that the spline knots $\boldsymbol{x}_k$, $k = 1, 2, \ldots$, will be outside the obstacle set $\mathcal{O}_k := \mathcal{O}_{t_k}$. However, it is possible for the intermediary points on the spline to enter the obstacle space. To avoid this situation, the set of obstacles $\mathcal{O}$ may be slightly expanded beyond the actual 'hard' obstacles to ensure that the entire path in addition to the knots will stay outside of the obstacles. Let $\mathcal{H}$ be a closed set representing the hard obstacle set and $B_\varepsilon = \{\boldsymbol{x} : \|\boldsymbol{x}\| \leq \varepsilon\}$, the closed ball of radius ε for some $\varepsilon > 0$. The following proposition shows that if $\mathcal{O} = \mathcal{H} + B_\varepsilon$, then $\boldsymbol{x}(t)$ is guaranteed to stay outside of $\mathcal{H}$, $\forall t \in [t_k - T/2, t_k + T/2]$, if the discrete positions $\boldsymbol{x}_k \notin \mathcal{O}$ for a sufficiently small sampling period T.

Proposition 1. *Let $\mathcal{S}_m$ be the spline space of order m given by (1) and suppose that $\mathcal{O} = \mathcal{H} + B_\varepsilon$ such that $\boldsymbol{x}_k \notin \mathcal{O}$, $\forall k \geq 0$. Then $\boldsymbol{x} \notin \mathcal{H}$, $\forall \boldsymbol{x} \in \mathcal{S}_m$, $\forall m \geq 1$, provided that the sampling period $T < 2\ln(1+\varepsilon)/\boldsymbol{\sigma}$ where $\boldsymbol{\sigma} = \max_{1 \leq j \leq m} \|(x^j_{1,\max}, \ldots, x^j_{n,\max})\|^{1/j}$ and $\|\cdot\|$ is any vector norm.*

4. Obstacle Free Motion Planning

In this section, we present the main time–optimal optimization algorithm. In terms of the notations introduced in the preceding section, the main objective of this algorithm will be to transfer a given state $\boldsymbol{z} \in \mathcal{Z}$ to the target state $\boldsymbol{\tau} = (\boldsymbol{x}^*, 0, \ldots, 0)$ in the minimum number of steps possible along the trajectories that minimize the secondary objective function (2) while avoiding the obstacles We begin our development in the absence of any obstacles and then extend the results to incorporate a fairly general set of obstacles. It turns out that the obstacle free case plays a key role in establishing the convergence properties of the general case.

To state our first preliminary result, let $\mathcal{Z}_0$ be the set of equilibrium states in $\mathcal{Z}$: $\mathcal{Z}_0 = \{(\boldsymbol{x}, \boldsymbol{0}, \ldots, \boldsymbol{0}) : \boldsymbol{x} \in \mathcal{X}\}$. The following Lemma, which exploits the convexity of $\mathcal{X}$, shows that any two states in $\mathcal{Z}_0$ can be joined by a spline in a finite number of steps with the resulting path lying entirely in $\mathcal{Z}$.

Lemma 2. *There exists an integer K such that any two states $\boldsymbol{z}, \boldsymbol{z}' \in \mathcal{Z}_0$ can be joined by a spline lying entirely in $\mathcal{Z}$ in at most K steps. That is, there exists a sequence of inputs $u_0, u_1, \ldots, u_{K-1} \in \mathcal{U}$ such that if $\boldsymbol{z}_0 = \boldsymbol{z}$ then $\boldsymbol{z}_k = \boldsymbol{\varphi}_+(\boldsymbol{z}_{k-1}, \boldsymbol{u}_{k-1}) \in \mathcal{Z}$ and $\boldsymbol{z}_K = \boldsymbol{z}'$.*

Definition 3. *The* forward *reachable set of $\boldsymbol{z} \in \mathcal{Z}$ is the set of all states in $\mathcal{Z}$ that can be reached from $\boldsymbol{z}$ in a single step:*

$$\mathcal{R}_+(\boldsymbol{z}) = \{\boldsymbol{y} \in \mathcal{Z} : \boldsymbol{y} = \boldsymbol{\varphi}_+(\boldsymbol{z}, \boldsymbol{u}),\ \boldsymbol{u} \in \mathcal{U}\}.$$

The backward *reachable set from $\boldsymbol{z} \in \mathcal{Z}$ is the set of all states in $\mathcal{Z}$ that can reach $\boldsymbol{z}$ in a single step:*

$$\begin{aligned}\mathcal{R}_-(\boldsymbol{z}) &= \{\boldsymbol{y} \in \mathcal{Z} : \boldsymbol{z} = \boldsymbol{\varphi}_+(\boldsymbol{y}, \boldsymbol{u}),\ \boldsymbol{u} \in \mathcal{U}\} \\ &= \{\boldsymbol{y} \in \mathcal{Z} : \boldsymbol{y} = \boldsymbol{\varphi}_-(\boldsymbol{z}, \boldsymbol{u}),\ \boldsymbol{u} \in \mathcal{U}\}.\end{aligned}$$

If $\mathcal{W} \subset \mathcal{Z}$, then $\mathcal{R}_+(\mathcal{W}) = \bigcup_{z \in \mathcal{W}} \mathcal{R}(\boldsymbol{z})$ and $\mathcal{R}_-(\mathcal{W}) = \bigcup_{z \in \mathcal{W}} \mathcal{R}_-(\boldsymbol{z})$. The kth composition of $\mathcal{R}_\pm$, denoted by $\mathcal{R}^k_\pm$, is defined as follows: $\mathcal{R}^k_\pm(\mathcal{W}) = \mathcal{R}_\pm(\mathcal{R}^{k-1}_\pm(\mathcal{W}))$ with $\mathcal{R}^0_\pm(\mathcal{W}) = \mathcal{W}$. In addition, for each $\boldsymbol{z} \in \mathcal{R}_\pm(\mathcal{W})$, the set of all control inputs $\boldsymbol{u} \in \mathcal{U}$ that connects $\boldsymbol{z}$ to some $\boldsymbol{w} \in \mathcal{W}$ or vice versa is denoted by $\mathcal{C}_\pm(\boldsymbol{z}, \mathcal{W})$:

$$\mathcal{C}_\pm(\boldsymbol{z}, \mathcal{W}) = \{\boldsymbol{u} \in \mathcal{U} : \boldsymbol{\varphi}_\pm(\boldsymbol{z}, \boldsymbol{u}) \in \mathcal{W}\}.$$

The following Proposition summarizes some of the important properties of the forward and backward reachable sets to be utilized in the sequel:

Proposition 4. *The forward and backward operation $\mathcal{R}_\pm$ posses the following properties:*

(i) The sets $\mathcal{R}^k_\pm(\boldsymbol{\tau})$, $\forall k \geq m$, are open sets and non-decreasing: $\mathcal{R}^k_\pm(\boldsymbol{\tau}) \subset \mathcal{R}^{k+1}_\pm(\boldsymbol{\tau})$.
(ii) There exists a finite integer $K \geq 1$ such that $\mathcal{Z}_0 \subset \mathcal{R}^k_\pm(\boldsymbol{\tau})$, for all $\boldsymbol{\tau} \in \mathcal{Z}_0$ and $k \geq K$.
(iii) $\mathcal{R}_\pm\left(\bigcap_{k\geq 1} \mathcal{W}_k\right) \subset \bigcap_{k\geq 1} \mathcal{R}_\pm(\mathcal{W}_k)$, $\forall \mathcal{W}_k \subset \mathcal{Z}$.
(iv) $\mathcal{R}_\pm\left(\bigcup_{k\geq 1} \mathcal{W}_k\right) = \bigcup_{k\geq 1} \mathcal{R}_\pm(\mathcal{W}_k)$, $\forall \mathcal{W}_k \subset \mathcal{Z}$.

https://doi.org/10.1017/S026357471500106X Published online by Cambridge University Press

Before presenting the general case, we consider the motion planning in the absence of any obstacles. The basic idea used here is inspired by the DP[16] approach. Starting from the target we first determine the set of all states that can reach the target in a single step, or simply $\mathcal{R}_-(\boldsymbol{\tau})$. Having identified $\mathcal{R}_-(\boldsymbol{\tau})$, we will then recursively expand our search to the sets $\mathcal{R}_-^2(\boldsymbol{\tau}), \mathcal{R}_-^3(\boldsymbol{\tau}), \ldots, \mathcal{R}_-^k(\boldsymbol{\tau})$, where as defined earlier, $\mathcal{R}_-^k(\boldsymbol{\tau})$ represents the set of all states within $\mathcal{Z}$ that can reach the target in k steps with all the intermediary states lying in $\mathcal{Z}$. This process continues until $\mathcal{R}_-^k(\boldsymbol{\tau})$ covers the entire equilibrium space $\mathcal{Z}_0$. Theorem 4(ii) justifies the viability of this approach by furnishing a finite integer K for which the set $\mathcal{R}_-^k(\boldsymbol{\tau})$ (and also $\mathcal{R}_+^k(\boldsymbol{\tau})$), $\forall k \geq K$, for an arbitrary state in $\boldsymbol{\tau} \in \mathcal{Z}_0$ fills the entire $\mathcal{Z}_0$. We shall refer to $\mathcal{D} := \mathcal{R}_-^K(\boldsymbol{\tau}) \supset \mathcal{Z}_0$ as the algorithm's *domain of operation.*

5. Obstacle Avoidance Motion Planning

In this section, we present the main path planning algorithm that incorporates both the obstacle avoidance requirement as well as the state constraints. The obstacle space including the state constraints is a subset of $\mathcal{Z}$ given by $\Theta := \mathcal{O} \times \mathcal{X}^1 \times \cdots \times \mathcal{X}^{m-1}$. Starting with the target set $\mathcal{W}_0 = \{\boldsymbol{\tau}\} \notin \Theta$, we first identify the set of all states except for those within the obstacle set Θ that can reach the target in a single step. We will then recursively remove the obstacle set Θ from each backward reachable space until all the states in $\mathcal{Z}_0$ are covered. At each step of the DP backward recursion, we assign the *next* input to each visited state that minimizes the objective function (2) starting from that state. We will prove that the algorithm terminates in finite number of steps not depending on the number of obstacles. The only restriction that we place on the set of obstacles is that it is a closed subset of the space $\mathcal{X}$ without additional requirements such as convexity.

Algorithm 5. Given the constrained state–space $\mathcal{Z} = \mathcal{X} \times \mathcal{X}^1 \times \cdots \times \mathcal{X}^{m-1}$, the obstacle set $\Theta := \mathcal{O} \times \mathcal{X}^1 \times \cdots \times \mathcal{X}^{m-1}$, an open set $\mathcal{O}_o \subset \mathcal{X}$ containing $\mathcal{O}$, target set $\{\boldsymbol{\tau}\}$, and secondary objective function (2) execute the following steps:

(i) Initialize the set $\mathcal{W}_0 = \{\boldsymbol{\tau}\}$ and the secondary cost-to-go, $s_0 = 0$.

(ii) For $k = 1, 2, \ldots$update $\mathcal{W}_k$: $\mathcal{W}_k = \mathcal{R}_-(\mathcal{W}_{k-1} - \Theta)$. To each $\boldsymbol{z} \in \overline{\mathcal{W}_k} - \overline{\mathcal{W}_{k-1}}$ assign the *next* input $\boldsymbol{u}^* \in \overline{\mathcal{U}}$ according to

$$\boldsymbol{u}^*(\boldsymbol{z}) = \arg \min_{\boldsymbol{u} \in \mathcal{C}_+(\boldsymbol{z}, \overline{\mathcal{W}_{k-1}})} \psi(\boldsymbol{z}, \boldsymbol{u}) + s_{k-1} \circ \boldsymbol{\varphi}_+(\boldsymbol{z}, \boldsymbol{u})$$

where $\mathcal{C}_+(\boldsymbol{z}, \overline{\mathcal{W}}) = \left\{\boldsymbol{u} \in \overline{\mathcal{U}} : \boldsymbol{\varphi}_+(\boldsymbol{z}, \boldsymbol{u}) \in \overline{\mathcal{W}}\right\}$. Update the kth cost-to-go function to $s_k(\boldsymbol{z}) = \psi(\boldsymbol{z}, \boldsymbol{u}^*) + s_{k-1} \circ \boldsymbol{\varphi}_+(\boldsymbol{z}, \boldsymbol{u}^*), \forall \boldsymbol{z} \in \overline{\mathcal{W}_k} - \overline{\mathcal{W}_{k-1}}$.

(iii) Continue until step k^*at which an upper bound for $\mathcal{W}_k$ is established in the following sense: $\left(\mathcal{W}_k - \bar{\mathbf{A}}_T^m \Theta_o\right) \cap \mathcal{Z}_m \subset \bar{\mathbf{A}}_T^m \mathcal{W}_{k^*}, \forall k \geq k^*$, where $\mathcal{Z}_m = \bigcap_{k=0}^{k=m} \bar{\mathbf{A}}_T^k \mathcal{Z}$ and $\Theta_o = \mathcal{O}_o \times \mathcal{X}^1 \times \cdots \times \mathcal{X}^{m-1}$.

Remark 6. The direct implementation of Algorithm 5 requires parametrization of the various sets involved and availability of elementary set operations such as union, intersection, and subtraction. While efficient algorithms do exist to manipulate convex sets, which is the case for the obstacle free case, the majority of the sets here are not necessarily convex making direct implementation of the algorithm challenging. Even though it is possible to decompose non-convex sets into convex,[22] we will employ a simple input space discretization later in the paper (6) to implement the aforementioned algorithm.

The next lemma, which provides an alternative representation for the identified sets $\mathcal{W}_k$, is useful in proving the convergence of the algorithm.

Lemma 7. *Let $\mathcal{W}$ and $\mathcal{W}^+$be open subsets and $\mathcal{B}$ a closed subset of $\mathcal{Z}$ such that $\mathcal{W} \subset \mathcal{W}^+$. Then $\mathcal{R}_-(\mathcal{W} - \mathcal{B}) = \mathcal{R}_-\left(\mathcal{W}^+\right) - \mathcal{V}$ where $\mathcal{V} = \widetilde{\mathcal{R}_-(\mathcal{W} - \mathcal{B})}$ is a closed subset of $\tilde{\mathcal{Z}}$ satisfying*

$$\mathcal{R}_+(\mathcal{V}) \cap \mathcal{W} \subset \mathcal{B}.$$

At each step of Algorithm 5, the next state (and input) is selected based on minimizing the secondary cost-to-go function $s_k(\boldsymbol{z})$. Even though $s_k(\boldsymbol{z})$ may not be continuous everywhere, the next

https://doi.org/10.1017/S026357471500106X Published online by Cambridge University Press

Lemma will be used to show that it is at least *lower semicontinuous* (l.s.c.). Semicontinuous functions are a generalization of continuous functions that enjoy many of the same properties such as attaining their minimum or maximum values on compact sets.[23]

Lemma 8. *Let $\mathcal{W}$ and $\mathcal{W}^+$ be compact subsets of $\mathcal{Z}$, and suppose that $f : \mathcal{W} \times \overline{\mathcal{U}} \to \mathbb{R}$ is a lower semicontinuous (l.s.c.) non-negative function. Then $f^* : \mathcal{W} \to \mathbb{R}$ given by*

$$f^*(\boldsymbol{z}) := \inf_{\boldsymbol{u} \in \mathcal{C}_+(\boldsymbol{z}, \mathcal{W}^+)} f(\boldsymbol{z}, \boldsymbol{u})$$

is well–defined and l.s.c Moreover, there exists a $\boldsymbol{u}^* \in \overline{\mathcal{U}}$ such that $f^*(\boldsymbol{z}) = f(\boldsymbol{z}, \boldsymbol{u}^*)$.

Applying Lemma 7 to $\mathcal{W}_2 = \mathcal{R}_-(\mathcal{W}_1 - \Theta)$ gives $\mathcal{W}_2 = \mathcal{R}_-(\mathcal{W}_1) - \mathcal{V}_2$ where $\mathcal{V}_2$ is a closed set satisfying $\mathcal{R}_+(\mathcal{V}_2) \cap \mathcal{W}_1 \subset \Theta$. Defining $\Theta_2 = \mathcal{V}_2 \cup \Theta$, then it can be seen that $\mathcal{W}_2 - \Theta = \mathcal{R}^2_-(\boldsymbol{\tau}) - \Theta_2$. Applying the lemma once more to $\mathcal{W}_2 - \Theta$ gives $\mathcal{R}_-(\mathcal{W}_2 - \Theta_2) = \mathcal{R}^3_-(\boldsymbol{\tau}) - \mathcal{V}_3$ where the closed set $\mathcal{V}_3$ satisfies $\mathcal{R}_+(\mathcal{V}_3) \cap \mathcal{R}^2_-(\boldsymbol{\tau}) \subset \Theta_2$. Continuing this procedure recursively using a simple induction, we arrive at the next proposition.

Proposition 9. *The set $\mathcal{W}_k$ in Algorithm 5 satisfies the following recursive relationships $\mathcal{W}_k = \mathcal{R}^k_-(\boldsymbol{\tau}) - \mathcal{V}_k$, $k \geq 1$, where $\mathcal{V}_k$ is a closed set satisfying $\mathcal{R}_+(\mathcal{V}_k) \cap \mathcal{R}^{k-1}_-(\boldsymbol{\tau}) \subset \Theta_{k-1}$, $\Theta_k = \mathcal{V}_k \cup \Theta$ and $\mathcal{V}_1 = \Theta$.*

Remark 10. The sets Θ_k, $k = 1, 2, \ldots$, which is an expansion of the obstacle set $\Theta \subset \mathcal{Z}$ contains all the states in $\mathcal{R}^k_-(\boldsymbol{\tau})$ that cannot reach the target in k steps or fewer (i.e., *blocked* states). Of course, in general, there may be states that can never reach the target. We will show that the main algorithm identifies all the blocked equilibrium states in a *finite* number of steps.

The next definition formalizes the notion of the blocked states alluded to in the preceding remark.

Definition 11. *Let $\mathcal{D} := \bigcup_{k \geq 1} \mathcal{R}^k_-(\boldsymbol{\tau})$ be the set of states in $\mathcal{Z}$ that can reach the target $\boldsymbol{\tau}$ in a finite number of steps in the* absence *of any obstacles. The set $\mathcal{V}_\infty = \bigcap_{k \geq K} \mathcal{V}_k$ of states in $\mathcal{D}$ that cannot reach the target in a finite number of steps in the presence of obstacles are called the blocked states. The sets $\mathcal{D}$ and $\mathcal{W}_\infty := \mathcal{D} - \mathcal{V}_\infty$ shall be referred to as the domain of operation and the admissible sets, respectively.*

As was shown in Proposition 9 the set of blocked states can be determined indirectly by the algorithm as a limit of $\mathcal{V}_k$ as k approaches infinity. The next lemma shows that once the domain of operation $\mathcal{D}$, which is independent of the obstacles, is known the set of blocked states $\mathcal{V}_\infty$ can be identified directly without going through an iterative process.

Lemma 12. *The set $\mathcal{V}_\infty = \bigcap_{k \geq 1} \mathcal{V}_k$ of blocked states introduced in Definition 11 satisfies the relationship $\mathcal{R}_+(\mathcal{V}_\infty) \cap \mathcal{D} \subset \mathcal{V}_\infty \cup \Theta$. Furthermore, for any open $\mathcal{O}_o \supset \mathcal{O}$, there exists an open subset $\mathcal{V}_o$ of $\mathcal{Z}$ containing $\mathcal{V}_\infty$ such that $\mathcal{V}_o \subset \bar{\mathbf{A}}_T^{-m} \mathcal{V}_\infty \cup \Theta_o$ where $\Theta_o = \mathcal{O}_o \times \mathcal{X}^1 \times \cdots \times \mathcal{X}^{m-1}$. For $m = 1$, the conclusion is strengthen to $\mathcal{V}_o \cup \mathcal{O} = \mathcal{V}_\infty \cup \mathcal{O}$.*

The convergence proof of Algorithm 5 is provided in the following theorem.

Theorem 13. *The trajectory optimization Algorithm 5 terminates in a finite number of steps K for an arbitrary open set $\mathcal{O}_o \subset \mathcal{X}$ containing the obstacle set $\mathcal{O}$. Upon termination, it generates the spline trajectory that connects each $\boldsymbol{x} \in \mathcal{W}_K$ to the target $\boldsymbol{\tau}$ in the fewest number of steps while minimizing the secondary objective function (2). Moreover, it generates a l.s.c optimal secondary cost-to-go function $s_k : \overline{\mathcal{W}_k} - \overline{\mathcal{W}_{k-1}} \to \mathbb{R}$ at each step $k \geq m$ of the algorithm.*

Proof. Applying the backward operation, $\mathcal{R}_-$, to $\mathcal{W}_{k-1} - \Theta = \mathcal{R}^{k-1}_-(\boldsymbol{\tau}) - \mathcal{V}_{k-1} \cup \Theta$ using Lemma (7) and $\mathcal{R}_-(\mathcal{D}) = \mathcal{D}$ yields $\mathcal{W}_k = \mathcal{D} - \mathcal{V}_k$. The set $\mathcal{W}_k$ represents the set of *all* sates that can reach the target in at most k steps. Let $\Theta_o \subset \mathcal{Z}$ be the open set given in Lemma 12 satisfying $\mathcal{V}_o \subset \bar{\mathbf{A}}_T^{-m} \mathcal{V}_\infty \cup \Theta_o$ where $\mathcal{V}_\infty = \lim_{k \to \infty} \mathcal{V}_k = \bigcap_{k \geq K} \mathcal{V}_k$ as before. The intersection of the *closed* subsets $\tilde{\mathcal{V}}_k = \mathcal{V}_k - \mathcal{V}_o$ of $\bar{\mathcal{Z}}$ are seen to be $\bigcap_{k \geq 1} \tilde{\mathcal{V}}_k = \mathcal{V}_\infty - \mathcal{V}_o = \varnothing$. By the Henie–Borel property of compact sets, there exists a finite $k^* \geq 1$ such that $\bigcap_{k=1}^{k^*} \tilde{\mathcal{V}}_k = \varnothing$ implying that $\mathcal{V}_k \subset \mathcal{V}_o \subset \bar{\mathbf{A}}_T^{-m} \mathcal{V}_\infty \cup \Theta_o$, $\forall k \geq k^*$. Intersecting both sides of $\mathcal{W}_k = \mathcal{D} - \mathcal{V}_k$ with $\mathcal{Z}_m = \bigcap_{k=0}^{k=m} \bar{\mathbf{A}}_T^k \mathcal{Z}$

https://doi.org/10.1017/S026357471500106X Published online by Cambridge University Press

and applying $\bar{\mathbf{A}}_T^{-m}$ to the result using that $\bar{\mathbf{A}}_T^{-1}\mathcal{D} \cap \mathcal{Z} \subset \mathcal{D}$, $\mathcal{V}_\infty \subset \mathcal{V}_k$, and $\mathcal{V}_o \subset \bar{\mathbf{A}}_T^{-m}\mathcal{V}_\infty \cup \Theta_o$, we have

$$\bar{\mathbf{A}}_T^{-m}(\mathcal{W}_k \cap \mathcal{Z}_m) - \Theta_o \subset \mathcal{D} - \mathcal{V}_o$$

$\forall k \geq k^*$ including $k = \infty$. This together with $\mathcal{W}_{k^*} \supset \mathcal{D} - \mathcal{V}_o$ implies that

$$(\mathcal{W}_k - \bar{\mathbf{A}}_T^m \Theta_o) \cap \mathcal{Z}_m \subset \bar{\mathbf{A}}_T^m \mathcal{W}_{k^*}$$

For $m = 1$, the open set $\bar{\mathbf{A}}_T^m \Theta_o$ is replaced by $\mathcal{O}$ and the conclusion follows from Lemma 12.

We first prove the secondary optimization results for $k = 1$. The cost-to-go $s_1(\boldsymbol{z}) = \psi(\boldsymbol{z}, \boldsymbol{u})$ from a given state $\boldsymbol{z} \in \mathcal{W}_1$ where $\boldsymbol{u}$ satisfies $\boldsymbol{z} = \boldsymbol{\varphi}_+(\boldsymbol{\tau}, \boldsymbol{u})$. From (3), it can be seen that $\boldsymbol{u} = \frac{m!}{\tau^m}(\pi(\boldsymbol{z}) - \pi(\boldsymbol{\tau}))$, which depend continuously on $\boldsymbol{z}$. Thus $s_1 : \mathcal{W}_1 \to \mathbb{R}$ is continuous hence l.s.c. The remainder of the proof proceeds by induction. Suppose that $s_k : \tilde{\mathcal{W}}_k \to \mathbb{R}$, $\tilde{\mathcal{W}}_k := \overline{\mathcal{W}_k - \mathcal{W}_{k-1}}$, is l.s.c and non-negative for $k > m$. Letting $f_k(\boldsymbol{z}, \boldsymbol{u}) = \psi(\boldsymbol{z}, \boldsymbol{u}) + s_k \circ \boldsymbol{\varphi}_+(\boldsymbol{z}, \boldsymbol{u})$, $\forall \boldsymbol{z} \in \tilde{\mathcal{W}}_{k+1}$, then the hypothesis of Lemma 8 is satisfied with $\mathcal{W} = \tilde{\mathcal{W}}_{k+1}$, $\mathcal{W}^+ = \overline{\mathcal{W}_k}$, and $f = f_k$. Thus there exists $\boldsymbol{u}^* \in \overline{\mathcal{U}}$ such that $s_{k+1} : \tilde{\mathcal{W}}_{k+1} \to \mathbb{R}$ given by

$$s_{k+1}(\boldsymbol{z}) = f_k(\boldsymbol{z}, \boldsymbol{u}^*) = \inf_{\boldsymbol{u} \in \mathcal{C}_+(\boldsymbol{z}, \bar{\mathcal{W}}_k)} f_k(\boldsymbol{z}, \boldsymbol{u})$$

is a well-defined and l.s.c on $\tilde{\mathcal{W}}_{k+1}$. Thus every $s_k : \tilde{\mathcal{W}}_k \to \mathbb{R}$ is well–defined and l.s.c, which completes the proof. □

Remark 14. As evidenced from the proof of the theorem, the execution time of algorithm 5 does not necessarily grow with the number of obstacles as many of the existing computational algorithms do. Instead, it is exactly equal to the maximum number of steps required to connect any point in the admissible space $\mathcal{X} - \mathcal{O}_0$ to the target while minimizing the secondary objective function and satisfying all the geometric and state constraints.

5.1. Motion replanning for changing obstacles

The motion planning Algorithm 5 presented earlier requires full knowledge of the obstacles to generate the robot's global navigation map toward the target. Even though the global map can be updated by rerunning the algorithm every time a change in the environment is detected, the complete recalculation of the global map is both inefficient and unnecessary. In this subsection, we introduce a local extension of Algorithm 5 that can efficiently update the global map when a change in the obstacle field represented by the set O is detected. The local updating is performed within a smaller domain surrounding the robot's current position than that of the global algorithm. As a consequence, far fewer states are needed to be expanded in order to generate the updated trajectories. To describe the details of this extension, let $\boldsymbol{\sigma} \in Z$ and $\boldsymbol{\tau} \in Z$ (as before) denote the starting and target states, respectively. Now suppose that $\boldsymbol{\sigma}$ can be transferred to the target $\boldsymbol{\tau}$ in $k_{\boldsymbol{\tau}}$ steps in an obstacle free environment, and in $k_{\boldsymbol{\sigma}} \geq k_{\boldsymbol{\tau}}$ steps while avoiding the current obstacle set Θ. Alternatively, the starting point $\boldsymbol{\sigma} \in R_-^{k_{\boldsymbol{\tau}}}(\boldsymbol{\tau}) \cap W_{k_{\boldsymbol{\sigma}}}$ where $R_-^k(\boldsymbol{\tau})$ represents the kth backward reachable set from $\boldsymbol{\tau}$ and W_k the set of all admissible states that can reach $\boldsymbol{\tau}$ in k steps while avoiding Θ. Before discussing the search details, we need to define the sequential length of a state $\boldsymbol{z} \in Z$ from $\boldsymbol{\sigma}$ to $\boldsymbol{\tau}$:

Definition 15. *The sequential length of $\boldsymbol{z} \in Z$ from $\boldsymbol{\sigma}$ to $\boldsymbol{\tau}$ is defined to be $\ell(\boldsymbol{z}) := \min_j \{j : \boldsymbol{z} \in Q_j\}$ where $Q_j = \bigcup_{0 \leq k \leq j} W_k \cap R_+^{j-k}(\boldsymbol{\sigma})$.*

It follows from Definition 15 that $\ell(\boldsymbol{z}) = k^*$ if and only if there exists a sequence $\{\boldsymbol{\tau}, \boldsymbol{z}_1, \ldots, \boldsymbol{z}_{k^*-1}, \boldsymbol{\sigma}\}$ of length k^* such that $\boldsymbol{z}_j = \boldsymbol{z}$ for some $0 \leq j \leq k^*$, $\boldsymbol{z}_k \in W_k$ for $0 \leq k \leq j$, and $\boldsymbol{z}_k \in R_+^{\ell-k}(\boldsymbol{\sigma})$ for $j \leq k \leq k^*$. Consequently, $\ell(\boldsymbol{\sigma}) = k_{\boldsymbol{\sigma}}$ and $\ell(\boldsymbol{\tau}) = k_{\boldsymbol{\tau}} \leq k_{\boldsymbol{\sigma}}$. The kth local search population, denoted by P_k, is initialized to $\{\boldsymbol{\tau}\}$, and expanded backwardly from the states with minimal sequential length according to (4) until it reaches the starting state $\boldsymbol{\sigma}$:

$$\mathcal{P}_{k+1} = (\mathcal{P}_k - Q_{l_k}) \cup \left(\mathcal{R}_-(Q_{l_k} \cap \mathcal{P}_k) - \Theta\right), \mathcal{P}_0 = \{\boldsymbol{\tau}\} \tag{4}$$

https://doi.org/10.1017/S026357471500106X Published online by Cambridge University Press

where $l_k = \min_j \{ j : \mathcal{P}_k \cap Q_j \neq 0 \}$ is the minimum sequential length of all the states in P_k and Q_{l_k} introduced in Definition (15) is the set of all states with sequential length of at most l_k. To choose the time optimal trajectory that minimizes the secondary cost, for states of equal sequential lengths, the optimal input $u^* \in C_+(z, \overline{\mathcal{P}_{k-1}})$ corresponding to $z \in \overline{\mathcal{P}_k} - \overline{\mathcal{P}_{k-1}}$ is given by

$$\boldsymbol{u}^*(z) = \arg \min_{\boldsymbol{u} \in \mathcal{C}_+(z, \overline{\mathcal{P}_{k-1}})} \psi(z, \boldsymbol{u}) + s_{k-1} \circ \varphi_+(z, \boldsymbol{u}) \tag{5}$$

where ψ is the current incremental cost function and $C_+(z, \overline{\mathcal{P}}) = \{ \boldsymbol{u} \in \overline{\mathcal{U}} : \boldsymbol{\varphi}_+(z, \boldsymbol{u}) \in \overline{\mathcal{P}} \}$ as before. The resulting cost-to-go function $s_k : \overline{\mathcal{P}_k} - \overline{\mathcal{P}_{k-1}} \to \mathbb{R}$, $s_0 = 0$, is updated according to Algorithm 5(ii) for the new members of the search population at each step.

The population updating (4) approach employed here is comparable to those in some of the graph based algorithms such as A* and D*–Lite with the sequential length used in lieu of the heuristic based aggregate cost to limit the population growth. It is also worth mentioning that the backward nature of the population search starting from the target matches those of D* and D*–Lite. The following Theorem (see the Appendix for the proof) shows that the local search given by (4) terminates in a few iterations depending on the sequential lengths of σ to τ.

Theorem 16. *The population set P_k updated according to (4) terminates in at most $k_{\max} := k_\tau + \frac{1}{2}(k_\sigma - k_\tau)(k_\sigma + k_\tau - 3)$ iterations where $k_\sigma = \ell(\sigma)$ and $k_\tau = \ell(\tau) \leq k_\sigma$. Furthermore, the optimal control sequence $\{ \boldsymbol{u}_k^* \}_{k=0}^{k=k_\sigma}$ specified by (5) transfers σ to τ in exactly k_σ steps while minimizing the secondary cost function(2).*

6. Discrete Implementation of the Optimization Algorithms

The trajectory optimization algorithm presented in the preceding section may be implemented by discretizing the input space. The simplest choice, and the most efficient one at each step of the backward search, is a bang–bang input taking on the discrete values $\pm u_{i,\max}$ and zero where $u_{i,\max} \leq x^m_{i,\max}$. This choice is further justified by the fact that the time optimal input for each dimension subject to input constraint is indeed of a bang–bang type.[16] According to Lemma 2 and its discrete counterpart (Lemma 19) this type of input discretization guarantees that any two points in $\mathcal{Z}_0$ can be connected to one another arbitrarily close for a sufficiently small sampling period T. Let $\widehat{\mathcal{U}} = \{ (u_1, \ldots, u_n) : u_i = 0 \text{ or } u_i = \pm u_{i,\max} \}$ denote the set of discretized inputs. Clearly the set of all the states that can reach the target with $\boldsymbol{u} \in \widehat{\mathcal{U}}$ is a discrete subset of the operation domain $\mathcal{D}$. Later in this section, we shall show that the equilibrium states contained in this discrete subset approximate $\mathcal{Z}_0$ to within a prescribed degree of accuracy by choosing the sampling period T sufficiently small. The optimization of the secondary objective function carries over to the discrete case without any significant changes. To this end, we first define the discrete counterpart of the forward and backward reachable sets.

Definition 17. *The discrete forward and backward reachable sets for any $z \in \mathcal{Z}$ are given by $\hat{\mathcal{R}}_\pm(z) = \{ \boldsymbol{y} \in \mathcal{Z} : \boldsymbol{y} = \boldsymbol{\varphi}_\pm(z, \boldsymbol{u}),\ \boldsymbol{u} \in \widehat{\mathcal{U}} \}$. The kth composition of $\hat{\mathcal{R}}_\pm$, denoted by $\hat{\mathcal{R}}^k_\pm$, is defined as follows: $\hat{\mathcal{R}}^k_\pm(\mathcal{W}) = \hat{\mathcal{R}}_\pm(\hat{\mathcal{R}}^{k-1}_\pm(\mathcal{W}))$ with $\hat{\mathcal{R}}^0_\pm(\mathcal{W}) = \mathcal{W}$.*

The following Proposition shows that the discretization of the input space leads to a 'natural' discretization of the state–space and a maximal reachable state.

Proposition 18. *For a given space $\mathcal{Z}$, discrete input space $\widehat{\mathcal{U}}$, and sampling period T, there exists an integer $K_T \geq 1$ such that $\hat{\mathcal{R}}^k_\pm(\tau) = \hat{\mathcal{R}}^{K_T}_\pm(\tau)$, $\forall k \geq K_T$.*

The following lemma shows that any two arbitrary positions in $\mathcal{X}$ can be *approximately* connected by means of a finite length spline to within a prescribed degree of accuracy. To state the lemma, let $B_\varepsilon = \{ \boldsymbol{x} \in \mathbb{R}^n : \|\boldsymbol{x}\| \leq \varepsilon \}$ be the closed ball of radius ε centered at the origin and define as follows:

$$\mathcal{X}_\varepsilon := \{ \boldsymbol{x} \in \mathcal{X} : \boldsymbol{x} + B_\varepsilon \subset \mathcal{X} \} \text{ and } \mathcal{Z}_0^\varepsilon = \{ z \in \mathcal{Z}_0 : \pi(z) \in \mathcal{X}_\varepsilon \}.$$

It can be easily seen that the set $\mathcal{X}_\varepsilon$ is an open and convex subset of $\mathbb{R}^n$ and $\mathcal{X}_\varepsilon + B_\varepsilon \subset \mathcal{X}$.

https://doi.org/10.1017/S026357471500106X Published online by Cambridge University Press

Lemma 19. *For any $\varepsilon > 0$ there exists a sampling period T, and a positive integer K_ε such that given any two equilibrium states $\boldsymbol{w}, \boldsymbol{z} \in \mathcal{Z}_0^\varepsilon$, a third equilibrium state $\hat{\boldsymbol{z}} \in \hat{\mathcal{R}}_\pm^k(\boldsymbol{w}) \cap \mathcal{Z}_0$, $\forall k \geq K_\varepsilon$, can be chosen to approximate $\boldsymbol{z}$ to within ε: $\|\hat{\boldsymbol{z}} - \boldsymbol{z}\| \leq \varepsilon$. In other words, $\hat{\mathcal{R}}_\pm^k(\boldsymbol{w}) + B_0^\varepsilon \supset \mathcal{Z}_0$, $\forall k \geq K_\varepsilon$, where $B_0^\varepsilon = B_\varepsilon \times \mathbf{0} \times \cdots \times \mathbf{0}$.*

Combining the results of Proposition 18 and Lemma 19, we may conclude that there exists a maximal reachable set $\hat{\mathcal{D}}$ that can approximate the equilibrium states in $\mathcal{Z}_0$ to within a prescribed degree of accuracy.

Theorem 20. *For $\varepsilon > 0$ there exist a sampling period T and a positive integer $\hat{K}$ such that $\hat{\mathcal{R}}_-^k(\boldsymbol{\tau}) = \hat{\mathcal{R}}_-^{\hat{K}}(\boldsymbol{\tau})$, and $\hat{\mathcal{R}}_-^k(\boldsymbol{\tau}) + B_0^\varepsilon \supset \mathcal{Z}_0$, $\forall k \geq \hat{K}$. Therefore, the maximal reachable set $\hat{\mathcal{D}} := \hat{\mathcal{R}}_-^{\hat{K}}(\boldsymbol{\tau})$ approximates any equilibrium state to within ε, i.e., $\forall \boldsymbol{z} \in \mathcal{Z}_0, \exists\, \hat{\boldsymbol{z}} \in \hat{\mathcal{D}}$ such that $\|\hat{\boldsymbol{z}} - \boldsymbol{z}\| \leq \varepsilon$.*

Proof. For $\varepsilon > 0$, choose $K_\varepsilon \geq 1$ and the sampling period T in accordance with Lemma 19 so that $\hat{\mathcal{R}}_-^k(\boldsymbol{\tau}) + B_0^\varepsilon \supset \mathcal{Z}_0$, $\forall k \geq K_\varepsilon$. Subsequently, choose the positive integer K_T as prescribed in Proposition 18 so that $\hat{\mathcal{R}}_-^k(\boldsymbol{\tau}) = \hat{\mathcal{R}}_-^{K_T}(\boldsymbol{\tau})$, $\forall k \geq K_T$. Letting $\hat{K} = \max(K_T, K_\varepsilon)$ proves the proposition. □

We now present the discrete version of the optimization algorithm. The difference between the two is that the discrete input set $\widehat{\mathcal{U}}$ is used in place of the continuous $\mathcal{U}$.

Algorithm 21. Given the constrained state–space $\mathcal{Z} = \mathcal{X} \times \mathcal{X}^1 \times \cdots \times \mathcal{X}^{m-1}$, equilibrium space $\mathcal{Z}_0 = \{(\boldsymbol{x}, \mathbf{0}, \ldots, \mathbf{0}) : \boldsymbol{x} \in \mathcal{X}\}$, the obstacle set $\Theta := \mathcal{O} \times \mathcal{X}^1 \times \cdots \times \mathcal{X}^{m-1}$, the target set $\{\boldsymbol{\tau}\}$, and secondary objective function (2) execute the following steps:

(*i*) Initialize the set $\widehat{\mathcal{W}}_0 = \{\boldsymbol{\tau}\}$ and cost-to-go $s_0 = 0$.

(*ii*) For $k = 1, 2, \ldots$update $\widehat{\mathcal{W}}_k$: $\widehat{\mathcal{W}}_k = \hat{\mathcal{R}}_-(\widehat{\mathcal{W}}_{k-1} - \Theta)$. To each $\boldsymbol{z} \in \widehat{\mathcal{W}}_k - \widehat{\mathcal{W}}_{k-1}$ assign the *next* discrete optimal input $\boldsymbol{u}^* \in \widehat{\mathcal{U}}$ according to

$$\boldsymbol{u}^*(\boldsymbol{z}) = \arg \min_{\boldsymbol{u} \in \mathcal{C}_+(\boldsymbol{z}, \widehat{\mathcal{W}}_{k-1})} \psi(\boldsymbol{z}, \boldsymbol{u}) + s_{k-1} \circ \boldsymbol{\varphi}_+(\boldsymbol{z}, \boldsymbol{u})$$

Update the kth cost-to-go function to $s_k(\boldsymbol{z}) = \psi(\boldsymbol{z}, \boldsymbol{u}^*) + s_{k-1} \circ \boldsymbol{\varphi}_+(\boldsymbol{z}, \boldsymbol{u}^*)$, $\forall \boldsymbol{z} \in \widehat{\mathcal{W}}_k - \widehat{\mathcal{W}}_{k-1}$.

(*iii*) Continue until $\widehat{\mathcal{W}}_k - \Theta$ reaches a limit: i.e., $\widehat{\mathcal{W}}_{k+1} - \Theta = \widehat{\mathcal{W}}_k - \Theta$.

The discrete version of the replanning approach presented in the preceding section complements Algorithm 21 by updating the admissible sets $\widehat{\mathcal{W}}_k$ based on the most recent obstacle information:.

Algorithm 22. Given the state–space Z, obstacle set $\Theta := O \times X^1 \times \cdots \times X^{m-1}$, output of algorithm 21, and the starting state $\boldsymbol{\sigma} \in \widehat{\mathcal{W}}_{k_{\boldsymbol{\sigma}}}$, $k_{\boldsymbol{\sigma}} \geq 1$, execute the following steps:

(*i*) Initialize the discrete population set $\widehat{\mathcal{P}}_0 = \{\boldsymbol{\tau}\}$ and cost-to-go $s_0 = 0$.

(*ii*) For $k = 1, 2, \ldots$update $\widehat{\mathcal{P}}_k$:

$$\widehat{\mathcal{P}}_k = (\widehat{\mathcal{P}}_{k-1} - \widetilde{\mathcal{P}}_{k-1}) \cup \left(\mathcal{R}_-(\widetilde{\mathcal{P}}_{k-1}) - \Theta\right)$$

where $l_k = \min_{\boldsymbol{z} \in \widehat{\mathcal{P}}_k} \ell(\boldsymbol{z})$ and $\widetilde{\mathcal{P}}_k = \left\{\boldsymbol{z} \in \widehat{\mathcal{P}}_k : \ell(\boldsymbol{z}) = l_k\right\}$. To each $z \in \widehat{\mathcal{P}}_k - \widehat{\mathcal{P}}_{k-1}$ assign the next input $u^* \in \widehat{\mathcal{U}}$ satisfying

$$\boldsymbol{u}^*(\boldsymbol{z}) = \arg \min_{\boldsymbol{u} \in \mathcal{C}_+(\boldsymbol{z}, \widehat{\mathcal{P}}_{k-1})} \psi(\boldsymbol{z}, \boldsymbol{u}) + s_{k-1} \circ \boldsymbol{\varphi}_+(\boldsymbol{z}, \boldsymbol{u})$$

and update $s_k(\boldsymbol{z}) = \psi(\boldsymbol{z}, u^*) + s_{k-1} \circ \varphi_+(\boldsymbol{z}, u^*)$, $\forall \boldsymbol{z} \in \widehat{\mathcal{P}}_k - \widehat{\mathcal{P}}_{k-1}$.

(*iii*) Continue until $\boldsymbol{\sigma} \in \widehat{\mathcal{P}}_k$. Upon termination update the global optimal input and cost-to-go for the states visited during the search if desired.

Remark 23. The sequential length ℓ of $\boldsymbol{z} \in \widehat{\mathcal{P}}_k$ is the sum of the length of the sequence in $\bigcup_{j \leq k} \widehat{\mathcal{P}}_j$ from $\boldsymbol{\tau}$ to $\boldsymbol{z}$ plus the minimum number of steps required to transfer $\boldsymbol{\sigma}$ to $\boldsymbol{z}$ without any obstacles. Both quantities are known at each step of the algorithm provided the obstacle free expansion steps (used in lieu of heuristics) from $\boldsymbol{\sigma}$ to $\boldsymbol{\tau}$ had been precomputed and stored. The replanning

https://doi.org/10.1017/S026357471500106X Published online by Cambridge University Press

information obtained by Algorithm 22 can be used to update the global navigation map and/or be updated by future local searches. As in the continuous case, the algorithm terminates in at most $k_{\max} = \ell(\boldsymbol{\tau}) + \frac{1}{2}(\ell(\boldsymbol{\sigma}) - \ell(\boldsymbol{\tau}))(\ell(\boldsymbol{\sigma}) + \ell(\boldsymbol{\tau}) - 3)$ iterations. The total number of node expansions is the product of $k_{\max}$ and the average number of states with equal sequential lengths, which could be quiet low depending on the obstacle topology.

The discrete counterpart of the operation domain $\mathcal{D}$ and admissible set $\mathcal{A}$ are defined next.

Definition 24. *For $\varepsilon > 0$, let the sampling period T, positive integer $\hat{K}$, and the maximal set $\hat{\mathcal{D}} = \hat{\mathcal{R}}_{-}^{\hat{K}}(\boldsymbol{\tau})$ be as specified in Theorem 20. Define $\widehat{\mathcal{A}} \subset \hat{\mathcal{D}}$, referred to as the discrete admissible set, to be a subset of $\mathcal{Z}$ such that $\hat{\boldsymbol{z}} \in \widehat{\mathcal{A}}$ can reach the target $\boldsymbol{\tau}$ in a finite number of steps ($\leq \hat{K}$) while remaining in $\mathcal{Z} - \Theta$.*

The following is the discrete counterpart of the continuous convergence theorem (Theorem 13):

Theorem 25. *Suppose that for $\varepsilon > 0$ the sampling period T is chosen in accordance with Theorem 20. Then, Algorithm 21 terminates in a finite number of steps depending on ε and $\mathcal{Z}$. Upon termination, it identifies the discrete spline trajectory that connects each $\hat{\boldsymbol{z}} \in \widehat{\mathcal{A}}$ to the target $\boldsymbol{\tau}$ in the fewest number of steps while minimizing the secondary objective function (2). Furthermore, Algorithm 22 used for motion replanning from a given start state $\boldsymbol{\sigma} \in \widehat{\mathcal{W}}_{k_\sigma}$, $k_{\boldsymbol{\sigma}} \geq 1$, terminates in at most $k_{\max} = \ell(\boldsymbol{\tau}) + \frac{1}{2}(\ell(\boldsymbol{\sigma}) - \ell(\boldsymbol{\tau}))(\ell(\boldsymbol{\sigma}) + \ell(\boldsymbol{\tau}) - 3)$ iterations after recomputing the optimal trajectory from $\boldsymbol{\sigma}$ to $\boldsymbol{\tau}$.*

Proof. Let $\hat{K}$ and $\hat{\mathcal{D}}$ be as specified in Theorem 20. The sets $\widehat{\mathcal{W}}_k$ are finite, non-decreasing (i.e., $\widehat{\mathcal{W}}_k \subset \widehat{\mathcal{W}}_{k+1}$), and subset of the finite set $\hat{\mathcal{D}}$ for all $k \geq 1$. Therefore, there exists a finite set $\bar{K}$ such $\widehat{\mathcal{W}}_k = \widehat{\mathcal{W}}_{\bar{K}}$, $\forall k \geq \bar{K}$. We claim that $\widehat{\mathcal{W}}_{\bar{K}} = \widehat{\mathcal{A}}$. It is clear that $\widehat{\mathcal{W}}_{\bar{K}} \subset \widehat{\mathcal{A}}$. To prove the converse, let $\hat{\boldsymbol{z}} \in \widehat{\mathcal{A}}$. Then by the definition of $\widehat{\mathcal{A}}$, there exist an integer $K \leq \bar{K}$ and input sequence $u_0, u_1, \ldots, u_{K-1} \in \widehat{\mathcal{U}}$ such that $z_{k+1} = \boldsymbol{\varphi}_{-}(z_k, \boldsymbol{u}_k)$, $z_0 = \boldsymbol{\tau}$, $z_K = \hat{\boldsymbol{z}}$, and $z_k \in \mathcal{Z} - \Theta$ for $k = 1, \ldots, K$. It can be seen that $z_k \in \hat{\mathcal{R}}_{-}(z_{k-1}) - \Theta$, which implies $z_k \in \widehat{\mathcal{W}}_k$, $k = 1, \ldots, K$, and $z_K = \hat{\boldsymbol{z}} \in \widehat{\mathcal{W}}_K \subset \widehat{\mathcal{W}}_{\bar{K}}$. Consequently, the algorithm terminates at $k = \bar{K}$ with $\widehat{\mathcal{W}}_{\bar{K}} = \widehat{\mathcal{A}}$. The secondary optimization in the discrete case is reduced to minimizing a finite set at each step k of the algorithm since $\widehat{\mathcal{W}}_k$ and $\mathcal{C}_{+}(\boldsymbol{z}, \widehat{\mathcal{W}}_{k-1})$ are finite sets of discrete points. The proof of the assertions made regarding Algorithm 22 is completely parallel to the continuous case (i.e., Theorem) and is omitted. □

Remark 26. The theorem shows that, similarly to its continuous counterpart, the discrete algorithm terminates in a finite number of steps depending on the size of the domain $\mathcal{X}$ and the complexity of the obstacles but not necessarily the number of obstacles. The computational complexity of the algorithm is directly related to the number of admissible states (i.e., states that can reach the target in a finite number of states). This number clearly is maximum in the absence of any obstacles (i.e., free space motion) and can only decrease as the number of obstacles increases. Furthermore, the efficiency of the algorithm can be further improved by updating the optimal trajectories using the replanning Algorithm 22.

7. Numerical Case Study

The discrete version of the proposed optimal control strategy is applied to the navigation of an arbitrary shaped planar mobile robot with translational and rotational degrees of freedom (x, y, θ) through a set of arbitrary obstacles. The robot dynamic model for the purpose of trajectory planning is $\ddot{x} = u_x$, $\ddot{y} = u_y$, and $\ddot{\theta} = u_\theta$, where u_x, u_y, and u_θ denote the acceleration inputs in the x, y, and θ directions, respectively. The domain of operation of the robot is a rectangular prism $\mathcal{X} = (-x_{\max}, x_{\max}) \times (-y_{\max}, y_{\max}) \times (-\pi, \pi)$ with $x_{\max} = y_{\max} = 5$ m. Additionally, the linear motion of the robot is subject to both velocity and acceleration limits ($m = 2$) of $v_{\max} = 3$ m/s and $a_{\max} = 5$ m/s^2, respectively, in both the x and y directions. The angular velocity and acceleration limits are $\omega_{\max} = 2\pi$ rad/s and $\alpha_{\max} = 5\pi$ rad/s, respectively. The spline trajectory time step-size of $T = 0.2$ s (typically much larger than the control sampling period) was used in all the numerical studies resulting in a mesh grid of $99 \times 99 \times 9 \times 7 \times 7 \times 3$ or a total of 129, 66, 723 possible discrete states. For simplicity, the origin $\tau = (0, 0, \theta_f)$ for an arbitrary $\theta_f \in (-\pi, \pi)$ is selected as the target position. The function $\psi(\boldsymbol{x}, \dot{\boldsymbol{x}}, \boldsymbol{u})$ of the secondary objective function (2) is chosen to minimize the input magnitude as well

https://doi.org/10.1017/S026357471500106X Published online by Cambridge University Press

Table I. Summary of the computational complexity of the optimization algorithms.

Search method	No. of obstacles	Mean no. of steps	No. of runs	Mean no. of states processed	Mean CPU time (s)
Global DP	0	16	1	16,45,736	309
Global DP	30	29	40	5,70,170	304
Global DP	50	32	40	4,83,220	249
Local DP	30	22	100	1357	2.7
Local DP	50	25	100	1450	4.5
A*-Type	30	4317	100	6749	28.2
A*-Type	50	4527	100	5381	26.2

as the robot *angle of attack* (angle between heading and velocity vectors):

$$\psi(\boldsymbol{x}, \dot{\boldsymbol{x}}, \boldsymbol{u}) = |u_1| + |u_2| + |\theta - \text{atan2}(\dot{y}, \dot{x})|$$

where 'atan2' is the 4–quadrant arctangent.

The obstacle set $\mathcal{G}$ in $(-x_{\max}, x_{\max}) \times (-y_{\max}, y_{\max})$ is the union of randomly placed closed squares and disks varying in size from 0.5 m and 0.8 m. Let the closed set $\mathcal{F}_0 \subset (-x_{\max}, x_{\max}) \times (-y_{\max}, y_{\max})$ denote the xy–plane points on the robot when the robot is at the origin and for $\boldsymbol{x} = (x, y, \theta)$ define $\mathcal{F}_{\boldsymbol{x}} = (x, y) + R_\theta \mathcal{F}_0$ where $R_\theta = \left[\begin{smallmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{smallmatrix}\right]$ is the rotation matrix. The three–dimensional obstacle set $\mathcal{O}$ in $\mathcal{X}$ may be expressed by $\mathcal{O} = \{\boldsymbol{x} \in \mathcal{X} : \mathcal{F}_{\boldsymbol{x}} \cap \mathcal{G} \neq \varnothing\}$, which can be shown to be a closed set. All the numerical studies to follow were performed on a 3.40 GHz Intel Core i7 desktop running Matlab R2013a. The discrete time–optimal trajectory planning and replanning algorithms (Algorithms 21 and 22) were evaluated for 30–50 randomly placed obstacles starting from near the four corners of X. Figures 1(a) and 2 show the optimal path and the time history of the velocities $\dot{x}$ and $\dot{y}$, respectively, for the trajectories starting at the top left and bottom right corners. As can be seen, each of the four trajectories avoid the obstacles and arrive at the destination along a smooth path while both velocities stay within their limits of ± 3 m/s. To demonstrate the benefits of the secondary objective function, the optimal path was also generated without using the secondary objective functions. As can be seen in Fig. 1(b) the resulting paths, particularly the robot orientation, are not as smooth as those in Fig. 1(a) illustrating the importance of the secondary objective function.

To evaluate its performance, Algorithm 22 was used to replan the optimal trajectories for randomly varying obstacles. Figure 3(a) and 3(b) demonstrate that the replanning algorithm successfully updated the trajectories for both expanding and shrinking obstacle sizes. In particular, Fig. 3(a) illustrates a case where the original trajectories both cross the obstacles whereas the updated ones detour them without collision. Table 1 provides a comparison of the two algorithms presented in this paper with an A*–type search algorithm. It was not feasible for us to implement any pure graph search algorithm such as Dijkstra or A* because of the shear size of the graph ($129,66,723$ nodes and 1.75×10^{18} edges). Instead we essentially implemented a forward search version of the main DP algorithm with the path length minimization replacing the number of steps as the primary optimization criterion. The Euclidean distance to the target was used as the Heuristic function for the A*–type search. First, the results confirm our claim that increasing the number of obstacles does not necessarily increase the algorithm complexity. In fact, they show the opposite for the global DP algorithm: A decrease in the mean CPU time (averaged over 40 runs) as the number is increased from 30 to 50 obstacles. The results show that the replanning algorithm processes far fewer states (i.e., fewer node expansions) than the global DP and even the A*–type search. The mean CPU times for the replanning algorithms (averaged over 100 runs) are significantly lower that those of the global and the A*–type search algorithms. In spite of a higher computational time, the global DP algorithm has the advantage of computing all the optimal trajectories reaching the goal. Finally, we would like to point out that the CPU times listed in Table 1 are for comparison purposes only as possibly much lower times may be obtained by coding the algorithms using a compiled as opposed to an interpreted programming language like Matlab.

https://doi.org/10.1017/S026357471500106X Published online by Cambridge University Press

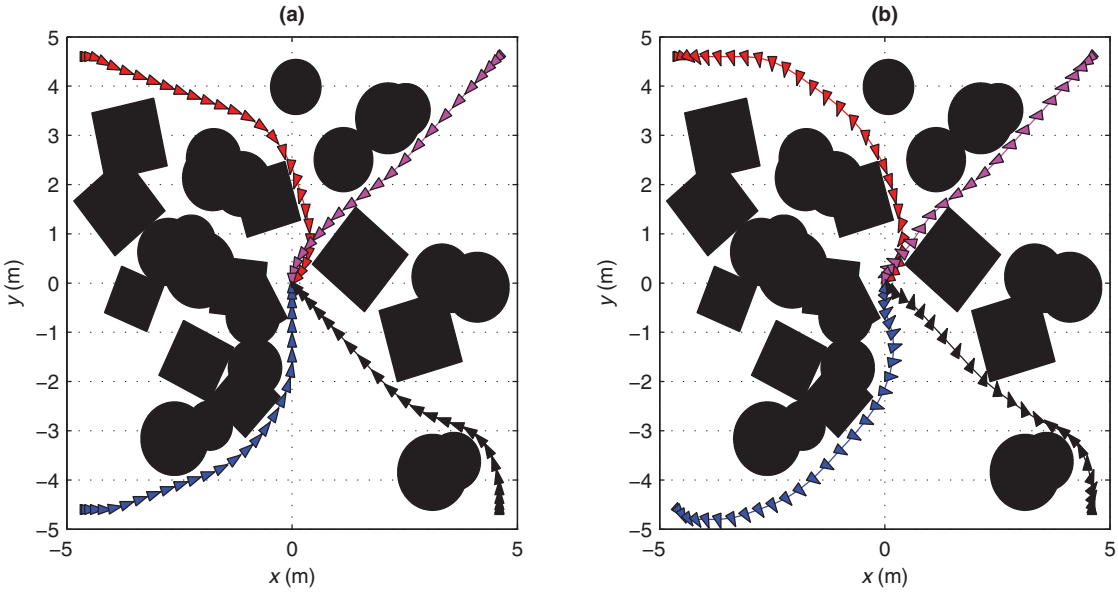


Fig. 1. (a) Time optimal paths for 30 obstacles, (b) Time optimal paths without secondary objective function.

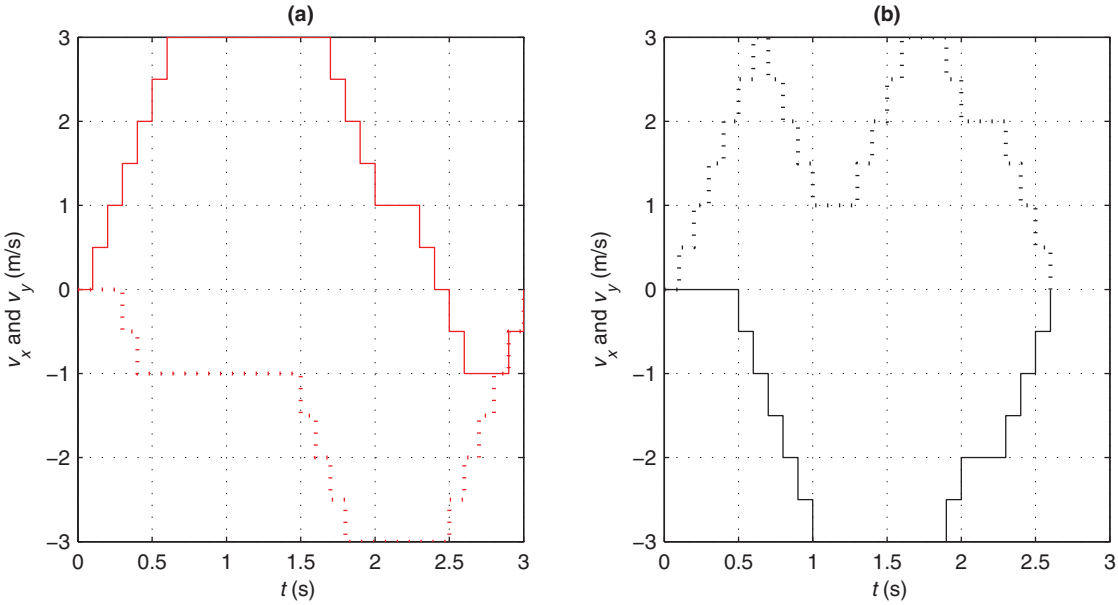


Fig. 2. Time history of $\dot{\boldsymbol{x}}(t)$ (solid lines) and $\dot{\boldsymbol{y}}(t)$ (dotted lines) for optimal trajectories starting at (a). The top left corner and (b). The bottom right corner.

8. Conclusions

This paper presented several motion planning algorithms applicable to both stationary (fixed–based) and mobile robots in the presence of both static and changing obstacles. The main algorithm presented in the paper generates a global navigation map based on DP, which enables a robot to reach its destination through an arbitrary obstacle field in a minimum number of time–steps while minimizing a secondary objective function such as its path length or the required control effort. The resulting path is a smooth spline incorporating the motion state and input constraints such as velocity, acceleration, and jerk limits. Even though in principal the continuous search algorithm may be implemented using elementary set operations, a discrete version of the continuous algorithm was formulated by discretizing the input space and reducing it to a finite tertiary set $\{0, \pm 1\}$ per input. It was shown that it is possible to maintain a prescribed level of planning accuracy in spite of such a significant reduction in

https://doi.org/10.1017/S026357471500106X Published online by Cambridge University Press

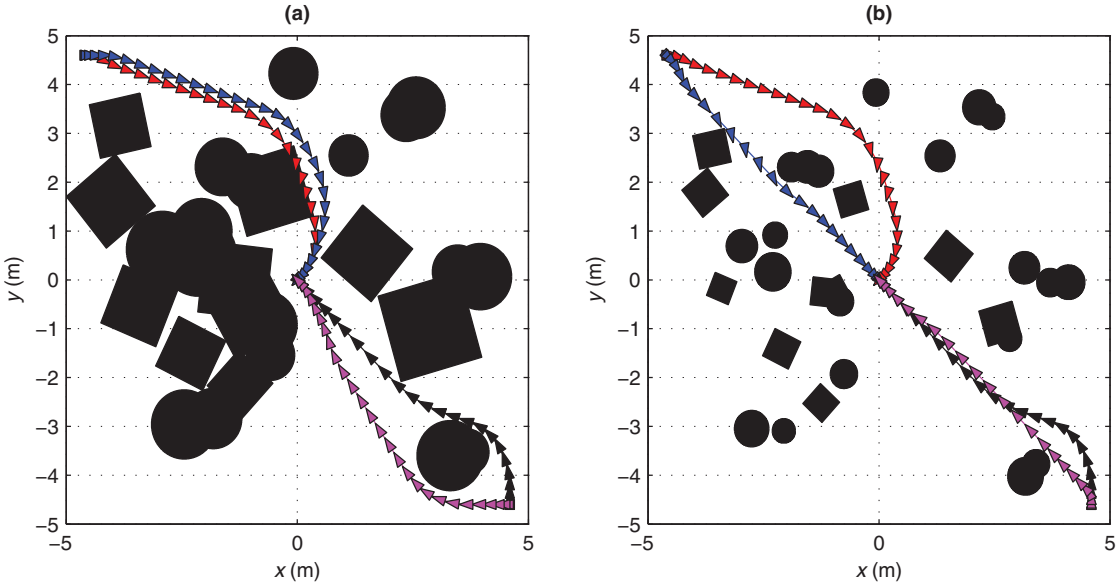


Fig. 3. The initial and updated optimal trajectories after replanning for (a). Random obstacle changes, and (b). Reduction in obstacle sizes.

the input and search spaces. To accommodate a dynamic obstacle field, a novel replanning algorithm was proposed to update the original optimal trajectory of a robot dynamically while it is searching for its destination. Each algorithm was shown to terminate in a finite number of steps with its overall computational complexity being independent of the type and number of obstacles. In particular, the replanning algorithm was proven to terminate in only a few iterations depending on the number of time steps separating the starting state from the target. The effectiveness of the discretized algorithms were demonstrated on a planar mobile robot with three degrees of freedom subject to velocity and acceleration constraints. The numerical studies confirmed that the complexity of global DP algorithm does not increase with the number of obstacles. They also revealed that the replanning algorithm is the most efficient one among all the ones tested in recomputing the optimal trajectories. Aside from enhancing the computational efficiency of the global and replanning algorithms, an important future direction of the present work would be its generalization to optimal motion planning and control of non-linear systems including underactuated/non-holonomic robots.

References

1. L. Blackmore, H. Li and B. Williams, "A Probabilistic Approach to Optimal Robust Path Planning with Obstacles," *Proceedings of the 2006 American Control Conference* (2006) pp. 2831–37.
2. J. Peterson, "Obstacle Avoidance using Hierarchical Dynamic Programming," *Proceedings of The 23rd Southeastern Symposium on System* (1991) pp. 192–6.
3. E. Dijkstra, "A note on two problems in connexion with graphs," *Numer. Math.* **1**(1), 269–271 (1959).
4. P. Hart, N. Nilsson and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.* **4**(2), 100–107 (1968).
5. R. Wein, J. van den Berg and D. Halperin, "The visibility–voronoi complex and its applications," *Comput. Geom.* **36**, 66–87 (2007).
6. D. D. Dunlap, C. V. Caldwell, J. E. G. Collins and O. Chuy, "Motion Planning for Mobile Robots Via Sampling-Based Model Predictive Optimization," Recent Advances in Mobile Robotics, vol. 1, Dr. Andon Topalov (Ed.), (InTech, 2011), ISBN: 978-953-307-909-7.
7. T. M. Howard, M. Pivtoraiko, R. Knepper, A. Kelly, et al., "Model-predictive motion planning: Several key developments for autonomous mobile robots," *Robot. Autom. Mag. IEEE* **21**(1), 64–73 (2014).
8. S. Koenig and M. Likhachev, "Fast replanning for navigation in unknown terrain,"*IEEE Trans. Robot.* **21**(3), 354–363 (2005).
9. E. Rimon and D. Koditschek, "Exact robot navigation using artificial potential functions," *IEEE Trans. Robot. Autom.* **8**(5), 501–18 (1992).
10. M. Eichhorn, "A Reactive Obstacle Avoidance System for an Autonomous Underwater Vehicle," *Proceedings of the IFAC World Congress* (2005) p. CDRom.

https://doi.org/10.1017/S026357471500106X Published online by Cambridge University Press

11. D. Bruijnen, J. van Helvoort and R. van de Molengraft, "Realtime Motion Path Generation using Subtargets in a Changing Enviornment," *Proceedings of the 2006 American Control Conference* (2006) pp. 4243–48.
12. J.-Y. Zhang, Z.-P. Zhao and D. Liu, "A path planning method for mobile robot based on artificial potential field," *Adv. Robot.* **38**(8), 1306–9 (2006).
13. B. Cetin, M. Bikdash and F. Hadaegh, "Hybrid mixed-logical linear programming algorithm for collision–free optimal path planning," *IET Control Theory Appl.* **1**(2), 522–31 (2007).
14. C. Ma and R. Miller, "Milp Optimal Path Planning for Real-Time Applications," *Proceedings of the 2006 American Control Conference* (2006) p. 6.
15. E. Balas, "Disjunctive programming," *Ann. Discrete Math.* **5**, 3–51 (1979).
16. R. Bellman, *Dynamic Programming* (Dover Publications, Inc., Mineola, NY, 2003).
17. D. Yershov and S. LaValle, "Simplicial dijkstra and A* algorithms: From graphs to continuous spaces," *J. Harbin Inst. Technol.* **26**(17), 2065–2085 (2012).
18. N. Sadegh, "Time–Optimal Motion Planning of Autonomous Vehicles in the Presence of Obstacles," *Proceedings of the 2008 American Control Conference* (2008) p. 6.
19. P. Nagabhushan and P. Manohara, "Cognition of free space for planning the shortest path: A framed free space approach," *Pattern Recognit. Lett.* **22**(9), 971–82 (2001).
20. B. Bakker, Z. Zivkovic and B. Krose, "Hierarchical Dynamic Programming for Robot Path Planning," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems* (2005) pp. 2756–61.
21. A. Meystel and A. Gues, H. G., "Minimum Time Path Planning for a Robot," *Proceedings of the 1986 IEEE International Conference on Robotics and Automation* (1986) pp. 1678–87.
22. B. Chazelle, "Convex partitions of polyhedra: A lower bound and worst-case optimal algorithm," *SIAM J. Comput.* **13**(3), 488–507 (1984).
23. M. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming* (Wiley–Interscience, 2005).
24. H. Royden, *Real Analysis* (Macmillan Publishing Co., Inc., New York, NY, 1968).

Appendix A. Technical Proofs

A.1. Proof of Proposition 1

Letting $\tau = t - t_k$ then from $x_i^m(t) = u_i(t)$ it can be seen that $\boldsymbol{x}(t) - \boldsymbol{x}_k = \sum_{j=1}^m \frac{\tau^j}{j!}\boldsymbol{x}_k^j$, where $\boldsymbol{x}_k^j = (d^j\boldsymbol{x}/dt^j)_{t=t_k}$. Letting $\boldsymbol{\sigma}$ be as specified in the Proposition, then it follows that $\|\boldsymbol{x}(t) - \boldsymbol{x}_k\| \le \sum_{j=1}^m \frac{T^j\boldsymbol{\sigma}^j}{2^j j!} \le e^{T\boldsymbol{\sigma}/2} - 1$ for all $t_k - T/2 \le t \le t_k + T/2$, $\forall k \ge 1$. Choosing $e^{T\boldsymbol{\sigma}/2} - 1 < \varepsilon$ or equivalently $T < 2\ln(1+\varepsilon)/\boldsymbol{\sigma}$ implies that $\sup_{t\in[t_k - T/2, t_k + T/2]} \|\boldsymbol{x}(t) - \boldsymbol{x}_k\| < \varepsilon$. Thus $\boldsymbol{x}(t) \in \boldsymbol{x}_k + B_\varepsilon$ and $\boldsymbol{x}(t) \notin \mathcal{H}$, $\forall t \in [t_k - T/2, t_k + T/2]$.

A.2. Proof of Lemma 2

Let $\boldsymbol{z} = (\boldsymbol{x}, \boldsymbol{0}, \ldots, \boldsymbol{0})$ and $\boldsymbol{z}' = (\boldsymbol{x}', \boldsymbol{0}, \ldots, \boldsymbol{0})$. Since $\mathcal{X}$ is convex, the line joining $\boldsymbol{x} = (x_1, \ldots, x_n)$ to $\boldsymbol{x}' = (x_1', \ldots, x_n')$ lies in $\mathcal{X}$: $\boldsymbol{x}_\lambda = (1-\lambda)\boldsymbol{x} + \lambda\boldsymbol{x}' \in \mathcal{X}$, $\lambda \in [0, 1]$. We first construct a scalar function $\lambda : [0, T_m][0, 1]$ for some integer multiple T_m of the sampling time T such that $\lambda(0) = 0$, $\lambda(t) = 1$, $t \ge T_m$ and $d^m\lambda/dt^m$ is a piecewise constant 2-value function. To this end, define the sequence of time functions, $f_j(t)$, $j = 1, \ldots, m$ in the following way: $f_1(t) = \begin{cases} 1/T_0, & 0 \le t < T_0 \\ 0, & \text{otherwise} \end{cases}$. and $f_{j+1}(t) = [f_j(t) - f_j(t - T_j)]/T_j$ with $T_j = 2^{j-1}T_0$, $j = 1, \ldots, m$, and T_0 an integer multiple of T. It can be easily verified that $f_j(t)$ is piecewise constant (i.e., constant on $[kT, (k+1)T]$) with values at $\pm 2^{-(j-1)(j-2)/2}T_0^{-j}$, $j \ge 3$, and vanishes outside of $[0, T_j)$. It remains to show that the solution of $\lambda^m := d^m\lambda/dt^m = f_m$ subject to zero initial conditions has the desired properties. Defining the sequence, $g_j(t)$, $j = 1, \ldots, m$, by $g_{j+1}(t) = \frac{1}{T_j}\int_{t-T_j}^t g_j(\tau)d\tau$, $g_1(t) = f_1(t)$ then it follows that that $\dot\lambda(t) = g_m(t)$. It can be seen that each function $g_j(t)$, $j \ge 2$, is $j-1$ times differentiable, and similar to $f_j(t)$, vanishes outside of $[0, T_j)$. Furthermore, since $g_1(t) = f_1(t) \ge 0$, it follows that $g_j(t) \ge 0$, $\forall t \ge 0$. Thus $\dot\lambda(t) \ge 0$, $\forall t < T_m$ and $\dot\lambda(t) = 0$, $\forall t \ge T_m$. This combined with that $\dot\lambda(t)$ is $m-1$ times differentiable imply $\lambda^j(t) = d^j\lambda/dt^j(t) = 0$, $t \ge T_m$, $j = 1, \ldots, m$.

To prove that $\lambda(mT) = 1$ and establish the bounds on $\lambda^j(t)$, we make use of the Laplace transform. The Laplace transform of $h_j(t) = \int_0^t g_j(\tau)d\tau$ can be shown to be $\hat h_j(s) = \frac{1}{s}\prod_{k=0}^{j-1}(\frac{1-e^{-sT_k}}{T_k s})$. By the Final Value Theorem, $\lim_{t\to\infty} h_j(t) = \lim_{t\to\infty} s\hat h_j(t) = 1$. Since $\dot h_j = g_j$ vanishes outside of $[0, T_j)$, $h_j(T_j) = \lim_{t\to\infty} h_j(t) = 1$. To evaluate $\lambda^j(t)$, $0 \le j < m-1$, we first note that in the Laplace domain $\widehat{\lambda^j}(s) = \hat h_{m-j}(s)\prod_{k=m-j}^{m-1}\frac{1-e^{-sT_k}}{T_k}$. Defining $\tilde h_0 = h_{m-j}$, $\tilde h_{k+1}(t) = \tilde h_k(t) - \tilde h_k(t - 2^kT_{m-j})$,

https://doi.org/10.1017/S026357471500106X Published online by Cambridge University Press

$k = 0, \ldots, j-1$, the preceding relation is equivalent to $\lambda^j(t) = \tilde{h}_j(t)(\prod_{k=m-j}^{m-1} T_k^{-1})$. Since $\dot{h}_{m-j}(t)$ vanishes outside of $[0, T_{m-j}]$, it follows that $\lambda^j(t) = h_{m-j}(t)(\prod_{k=m-j}^{m-1} T_k^{-1}) \geq 0$, $\forall t \in [0, T_{m-j}]$ and $\lambda^j(t) = \pm\, \lambda^j(\tau)$ for some $\tau \in [0, T_{m-j}]$, $\forall t \in [T_{m-j}, T_m]$. Thus

$$\sup_{t\in[0,T_m]} \left|\lambda^j(t)\right| = \lambda^j(T_{m-j}) = 2^{-j(2m-j-3)/2} T_0^{-j} \tag{A.1}$$

and in particular $\lambda(T_m) = 1$ and $0 \leq \lambda(t) \leq 1$, $\forall t \in [0, T_m]$.

To construct the actual solution, let $\rho_i : \overline{\mathcal{X}} \to \mathbb{R}$ be the projection map that maps $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathcal{X}$ to its ith component x_i: $\rho_i(\boldsymbol{x}) = x_i$. Since $\mathcal{X}$ is compact and convex, so is its image under the projection $\rho_i(\mathcal{X})$ implying that each $\rho_i(\mathcal{X})$ is a compact interval $[x_{i,\min}, x_{i,\max}]$. Let $L_i := |x_{i,\max} - x_{i,\min}|$ and choose integer P such that

$$P > \sup_{i,j} 2^{-\kappa_j} T^{-1} \left(L_i / x_{i,\max}^j\right)^{1/j}$$

where $\kappa_j = \frac{2m-j-3}{2}$, $1 \leq j < m$, and $\kappa_m = \frac{(m-1)(m-2)}{2m}$. Setting $T_0 = PT$ ensures that $\boldsymbol{x}(t) = \boldsymbol{x} + (\boldsymbol{x}' - \boldsymbol{x})\lambda(t) \in \mathcal{X}$, $t \in [0, T_m]$, is the desired spline that transfers $\boldsymbol{x}$ to $\boldsymbol{x}'$. First, the input function corresponding to this spline, $\boldsymbol{u}(t) = d^m\boldsymbol{x}(t)/dt^m = f_m(t)(\boldsymbol{x}' - \boldsymbol{x})$ is piecewise constant function such that $f_m(t) = \pm 2^{-(m-1)(m-2)/2} P^{-m} T^{-m}$ and $|u_i(t)| < x_{i,\max}^m$. Moreover,

$$\sup_{t\in[0,Tm]} \left|x_i^j(t)\right| \leq L_i \sup_{t\in[0,T_m]} \left|\lambda^j(t)\right| < x_{i,\max}^j$$

This completes the proof.

A.3. Proof of Proposition 4

To prove part i, let $\boldsymbol{\varphi}_\pm^k : \mathbb{R}^{mn} \times \mathbb{R}^{kn} \to \mathbb{R}^{mn}$ be defined by $\boldsymbol{\varphi}_\pm^k(\boldsymbol{z}, \boldsymbol{u}_0, \ldots, \boldsymbol{u}_{k-1}) = \bar{\mathbf{A}}_{\pm T}^k \boldsymbol{z} + \sum_{p=0}^{k-1} \bar{\mathbf{A}}_{\pm T}^{k-p-1} \bar{\mathbf{B}}_{\pm T} \boldsymbol{u}_p$, where $\bar{\mathbf{A}}_{\pm T} = \mathbf{A}_{\pm T} \otimes I_{n\times n}$ and $\bar{\mathbf{B}}_{\pm T} = \mathbf{B}_{\pm T} \otimes I_{n\times n}$. It is well-known that $(\mathbf{A}_\tau, \mathbf{B}_\tau)$ in (3) is controllable for any τ and m, i.e., $\operatorname{rank}([\, \mathbf{A}_\tau^{k-1}\mathbf{B}_\tau \cdots \mathbf{B}_\tau\,]) = m$, $\forall k \geq m$. This is equivalent to the controllability of $(\bar{\mathbf{A}}_\tau, \bar{\mathbf{B}}_\tau)$ since $\bar{\mathbf{A}}_\tau^p \bar{\mathbf{B}}_\tau = \mathbf{A}_\tau^p \mathbf{B}_\tau \otimes I_{n\times n}$ and

$$\operatorname{rank}\left(\left[\, \bar{\mathbf{A}}_\tau^{k-1}\mathbf{B}_\tau \cdots \bar{\mathbf{B}}_\tau\,\right]\right) = mn, \forall k \geq m$$

implying that $\boldsymbol{\varphi}_\pm^k(\boldsymbol{\tau}, \boldsymbol{u}_0, \ldots, \boldsymbol{u}_{k-1}) = \boldsymbol{\tau} + \sum_{p=0}^{k-1} \bar{\mathbf{A}}_{\pm T}^{k-p-1} \bar{\mathbf{B}}_{\pm T} \boldsymbol{u}_p$ maps $\mathbb{R}^{kn}$ *onto* $\mathbb{R}^{mn}$, $\forall k \geq m$. Therefore $\boldsymbol{\varphi}_\pm^k(\boldsymbol{\tau}, .)$, is a bounded linear open–mapping,[24] which maps open sets to open sets. Let

$$\mathcal{U}_p' = \left\{(\boldsymbol{u}_0, \ldots, \boldsymbol{u}_{k-1}) : \boldsymbol{\varphi}_\pm^p(\boldsymbol{\tau}, \boldsymbol{u}_0, \ldots, \boldsymbol{u}_{p-1}) \subset \mathcal{Z}\right\}.$$

Then by the boundedness of $\boldsymbol{\varphi}_\pm^p(\boldsymbol{\tau}, .)$ for each finite p, the set $\mathcal{U}_p'$ is an open subset of $\mathbb{R}^{kn}$. Defining the open set $\mathcal{U}_k := (\bigcap_{p\leq k} \mathcal{U}_p') \cap \mathcal{U}$, it follows that $\mathcal{R}_\pm^k(\boldsymbol{\tau}) = \{\boldsymbol{z} \in \mathbb{R}^{mn} : \boldsymbol{z} = \boldsymbol{\varphi}_\pm^k(\boldsymbol{\tau}, \boldsymbol{u}),\ \boldsymbol{u} \in \mathcal{U}_k\}$. Therefore, by the open–mapping property of $\boldsymbol{\varphi}_\pm^k(\boldsymbol{\tau}, .)$, $\mathcal{R}_\pm^k(\boldsymbol{\tau})$ is an open subset of $\mathcal{Z}$, which completes the proof. To prove the non-decreasing property let $\boldsymbol{z} \in \mathcal{R}_\pm^k(\boldsymbol{\tau})$. Then there exists $(\boldsymbol{u}_0, \ldots, \boldsymbol{u}_{k-1}) \in \mathcal{U}_k$ such that $\boldsymbol{z} = \boldsymbol{\varphi}_\pm^k(\boldsymbol{\tau}, \boldsymbol{u}_0, \ldots, \boldsymbol{u}_{k-1})$. Clearly, $\boldsymbol{z} = \boldsymbol{\varphi}_\pm^k(\boldsymbol{\tau}, 0, \boldsymbol{u}_0, \ldots, \boldsymbol{u}_{k-1}) \in \mathcal{R}_\pm^{k+1}(\boldsymbol{\tau})$ since

$$\boldsymbol{z} = \boldsymbol{\varphi}_\pm(\boldsymbol{\tau}, 0) = \boldsymbol{\tau}.$$

To prove part (ii), let $\boldsymbol{w} \in \mathcal{Z}_0$ be given. By Lemma (2), there exists an integer K such that $\boldsymbol{w} \in \mathcal{R}_+^K(\boldsymbol{\tau})$. By the first part of the theorem, $\mathcal{R}_+^K(\boldsymbol{\tau}) \subset \mathcal{R}_+^k(\boldsymbol{\tau})$, $\forall k \geq K$ proving that $\mathcal{Z}_0 \subset \mathcal{R}_+^k(\boldsymbol{\tau})$, $\forall k \geq K$. The proof of $\mathcal{Z}_0 \subset \mathcal{R}_-^k(\boldsymbol{\tau})$ follows from the preceding proof by switching $\boldsymbol{w}$ and $\boldsymbol{\tau}$.

We first prove part (iii) for $k = 1, 2$: $\mathcal{R}_\pm(\mathcal{W}_1 \cap \mathcal{W}_2) \subset \mathcal{R}_\pm(\mathcal{W}_1) \cap \mathcal{R}_\pm(\mathcal{W}_2)$. Let $\boldsymbol{z} \in \mathcal{R}_\pm(\mathcal{W}_1 \cap \mathcal{W}_2)$. The there exists $\boldsymbol{y} \in \mathcal{W}_1 \cap \mathcal{W}_2$ and $\boldsymbol{u} \in \mathcal{U}$ such that $\boldsymbol{z} = \boldsymbol{\varphi}_\pm(\boldsymbol{y}, \boldsymbol{u})$. Since $\boldsymbol{y} \in \mathcal{W}_1$ and $\boldsymbol{y} \in \mathcal{W}_2$, we have $\boldsymbol{z} = \boldsymbol{\varphi}_\pm(\boldsymbol{y}, \boldsymbol{u}) \in \mathcal{R}_\pm(\mathcal{W}_1) \cap \mathcal{R}_\pm(\mathcal{W}_2)$, which proves the claim for $k = 1, 2$. By

https://doi.org/10.1017/S026357471500106X Published online by Cambridge University Press

induction, $\mathcal{R}_{\pm}(\bigcap_{k=1}^{p}\mathcal{W}_k)\subset\bigcap_{k=1}^{p}\mathcal{R}_{\pm}(\mathcal{W}_k)$, $\forall p\geq 1$. To extend the proof to the infinite case, let $\mathcal{W}_{\pm}=\mathcal{R}_{\pm}(\bigcap_{k\geq 1}\mathcal{W}_k)$. Then clearly,

$$\mathcal{W}_{\pm}\subset\mathcal{R}_{\pm}\left(\bigcap_{k=1}^{p}\mathcal{W}_k\right)\subset\bigcap_{k=1}^{p}\mathcal{R}_{\pm}\left(\mathcal{W}_k\right),\ \forall p\geq 1. \tag{A.2}$$

We claim that $\mathcal{W}_{\pm}\subset\bigcap_{k\geq 1}\mathcal{R}_{\pm}(\mathcal{W}_k)$. Otherwise, there exists $\boldsymbol{z}\in\mathcal{W}_{\pm}$ such that $\boldsymbol{z}\notin\mathcal{R}_{\pm}(\mathcal{W}_p)$ for some $p\geq 1$, which violates (A.2). Thus $\mathcal{W}_{\pm}\subset\bigcap_{k\geq 1}\mathcal{R}_{\pm}(\mathcal{W}_k)$ proving part (iii). The proof of the last part is similar to that of part (iii) and is omitted.

A.4. Proof of Lemma 7

First it is clear that

$$\mathcal{R}_{-}\left(\mathcal{W}-\mathcal{B}\right)=\mathcal{R}_{-}\left(\mathcal{W}^{+}\right)-\mathcal{V}=\mathcal{R}_{-}\left(\mathcal{W}^{+}\right)\cap\mathcal{R}_{-}\left(\mathcal{W}-\mathcal{B}\right)$$

since $\mathcal{W}-\mathcal{B}\subset\mathcal{W}^{+}$ and that $\mathcal{V}$ is closed being the complement of the open set $\mathcal{R}_{-}(\mathcal{W}-\mathcal{B})$. Let $\boldsymbol{z}\in\mathcal{R}_{-}(\mathcal{W}-\mathcal{B})$. To prove the second assertion let $\boldsymbol{z}\in\mathcal{V}$ or equivalently $\boldsymbol{z}\notin\mathcal{R}_{-}(\mathcal{W}-\mathcal{B})$. Then, $\boldsymbol{\varphi}_{+}(\boldsymbol{z},\boldsymbol{u})\notin\mathcal{W}-\mathcal{B}$ or equivalently $\boldsymbol{\varphi}_{+}(\boldsymbol{z},\boldsymbol{u})\in\widetilde{\mathcal{W}-\mathcal{B}}=\mathcal{B}\cup\tilde{\mathcal{W}}$, $\forall\boldsymbol{u}\in\mathcal{U}$, implying that $\mathcal{R}_{+}(\mathcal{V})\subset\mathcal{B}\cup\tilde{\mathcal{W}}$. Taking the intersection of the preceding inclusion with $\mathcal{W}$ yields $\mathcal{R}_{+}(\mathcal{V})\cap\mathcal{W}\subset\mathcal{B}\cap\mathcal{W}\subset\mathcal{B}$, which completes the proof.

A.5. Proof of Lemma 8

First, $f^{*}(\boldsymbol{z})$ is well-defined for each $\boldsymbol{z}\in\mathcal{Z}$ as $f(z,u)$ is lower bounded. Furthermore, by the lower semicontinuity[23] [Puterman (2005), Appendix B] of f there exists $\boldsymbol{u}^{*}\in\overline{\mathcal{U}}$ such that $f^{*}(\boldsymbol{z})=f(\boldsymbol{z},\boldsymbol{u}^{*})$, which proves the last statement. To establish the semicontinuity of f^{*}, it suffices to prove that the *preimage* $f^{*-1}((a,\infty))=\{\boldsymbol{z}\in\mathcal{W}:f^{*}(z)>a\}$, $a\geq 0$ of f^{*} are open sets. Define the map $\tilde{\boldsymbol{\varphi}}_{+}:\mathcal{W}\times\overline{\mathcal{U}}\to\mathbb{R}^{mn}\times\mathbb{R}^{mn}$ to be the restriction of $(\boldsymbol{z},\boldsymbol{\varphi}_{+}(\boldsymbol{z},\boldsymbol{u}))$ to $\mathcal{W}\times\overline{\mathcal{U}}$, and let $\pi_{\mathcal{U}}$ denote the canonical projection map: $\pi_{\mathcal{U}}((\boldsymbol{z},\boldsymbol{u}))=\boldsymbol{u}$. To prove that $f^{*-1}((a,\infty))$ is open, let $\boldsymbol{z}\in f^{*-1}((a,\infty))$, or equivalently $f^{*}(\boldsymbol{z})>a$, and choose the real number b between a and $f^{*}(\boldsymbol{z})$ so that $f(\boldsymbol{z},\boldsymbol{u})>b$, $\forall\boldsymbol{u}\in\mathcal{U}_{\boldsymbol{z}}^{+}:=\mathcal{C}_{+}(\boldsymbol{z},\mathcal{W}^{+})$. By the lower semicontinuity of f, there exist open neighborhoods $\mathcal{N}_{\boldsymbol{z}}^{\boldsymbol{u}}$ and $\mathcal{N}_{\boldsymbol{u}}$ of $\boldsymbol{z}$ and $\boldsymbol{u}$, respectively, such that $\mathcal{N}_{\boldsymbol{z}}^{\boldsymbol{u}}\times\mathcal{N}_{\boldsymbol{u}}\subset f^{-1}((b,\infty))$. The open sets $\{\mathcal{N}_{\boldsymbol{u}}\}_{\boldsymbol{u}\in\mathcal{U}_{\boldsymbol{z}}^{+}}$ cover the compact set $\mathcal{U}_{\boldsymbol{z}}^{+}$. By the properties of compact sets, there exists a finite subcover $\{\boldsymbol{u}_i\}_{i=1}^{N}$ so that $\cup_{i=1}^{N}\mathcal{N}_{\boldsymbol{u}_i}\supset\mathcal{U}_{\boldsymbol{z}}^{+}$. Letting $\mathcal{N}_{\boldsymbol{z}}=\bigcap_{i=1}^{N}\mathcal{N}_{\boldsymbol{z}}^{\boldsymbol{u}_i}$ and $\mathcal{U}_{\boldsymbol{z}}=\bigcup_{i=1}^{N}\mathcal{N}_{\boldsymbol{z}}^{\boldsymbol{u}_i}$, then $\mathcal{N}_{\boldsymbol{z}}\times\mathcal{U}_{\boldsymbol{z}}\subset f^{-1}((b,\infty))$. To complete the proof, define the compact sets $\mathcal{U}_j=\pi_{\mathcal{U}}\circ\tilde{\boldsymbol{\varphi}}_{+}^{-1}(\overline{\mathcal{N}_{\boldsymbol{z}}^{j}},\mathcal{W}^{+})\subset\overline{\mathcal{U}}$ where $\mathcal{N}_{\boldsymbol{z}}^{j}=\{\boldsymbol{\xi}\in\mathcal{W}:\|\boldsymbol{\xi}-\boldsymbol{z}\|<1/j\}$, $j=1,2,\ldots$. It is clear that $\mathcal{U}_j$'s are non-increasing ($\mathcal{U}_{j+1}\subset\mathcal{U}_j$) and that $\bigcap_{j=1}^{\infty}\overline{\mathcal{N}_{\boldsymbol{z}}^{j}}=\boldsymbol{z}$. Thus $\bigcap_{j=1}^{\infty}\mathcal{U}_j=\pi_{\mathcal{U}}\circ\tilde{\boldsymbol{\varphi}}_{+}^{-1}(\boldsymbol{z},\mathcal{W}^{+})=\mathcal{U}_{\boldsymbol{z}}^{+}$. Letting $\widetilde{\mathcal{U}}_j=\mathcal{U}_j-\mathcal{U}_{\boldsymbol{z}}$, then $\bigcap_{i=1}^{N}\widetilde{\mathcal{U}}_j=\emptyset$ since $\mathcal{U}_{\boldsymbol{z}}\supset\mathcal{U}_{\boldsymbol{z}}^{+}$ for closed and bounded (hence compact) sets $\widetilde{\mathcal{U}}_j$. By the Heine–Borel properties of compact sets, there exists $J\geq 1$ such that $\forall j\geq J$, $\widetilde{\mathcal{U}}_j=\varnothing$ or $\mathcal{U}_j\subset\mathcal{U}_{\boldsymbol{z}}$. Choosing $j\geq J$ large enough so that $\mathcal{N}_{\boldsymbol{z}}^{j}\subset\mathcal{N}_{\boldsymbol{z}}$, we have

$$f^{*}(\xi)=\inf_{u\in\mathcal{C}_{+}(\xi,\overline{\mathcal{W}^{+}})}f(\xi,u)\geq\inf_{u\in\mathcal{U}_{\boldsymbol{z}}}f(\xi,u)\geq b>a$$

$\forall\xi\in\mathcal{N}_{\boldsymbol{z}}^{j}\subset\mathcal{N}_{\boldsymbol{z}}$ since $\mathcal{C}_{+}(\xi,\mathcal{W}^{+})\subset\mathcal{U}_j\subset\mathcal{U}_{\boldsymbol{z}}$. Thus $\mathcal{N}_{\boldsymbol{z}}^{j}\subset f^{*-1}((a,\infty))$, and completing the proof.

A.6. Proof of Lemma 12

To prove the first assertion, by Propositions 4(iii) and 9, we have $\mathcal{R}_{+}(\mathcal{V}_{\infty})=\mathcal{R}_{+}(\bigcap_{k\geq 1}\mathcal{V}_k)\subset\bigcap_{k\geq 1}\mathcal{R}_{+}(\mathcal{V}_k)$ and $\mathcal{R}_{+}(\mathcal{V}_{k+1})\cap\mathcal{R}_{-}^{k}(\boldsymbol{\tau})\subset\mathcal{V}_k\cup\Theta$, respectively. Since $\mathcal{R}_{-}^{k}(\boldsymbol{\tau})\subset\mathcal{R}_{-}^{k+1}(\boldsymbol{\tau})$, $\forall k\geq 1$, the last two inclusions imply $\mathcal{R}_{+}(\mathcal{V}_{\infty})\cap\mathcal{R}_{-}^{k}(\boldsymbol{\tau})\subset\mathcal{V}_{\infty}\cup\Theta$, $\forall k\geq 1$. Taking the union of both sides over all $k\geq 1$ establishes

$$\mathcal{R}_{+}\left(\mathcal{V}_{\infty}\right)\cap\mathcal{D}\subset\mathcal{V}_{\infty}\cup\Theta \tag{A.3}$$

where $\mathcal{D}=\bigcup_{k\geq 1}\mathcal{R}_{-}^{k}(\boldsymbol{\tau})$ is as previously defined.

https://doi.org/10.1017/S026357471500106X Published online by Cambridge University Press

For an arbitrary $0 \leq \varepsilon \leq 1$, let $\mathcal{R}_\varepsilon(z) := \{y \in \mathcal{D} : y = \varphi_+(z, u),\ u \in \varepsilon\mathcal{U}\}$ and define $\mathcal{R}_\varepsilon^k$ similarly to $\mathcal{R}_+^k$, i.e., $\mathcal{R}_\varepsilon^0(z) = z$ and $\mathcal{R}_\varepsilon^k = \mathcal{R}_\varepsilon^{k-1}(\mathcal{R}_\varepsilon(z))$. First it can be easily seen that $\mathcal{R}_\varepsilon(z) \subset \mathcal{R}_+(z) \cap \mathcal{D}$. Moreover, using an argument similar to that used in the proof of Proposition 4 parts (i) and (iv) (replace $\mathcal{Z}$ and $\mathcal{U}$ by the open sets $\mathcal{D}$ and $\varepsilon\mathcal{U}$, respectively), it follows that $\mathcal{R}_\varepsilon^k(z)$ is an open set for $k \geq m$ and

$$\mathcal{R}_\varepsilon(\mathcal{W}_1 \cup \mathcal{W}_2) \subset \mathcal{R}_\varepsilon(\mathcal{W}_1) \cup \mathcal{R}_\varepsilon(\mathcal{W}_2) \tag{A.4}$$

for all sets $\mathcal{W}_1, \mathcal{W}_2 \subset \mathcal{Z}$. Combining $\mathcal{R}_\varepsilon(z) \subset \mathcal{R}_+(z) \cap \mathcal{D}$ with (A.3) gives $\mathcal{R}_\varepsilon(\mathcal{V}_\infty) \subset \mathcal{V}_\infty \cup \Theta$. Applying $\mathcal{R}_\varepsilon$ to the preceding inclusion $m-1$ times (for $m > 1$) making use of (A.3) and (A.4), we have

$$\mathcal{R}_\varepsilon^m(\mathcal{V}_\infty) \subset \mathcal{V}_\infty \cup \left(\bigcup_{k=0}^{m-1} \mathcal{R}_\varepsilon^k(\Theta) \right). \tag{A.5}$$

Let $\varphi_+(z, u) = \bar{\mathbf{A}}_T z + \bar{\mathbf{B}}_T u$ be as previously defined. Using the definition of $\mathcal{R}_\varepsilon(z)$ it can be seen that

$$\mathcal{R}_\varepsilon^k(z) = \bar{\mathbf{A}}_T^k \left(z + \varepsilon \bar{\mathbf{Q}}_k \mathcal{U}_k \right) \tag{A.6}$$

where $\bar{\mathbf{A}}_T^k \bar{\mathbf{Q}}_k$ is the controllability matrix associated with $(\bar{\mathbf{A}}_T, \bar{\mathbf{B}}_T)$:

$$\bar{\mathbf{Q}}_k = \left[\bar{\mathbf{A}}_{-T} \bar{\mathbf{B}}_T \ \bar{\mathbf{A}}_{-2T} \bar{\mathbf{B}}_T \ \cdots \ \bar{\mathbf{A}}_{-kT} \bar{\mathbf{B}}_T \right] \tag{A.7}$$

and $\mathcal{U}_k$ is the open subset of $\mathcal{U} \times \cdots \times \mathcal{U}$ (k–times) such that $\mathcal{R}_+^p(z) \subset \mathcal{D}, \forall p \leq k$ (see the proof of Proposition 4(i) for a more detailed construction of $\mathcal{U}_k$). Applying the linear operator $\bar{\mathbf{A}}_T^{-m}$ to (A.5) using (A.6) we obtain

$$\mathcal{V}_\infty + \varepsilon \bar{\mathbf{Q}}_m \mathcal{U}_m \subset \bar{\mathbf{A}}_T^{-m} \mathcal{V}_\infty \cup \Theta_\varepsilon \tag{A.8}$$

where $\Theta_\varepsilon = (\Theta + \varepsilon \bigcup_{k=0}^{m-1} \bar{\mathbf{Q}}_k \mathcal{U}_k)$. Now let $\Theta_o \supset \Theta$ be an open subset of $\mathcal{Z}$. By the compactness of Θ it follows that $\Theta_\varepsilon \subset \Theta_o$ for some $\varepsilon > 0$. Defining the open set $\mathcal{V}_o = \mathcal{V}_\infty + \varepsilon \bar{\mathbf{Q}}_m \mathcal{U}_m$, (A.8) yields $\mathcal{V}_o \subset \bar{\mathbf{A}}_T^{-m} \mathcal{V}_\infty \cup \Theta_o$. If $m = 1$, then $\bar{\mathbf{A}}_T^{-m} = 1$ and $\Theta_\varepsilon = \Theta = \mathcal{O}$. Replacing Θ_ε in (A.8) by $\mathcal{O}$ implies that $\mathcal{V}_o \subset \mathcal{V}_\infty \cup \mathcal{O}$, which together with $\mathcal{V}_\infty \subset \mathcal{V}_o$ proves the lemma for $m = 1$.

A.7. Proof of Theorem 16

We first prove that the minimal sequential length l_k of $\mathcal{P}_k$ is non-decreasing and can remain constant for no more than l_k iterations. To prove that it is non-decreasing we use a contradiction proof. Suppose that $l_{k_1} < l_{k_0}$ for some $k_1 > k_0$. Then there exists a sequence $\{\tau, z_1, \ldots, z_{k_0}, \ldots, z_{k_1}\}$ such that $z_k \in W_k \subset R_-^k(\tau)$, and $\ell(z_{k_1}) < \ell(z_{k_0})$. Since $z_{k_1} \in W_{k_1}$ and $\ell(z_{k_1}) = l_{k_1}$, from Definition 15, it follows that $z_{k_1} \in W_{k_1} \cap R_+^{l_{k_1} - k_1}(\sigma)$. This fact together with $z_{k_0} \in W_{k_0} \cap R_+^{k_1 - k_0}(z_{k_1})$ imply that $z_{k_0} \in W_{k_0} \cap R_+^{l_{k_1} - k_0}$. Hence, $\ell_{k_0} = \ell(z_{k_0}) \leq \ell_{k_1}$, which is a contradiction. Therefore, l_k is always non-decreasing. For l_k to remain constant between $k_0 > 0$ and $k_1 < k_\sigma$, there must again exist a sequence $\{z_{k_0}, \ldots, z_{k_1}\}$ such that $z_k \in R_+^{\ell_{k_0} - k}(\sigma) \cap W_k$, $k_0 \leq k \leq k_1$. We claim that the sequence length $k_1 - k_0 \leq l_{k_0} - 2$. Otherwise, if $k_1 \geq l_{k_0}$ then $z_{k_1} = \sigma$ thus terminating the population iterations. Finally, when l_k reaches k_σ at some $k_t \geq 1$, we shall show that the search ends in at most $k_\sigma - 1$ iterations. Since $\sigma \in R_-^{k_\tau}(\tau) \cap W_{k_\sigma}$, there exists a sequence $\{\tau, \ldots, z_{k_t}, \ldots, z_{k_\sigma - 1}, \sigma\}$, $z_k \in W_k$, implying that $k_\sigma - k_t \leq k_\sigma - 1$. Thus the total number of iterations is at most

$$\begin{aligned} k_{\max} &= 2 + (k_\tau - 2) + (k_\tau - 1) + \cdots + (k_\sigma - 2) \\ &= k_\tau + \tfrac{1}{2}(k_\sigma - k_\tau)(k_\sigma + k_\tau - 3). \end{aligned}$$

The proof of the second half of the Theorem, which includes proving that each s_k updated according to Algorithm 5(ii) is l.s.c and yields an optimal input u_k^* in (5) is completely parallel to that of Theorem 13 and is omitted.

https://doi.org/10.1017/S026357471500106X Published online by Cambridge University Press

A.8. Proof of Proposition 18

Let $\boldsymbol{z} = (\boldsymbol{x}, \boldsymbol{x}^1, \ldots, \boldsymbol{x}^{m-1}) \in \mathcal{Z}$, where $\boldsymbol{x}^j = d^j\boldsymbol{x}/dt^j$, $\boldsymbol{x}^0 = \boldsymbol{x}$. Then $\boldsymbol{z}' = \boldsymbol{\varphi}_\pm(\boldsymbol{z}, \boldsymbol{u})$ can be expressed as $\boldsymbol{z}' = (\boldsymbol{y}, \boldsymbol{y}^1, \ldots, \boldsymbol{y}^{m-1})$ with

$$y_i^j = \sum_{\upsilon=1}^{m-j-1} \frac{(\pm T)^\upsilon x_i^{\upsilon+j}}{\upsilon!} + \frac{(\pm T)^{(m-j)}}{(m-j)!} u_i \tag{A.9}$$

for $i = 1, \ldots, n$, and $j = 0, \ldots, m-1$ where x_i^j and y_i^j are the ith component of $\boldsymbol{x}^j$ and $\boldsymbol{y}^j \in \mathbb{R}^n$, respectively. We shall use an induction argument to prove that there exists $\delta_i^j > 0$ that divides x_i^j of $(\boldsymbol{x}, \boldsymbol{x}^1, \ldots, \boldsymbol{x}^{m-1}) = \hat{\mathcal{R}}_\pm^k(\boldsymbol{\tau})$, $\boldsymbol{\tau} = (\boldsymbol{x}^*, 0, \ldots, 0)$. From (A.9) it can be seen that for $k = 1$, $x_i^j = \frac{(\pm T)^{(m-j)}}{(m-j)!} u_i$ if $j = 1, \ldots, m-1$ and $x_i = x_i^* + \frac{(\pm T)^m}{m!} u_i$ where $u_i = 0$ or $u_i = \pm u_{i,\max}$. Clearly, $\frac{T^{(m-j)}}{(m-j)!} u_{i,\max}$ divides x_i^j for $j = 1, \ldots, m-1$ and $y_i - x_i^*$ for $j = 0$. Now suppose that $\boldsymbol{z} \in \hat{\mathcal{R}}_\pm^k(\boldsymbol{\tau})$ satisfies the induction hypothesis. We shall show that $\boldsymbol{z}' = \boldsymbol{\varphi}_\pm(\boldsymbol{z}, \boldsymbol{u}) \in \hat{\mathcal{R}}_\pm^{k+1}(\boldsymbol{\tau})$ also satisfies the induction hypothesis. Let $x_i^j = \alpha_i^j \frac{T^{(m-j)}}{(m-j)!} u_{i,\max}$ where $\alpha_i^j \in \mathbb{Z}$, for $j = 1, \ldots, m-1$. Substituting this x_i^j into (A.9) yields

$$y_i^j = \sum_{\upsilon=0}^{m-j-1} \frac{(\pm 1)^\upsilon \alpha_i^{\upsilon+j} T^{(m-j)} u_i}{\upsilon!(m-j-\upsilon)!} + \frac{(\pm T)^{(m-j)} u_i}{(m-j)!}.$$

Since the ratio $\frac{(m-j)!}{\upsilon!(m-j-\upsilon)!}$ is an integer (υ out of $m-j$ combinations), it follows that $\frac{T^{(m-j)}}{(m-j)!} u_{i,\max}$ divides y_i^j for $j = 1, \ldots, m-1$ and $y_i - x_i^*$ for $j = 0$. Furthermore, each $y_i^j \in (-x_{i,\max}^j, x_{i,\max}^j)$, $j = 1, \ldots, m-1$ and $y_i \in (x_{i,\min}, x_{i,\max})$ for some finite $x_{i,\min}$ and $x_{i,\max} \in \mathbb{R}$ (see the proof of Lemma 2 for details) implying that the number of discrete states within $\hat{\mathcal{R}}_\pm^k(\boldsymbol{\tau})$, denoted by n_k, is an increasing sequence of integers with an upper bound. Thus there exist K_T such that $n_k = n_{K_T}$, $\forall k \geq K_T$, and the conclusions follow.

A.9. Proof of Lemma 19

Let $\boldsymbol{z} = (\boldsymbol{y}, \boldsymbol{0}, \ldots, \boldsymbol{0})$ and $\boldsymbol{w} = (\boldsymbol{x}, \boldsymbol{0}, \ldots, \boldsymbol{0})$. Since $\mathcal{X}_\varepsilon$ is convex, the line joining $\boldsymbol{x} = (x_1, \ldots, x_n)$ to $\boldsymbol{y} = (y_1, \ldots, y_n)$ lies in $\mathcal{X}_\varepsilon$: $\boldsymbol{x}_\lambda = (1-\lambda)\boldsymbol{x} + \lambda\boldsymbol{y} \in \mathcal{X}_\varepsilon$, $\lambda \in [0, 1]$. As in the proof of Lemma 2 we can construct $\lambda : [0, T_m] \to [0, 1]$, $T_m = 2^{m-1}T$ ($T_0 = T$) such that $\lambda(0) = 0$, $\lambda(t) = 1$, $t \geq T_m$ and $d^m\lambda/dt^m = f_m(t)$ is a piecewise constant two-value function on $[0, T_m]$.

To construct the actual solution, let $\rho_i : \overline{\mathcal{X}} \to \mathbb{R}$ be the projection map that maps $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathcal{X}$ to its ith component x_i: $\rho_i(\boldsymbol{x}) = x_i$. Since $\overline{\mathcal{X}}$ is compact and convex, so is its image under the projection $\rho_i(\overline{\mathcal{X}})$ implying that each $\rho_i(\overline{\mathcal{X}})$ is a compact interval $[x_{i,\min}, x_{i,\max}]$. Letting $\mu := |f_m(t)| = c_m T^{-m}$, $c_m = 2^{-(m-1)(m-2)/2}$, choose the integer $P_i \geq 1$ such that

$$\frac{\mu\,|y_i - x_i|}{u_{i,\max}} - 1 \leq P_i \leq \frac{\mu\,|y_i - x_i|}{u_{i,\max}} \tag{A.10}$$

The compactness of $\overline{\mathcal{X}}$ ensures that $P := \max_i P_i \leq \sup_i \frac{\mu|x_{i,\max} - x_{i,\min}|}{u_{i,\max}}$ exists. Now let

$$\hat{y}_i = x_i + \mu^{-1} u_{i,\max} P_i \operatorname{sign}(y_i - x_i) \tag{A.11}$$

and $L_i = |\hat{y}_i - x_i| = \mu^{-1} u_{i,\max} P_i$. The difference between the ith component of $\boldsymbol{y}$ and its approximation $\hat{y}_i$ is

$$y_i - \hat{y}_i = (y_i - x_i)\left(1 - \frac{\mu^{-1} u_{i,\max} P_i}{|y_i - x_i|}\right).$$

By Inequality (A.10), $0 \leq 1 - \frac{\mu^{-1} u_{i,\max} P_i}{|y_i - x_i|} \leq \frac{\mu^{-1} u_{i,\max}}{|y_i - x_i|}$. Thus $|\hat{y}_i - y_i| \leq \mu^{-1} u_{i,\max}$, which tends to zero as $T \to 0$.

https://doi.org/10.1017/S026357471500106X Published online by Cambridge University Press

Let $q_{i,p}$ be the closest integer to pP_i/P, $i = 1, \ldots, n$ so that $|pP_i/P - q_{i,p}| \leq 1/2$, and define the sequences $x^0_{p,i} = x_i + p(y_i - x_i)/P \in \mathcal{X}_\varepsilon$ and $\hat{x}^0_{p,i} = x_i + q_{i,p}(\hat{y}_i - x_i)/P_i$, $p = 1, \ldots, P$. From (A.11) and the inequality $|\hat{y}_i - y_i| \leq \mu^{-1}u_{i,\max}$, it follows that $|\hat{x}^0_{p,i} - x^0_{p,i}| \leq |\frac{p}{P} - \frac{q_{i,p}}{P_i}||\hat{y}_i - x_i| + \frac{p}{P}|\hat{y}_i - y_i| \leq \frac{3}{2}\mu^{-1}u_{i,\max}$. Choosing

$$T^m < \frac{2\varepsilon c_m}{3} \min_i u^{-1}_{i,\max} \tag{A.12}$$

guarantees that $\max_i |\hat{y}_i - y_i| \leq \varepsilon$ and $\hat{x}^0_{p,i} \in \mathcal{X}_\varepsilon + B_\varepsilon \subset \mathcal{X}$. Furthermore, it can be seen that

$$\left|\hat{x}^0_{p+1,i} - \hat{x}^0_{p,i}\right| = \frac{L_i}{P_i}\left|q_{i,p+1} - q_{i,p}\right| = 0 \text{ or } \frac{L_i}{P_i}. \tag{A.13}$$

We now show that $\hat{\boldsymbol{x}}_p(t) = \hat{\boldsymbol{x}}^0_p + (\hat{\boldsymbol{x}}^0_{p+1} - \hat{\boldsymbol{x}}^0_p)\lambda(t) \in \mathcal{X}$, $t \in [0, T_m]$, $p = 1, \ldots P$, is the desired spline that transfers $\boldsymbol{w}$ to $\hat{\boldsymbol{z}}$. First, the input function corresponding to this spline, $\boldsymbol{u}_p(t) = d^m\boldsymbol{x}_p(t)/dt^m = (\hat{\boldsymbol{x}}^0_{p+1} - \hat{\boldsymbol{x}}^0_p)\lambda^m(t)$, is piecewise constant and using (A.13) we have $u_{p,i}(t) = 0$ or $u_{p,i}(t) = \frac{L_i\lambda^m(t)}{P_i} = \mu^{-1}u_{i,\max}\lambda^m(t)$ for $t \in [0, T_m]$. Thus $u_{p,i}(t)$ is piecewise constant with values 0 and $\pm u_{i,\max}$ as required. Using (A.13) and (A.1) for $j = 1, \ldots, m-1$ and $p = 1, \ldots P$, we have

$$\begin{aligned}\sup_{t\in[0,T_m]}\left|\hat{x}^j_{p,i}\right| &\leq \tfrac{L_i}{P_i}\sup\nolimits_{t\in[0,T_m]}\left|\lambda^j(t)\right| = \mu^{-1}u_{i,\max}\sup\nolimits_{t\in[0,T_m]}\left|\lambda^j(t)\right| \\ &\leq 2^{-(m-j-1)(m-j-2)/2}T^{(m-j)}u_{i,\max}\end{aligned}$$

Therefore, choosing T to satisfy

$$T < \min_{i,j}\left(2^{(m-j-1)(m-j-2)/2}\frac{x^j_{i,\max}}{u_{i,\max}}\right)^{1/(m-j)}$$

in addition to (A.12) yields

$$\max_p \sup_{t\in[0,T_m]}\left|\hat{x}^j_{p,i}\right| \leq x^j_{i,\max}.$$

Setting $\boldsymbol{u}_k = \boldsymbol{u}_p(\ell T)$, $k = Mp + \ell$, $p = 0, \ldots, P-1$, $\ell = 0, \ldots, M-1$, $M = T_m/T = 2^{m-1}$, results in $\hat{\boldsymbol{x}}_k = \hat{\boldsymbol{x}}_p(\ell T) \in \mathcal{X}$ with $\boldsymbol{x}_0 = \boldsymbol{x}$ and $\hat{\boldsymbol{x}}_{MP} \in \boldsymbol{y} + B_\varepsilon$ thus completing the proof.

https://doi.org/10.1017/S026357471500106X Published online by Cambridge University Press