2022 IEEE International Conference on Robotics and Automation (ICRA)
May 23-27, 2022. Philadelphia, PA, USA

# Optimal-Horizon Model Predictive Control with Differential Dynamic Programming

Kyle Stachowicz and Evangelos A. Theodorou

*Abstract*— **We present an algorithm, based on the Differential Dynamic Programming framework, to handle trajectory optimization problems in which the horizon is determined online rather than fixed a priori. This algorithm exhibits exact one-step convergence for linear, quadratic, time-invariant problems and is fast enough for real-time nonlinear model-predictive control. We show derivations for the nonlinear algorithm in the discrete-time case, and apply this algorithm to a variety of nonlinear problems. Finally, we show the efficacy of the optimal-horizon model-predictive control scheme compared to a standard MPC controller, on an obstacle-avoidance problem with planar robots.**

## I. Introduction

Trajectory optimization provides a powerful numerical approach to robot motion planning in systems with complex dynamics. The approach is applicable to offline planning problems [1], and with increased availability of high-powered onboard compute, to online settings. In particular, model-predictive control approaches [2], [3] present a compelling framework for achieving reactive control in complex environments, by iteratively replanning up until a receding horizon at each instant in time. These approaches allow the controller to approximate a infinite-horizon control policy in a computationally tractable manner.

Planning and control tasks often have the property that they should eventually *complete*, rather than running indefinitely. Parking problems, mobile robot navigation, and swing-up problems share this requirement. Additionally, it is desirable to complete the task quickly (balanced against other objective costs), as shorter solutions may allow the robot to immediately move on to another task.

With standard online trajectory optimization approaches, the time horizon is a fixed design parameter with significant impact on the resulting behavior. While some existing methods can bypass this problem — particularly direct solution methods using powerful general-purpose optimizers — these are often too slow to use in an online model-predictive control setting, which can require the optimal control to be computed tens to hundreds of times per second.

In these settings, which in general may include nonlinear dynamics and non-quadratic costs, methods based on the differential dynamic programming algorithm and its derivatives (i.e. iLQR [4]) have become very popular due to their relative computational efficiency [5]. These approaches take the problem's temporal structure into account, using dynamic programming to achieve an efficient solution at each instant in time. In addition, they generalize easily to the stochastic case [6] and have been applied to partially-observable problems through use of belief-space planning [7]. There exist well-known techniques to easily introduce control constraints [8] and arbitrary nonlinear state constraints [9], [10] using Augmented Lagrangian and penalty methods.

In this work we are primarily concerned with solving the following problem for a system with fixed discrete-time dynamics $f$ and cost $\ell$:

$$\min_{T,u} \sum_{t=0}^{T-1} \ell(x_t, u_t) + \Phi(x_T),\ x_{t+1} = f(x_t, u_t) \tag{1}$$

This paper is organized as follows: in Section III, we present an exact one-pass solution for the linear time-invariant problem in continuous and discrete time. Section IV describes an approximate extension of this solution to the general case and uses it to formulate a real-time model-predictive control algorithm. Section V applies this algorithm to several problems, including a nonlinear quadrotor and a point-mass navigation problem.

## II. Related Works

In the formulation described in equation 1, the number of knot points $T$ (which also uniquely determines the horizon) must be determined by the solver. This does not fit well into standard optimization frameworks, so prior attempts to solve the optimal-horizon trajectory optimization problem have used different formulations.

Several methods [11] [12] handle a free final time horizon in the continuous-time case by computing the value function's expansions with respect to the horizon. However, these methods are unstable without large regularization and converge slowly in practice, and cannot be applied to discrete-time problems. A common approach is to fix the number of steps and optimize the timestep $\Delta t$ [13], [14]:

$$\min_{\Delta t,u} \sum_{k=0}^{N-1} \ell(x_k, u_k)\Delta t + \Phi(x_N),\ \Delta x_{k+1} = f(x_k, u_k)\Delta t \tag{2}$$

This method is sensitive to the initial choice of $\Delta t$ [15], and can be slow to converge. Additionally, the modification of the timestep by the solver can allow solutions that exploit discretization error, for example by making $\Delta t$ large and stepping across an obstacle within a single step [14].

The Timed-Elastic Bands formulation [16] uses a general-purpose nonlinear optimizer to attempt to achieve time-optimal point-to-point robot motion. However, unlike im-

The authors are with the Autonomous Control and Decision Systems Laboratory, Georgia Institute of Technology, Atlanta, GA, USA. Email correspondance to: kwstach@gatech.edu.

E. A. Theodorou is supported by NSF-CMMI Award #1936079.

2022 IEEE International Conference on Robotics and Automation (ICRA) | 978-1-7281-9681-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICRA46639.2022.9812036

978-1-7281-9680-0/22/$31.00 ©2022 IEEE

plicit methods like DDP, general-purpose NLP solvers are unable to efficiently exploit the temporal structure present in planning problems. Additionally, problems attempting to solve for the exact time-optimal path must rely on heuristics to avoid chattering when nearing the goal.

In our approach, the expansion of the value function is evaluated (cheaply) at the initial conditions over many possible horizons after each backwards sweep to find an estimate of the objective function for each timestep. The iterative nature of DDP allows us to optimize the horizon and the policy jointly rather than in a bilevel fashion as in [15]. This means that we do not need to wait for convergence before adjusting the horizon.

## III. Linear Time-Invariant Case

The linear time-invariant case gives important insight into the structure of the optimal-horizon control problem. Define the objective function $J(u,T)$ as follows:

$$J(u,T) = \sum_{k=0}^{T-1} \ell(x_k,u_k) + \Phi(x_T)$$

Define the optimal value function (also known as "cost-to-go") with horizon $T$ at time $t$ as $V^{t:T}(x)$:

$$V^{t:T}(x) = \min_u \left[ \sum_{k=t}^{T-1} \ell(x_k,u_k) + \Phi(x_T) \right]$$

Subject to initial condition $x_t = x$ and linear dynamics constraints, and quadratic costs:

$$x_{k+1} = f(x_k,u_k) = Ax_k + Bu_k$$
$$\ell(x,u) = \frac{1}{2}\left[x^\top Qx + u^\top Ru\right] \quad \Phi(x) = \frac{1}{2}x^\top Q_f x$$

In the LQR case $V^{t:T}$ is quadratic with form given by $V^{t:T}(x) = \frac{1}{2}x^\top P^{t:T}x$. The discrete Ricatti equation states that (with $P = P^{t-1:T}$ and $P' = P^{t:T}$ for clarity):

$$P = A^\top P'A - A^\top P'B(R + B^\top P'B)^{-1}B^\top P'A + Q$$

Thus for a fixed horizon $T$ we can calculate $P^{t-1:T}$ backwards starting at $t = T, P^{T:T} = Q_f$.

**Observation 1.** Assume we have a system with time-invariant dynamics and cost. Because dynamics and costs are stationary, shifting the window of time under consideration by an integer $d$, from $t:T$ to $t+d:T+d$, does not change the optimization problem. Thus, $V^{t+d:T+d}(x) = V^{t:T}(x)$.

Note that this property still holds for arbitrary dynamics and costs, provided that both are stationary. We will assume stationary dynamics and costs throughout the remainder of this paper.

Using Observation 1, we can rewrite our minimization:

$$\min_{\pi,T} J(\pi,T) = \min_T V_0^{0:T}(x_0) = \min_T \frac{1}{2}x_0^\top P^{-T:0}x_0$$

The right-hand side can be directly evaluated for each possible horizon $T > 0$ to find a minimizer. However, it is possible for some problems that there is no minimizer for $T$ and that we can make $J$ arbitrarily small by increasing the horizon arbitrarily. For example:

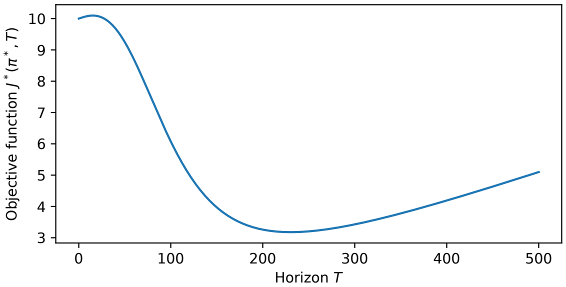


Fig. 1: Optimum objective function by horizon $T$ for a LTI system coincides exactly with $\frac{1}{2}x_0^\top P_{\bar{T}-T}x_0$.

$$A = \begin{pmatrix} 1 & \alpha \\ 0 & 1 \end{pmatrix}, B = \begin{pmatrix} 0 \\ \beta \end{pmatrix}, Q = 0, R = I, Q_f = I$$

In this case $J$ does not achieve its lower-bound $\inf J$ (given by the solution to the infinite-horizon problem) and so $J^*$ does not exist.

We can avoid this in two ways. Firstly, we can constrain $T$ such that $T^- \leq T \leq T^+$, in which case we will clearly perform finitely many steps of the backwards sweep. Otherwise, if we have a lower bound $\delta$ such that $0 < \delta \leq \ell(x,u)$ then a bound on $T^*$ is given by noting that a horizon longer than $T$ will have cost at least $\delta T$, which will for large $K$ be larger than the cost of doing nothing ($T = 0$):

$$\delta T^* \leq J(T^*,u) \leq \frac{1}{2}x_0^\top Q_f x_0$$

This second condition can be guaranteed when time is explicitly penalized by adding a constant term $c_t$ to $\ell(x,u)$ so that the objective function is equivalent to:

$$J(T,u) = \sum_{k=0}^{T-1} \frac{1}{2}\left[x_k^\top Qx_k + u_k^\top Ru_k\right] + \frac{1}{2}x_T^\top Q_f x_T + c_t T$$

Figure 1 shows a plot of the value function $V^{0:T}(x_0)$ for a linear time-invariant system with positive time penalty $c_t$ and large terminal cost $Q_f$ for fixed $x_0$. Notice that $V^{0:T}(x_0)$ is very large with small horizon and decreases rapidly as it becomes feasible to reach closer to the goal within the time horizon, before again increasing as the $c_t$ term dominates.

## IV. Iterative Algorithm for Nonlinear/Non-Quadratic Case

### A. Backwards Sweep

We will again use the notation $V^{t:T}(x)$ to denote the value function at time $t$ with horizon $T$. However, we will drop the horizon (i.e. $V^t(x)$) for clarity when only a single time horizon is under consideration, including in the backwards sweep. For a more thorough treatment of the derivation of iLQR/DDP refer to [4] or [11].

We expand the value function around a nominal trajectory $\bar{x}_t, \bar{u}_t$, with $\delta x_t = x_t - \bar{x}_t, \delta u_t = u_t - \bar{u}_t$.

$$V^t(x) \approx \tilde{V}^t(x) = \frac{1}{2}\delta x_t^\top V_{xx}^t \delta x + V_x^t \delta x_t + V_0^t \qquad (3)$$

At the final timestep $T$ we simply have:

$$V^T(x) = \Phi(x) \approx \frac{1}{2}\delta x^\top \Phi_{xx}\delta x + \Phi_x \delta x + \Phi$$

So $V_{xx}^T = \Phi_{xx}$, $V_x^T = \Phi_x$, and $V_0^T = \Phi(\bar{x})$. Then, we can apply Bellman's principle to get a state-action value function $Q_t^t(x,u) = \min_u \left[\ell(x,u) + V^{t+1}(f(x,u))\right]$ and expand around the nominal trajectory, substituting our value function approximation for $V$:

$$\begin{aligned} V^t(x) &= \min_u \left[\ell(x,u) + V^{t+1}(f(x,u))\right] \qquad (4) \\ &\approx \frac{1}{2}\min_u \begin{pmatrix}\delta x\\ \delta u\\ 1\end{pmatrix}^\top \begin{pmatrix} Q_{xx} & Q_{ux}^\top & Q_x^\top \\ Q_{ux} & Q_{uu} & Q_u^\top \\ Q_x & Q_u & Q\end{pmatrix}\begin{pmatrix}\delta x\\ \delta u\\ 1\end{pmatrix}\end{aligned}$$

The expansions of $Q(x,u)$ are derived explicitly in [11] and in [4], which ignores second-order dynamics terms. Let $K_t = -Q_{uu}^{-1}Q_{ux}$ and $k_t = -Q_{uu}^{-1}Q_u$. Then, the minimizing control $u_t$ is given by $\delta u_t = K_t \delta x + k_t$, and we have:

$$\begin{aligned} V_{xx}^t &= Q_{xx} - Q_{ux}^\top Q_{uu}^{-1} Q_{ux} \\ V_x^t &= Q_x - Q_{ux}^\top Q_{uu}^{-1} Q_u \\ V_0^t &= Q - \frac{1}{2} Q_u^\top Q_u^{-1} Q_u \end{aligned}$$

Given the terminal conditions and the backwards recurrence equation we can then solve for $V_{xx}^t, V_x^t, V_0^t$ for all $t < T$, yielding an approximation for the cost of a trajectory starting at any state $x$ at time $t$ and continuing until $T$.

### B. Horizon Selection

Analogously to the LTI case, we see that the value function approximation $V^{t:\bar{T}}(x)$ provides an estimate for the true value function $V^{0:\bar{T}-t}(x)$ for $x$ near $x_t$. The minimizing horizon will by definition be achieved by selecting $T$ such that $V^{0:T}(x_0)$ is minimal. While unlike the LTI case we lack an exact description of $V^{0:T}(x)$, we can approximate it using our quadratic value function approximation $\tilde{V}$. Then, our problem becomes:

$$T = \bar{T} - \min_t \tilde{V}^{t:\bar{T}}(x_0) \qquad (5)$$

This optimization problem simply consists of evaluating finitely many quadratic functions at $x_0$ and selecting the minimum, which is computationally trivial. In practice, we can restrict this search to only a window around the nominal horizon $\bar{T}$, from $\bar{T} - S$ to $\bar{T} + S$ for some integer $S$ as the value function approximation is unlikely to remain accurate for large jumps in horizon.

Calculating this local expansion requires a nominal trajectory around which to calculate derivatives of the cost and dynamics. While this is acceptable for selecting a horizon that is *shorter* than the current nominal horizon — corresponding to selecting the window $t : T$ for positive $t$ — it does not immediately allow for *lengthening* the horizon, which would correspond to a negative $t$. To remedy this we can lengthen our nominal trajectory by adding a segment from $-S$ to 0, such that $(\bar{x}_t, \bar{u}_t)$ exists over $-S \le t < \bar{T}$. For the above equations to apply, the trajectory need only be dynamically feasible and is otherwise arbitrary - for example, it can be calculated by integrating the system backwards with zero input.

The error of the value function approximation determines the accuracy of the overall algorithm: this selection process may choose a suboptimal horizon depending on the accuracy of the approximation. In particular, if the true optimal horizon $T^*$ has optimal value $V^*(x_0)$ but is overapproximated as $\tilde{V}^{(\bar{T}-T^*):\bar{T}}(x_0) = V^*(x_0)+\epsilon$ by the quadratic value function approximation, then any horizon with value less than $\tilde{V}(x_0)$ may be chosen instead, giving a suboptimal cost by as much as $\epsilon$. In the next section, a bound on this error is derived in the case of bounded higher-order derivative terms.

**Algorithm 1:** Horizon Selection and Forwards Pass

```
while No cost-reducing solution found do
    $T^* \leftarrow \bar{T}, J^* \leftarrow \infty$;
    for $T \leftarrow \bar{T} - S$ to $\bar{T} + S$ do
        $t_0 \leftarrow \bar{T} - T$;
        $\delta x_0 \leftarrow x_0 - \bar{x}_{t_0}$;
        $J_T \leftarrow \frac{1}{2}\delta x_0^\top V_{xx}^{t_0:\bar{T}}\delta x_0 + V_x^{t_0:\bar{T}}\delta x_0 + V_0^{t_0:\bar{T}}$;
        if $J_T < J^*$ then
            $J^* \leftarrow J_T$;
            $T^* \leftarrow T$;
        end
    end
    $J, \bar{x}', \bar{u}' \leftarrow$ Rollout policy:
        $u_t = K_{t_0+t}\delta x_t + \alpha k_{t_0+t} + \bar{u}_{t_0+t}$;
    if $J < J_{prev}$ then
        return $T^*, \bar{x}', \bar{u}'$;
    else
        Adjust $S$, $\alpha$
end
return $\bar{x}', \bar{u}', T^*$
```

**Algorithm 2:** Optimize Trajectory and Horizon

```
$\bar{T}, \bar{x}, \bar{u} \leftarrow$ initial trajectory;
while $J < J_{prev} - \epsilon$ do
    $V_{xx}, V_x, V_0, K, k \leftarrow$ BackwardsPass($\bar{T}, \bar{x}, \bar{u}$);
    $\bar{T}, \bar{x}, \bar{u} \leftarrow$
      FormwardsPassSelectHorizon($V_{xx}, V_x, V_0, K, k$);
end
```

### C. Analysis of Approximation Error

Throughout this section, multivariate polynomials of higher than second order will be considered in the Taylor expansion of the value function. In keeping with the convention of expressing quadratic forms as bilinear forms

$(\mathbb{R}^k, \mathbb{R}^k) \to \mathbb{R}$, we will likewise express homogeneous cubics as 3-forms $(\mathbb{R}^k, \mathbb{R}^k, \mathbb{R}^k) \to \mathbb{R}$, and so on.

For a linear map $f : \mathbb{R}^j \to \mathbb{R}^k$, we define the operator norm of $f$ as:

$$\|f\| = \max_x \left(\|fx\| \cdot \|x\|^{-1}\right)$$

For a quadratic or cubic $n$-form $G : (\mathbb{R}^k)^n \to \mathbb{R}$ we will define a similar "norm" noting that for symmetric multilinear $G$, it holds that $G(ax, ax, \dots) = a^n G(x)$ and observing that the triangle inequality and positivity hold:

$$\|G\| = \max_x \left(|G(x, x, ...)| \cdot \|x\|^{-n}\right)$$

Note that with this definition $|G(x, x, ...)| < \|G\| \|x\|^n$. This extends the idea of the spectral norm of a symmetric matrix.

Additionally, for a function $G(x, u)$ of the state and control at time $t$, under a fixed policy $u = \pi(x)$, we will denote the total derivative as $G_h = \frac{\partial}{\partial x} G(x, \pi(x)) = G_x + G_u \pi_x$.

**Theorem 1.** *Let $V^{t:T}(x)$ be the true value function for a fixed horizon, and let $\tilde{V}^{t:T}(x)$ be its second-order Taylor approximation around a nominal trajectory $\bar{x}_t, \bar{u}_t$. Assume the true optimal policy is given by $\pi$, and that the closed-loop error dynamics $\delta x_{t+1} = f(\bar{x}_t + \delta x_t, \pi(\bar{x}_t + \delta x_t))$ are stable at each timestep with all eigenvalues less than some constant $L < 1$. Finally, assume that the high-order terms are bounded as follows for some $M \in \mathbb{R}$:*

$$\|\ell_{hhh}\| + \|V_x^t f_{hhh}\| + 3\|f_h^\top V_{xx}^t f_{hh}\| \le M$$

*Then, the total approximation error of the value function at any stage is bounded by:*

$$|V^t(x) - \tilde{V}^t(x)| \le \frac{M}{1 - L^3} \|x - \bar{x}_t\|^3 + \mathcal{O}(\|x - \bar{x}_t\|^4)$$

*Proof.* Because the value function approximation is a Taylor expansion up to degree 2, we only need to consider third-order and higher terms. Let $c = \|x_t - \bar{x}_t\|$:

$$\begin{aligned} |V^t(x) - \tilde{V}^t(x)| &\le \left|V^t_{xxx}(x - \bar{x}_t)\right| + \mathcal{O}(c^4) \\ &\le \left|\max_{v \in S^{n-1}} V^t_{xxx}(v) c^3\right| + \mathcal{O}(c^4) \end{aligned}$$

Then, apply the Bellman equation to the third-order term:

$$\begin{aligned} &\|V^t_{xxx}\| = \left\|\ell_{hhh}(v, v, v) + (V^{t+1} \circ f)_{hhh}(v, v, v)\right\| \\ &\le \|\ell_{hhh}\| + \|V_x^{t+1} f_{hhh}\| + 3\|f_h^\top V_{xx}^{t+1} f_{hh}\| + \|V_{xxx}^{t+1}\| \|f_h\|^3 \end{aligned}$$

This yields the recurrence relation $\|V^t_{xxx}\| \le M + \|V^{t+1}_{xxx}\| L^3$, which can be solved to yield an upper bound of $\|V^t_{xxx}\| \le \frac{M}{1 - L^3}$. Substituting into our original bound, we get:

$$|V^t(x) - \tilde{V}^t(x)| \le \frac{M}{1 - L^3} \|x - \bar{x}_t\|^3 + \mathcal{O}(c^4)$$

□

This inequality yields several key insights:

1) The error bound is only nonzero in the presence of terms ignored by DDP.
2) If the optimal fixed-horizon solution is stable, the bound exists for any horizon.
3) The error is cubic in deviation from the trajectory.

Therefore, assuming stationary dynamics and cost functions, the horizon selection algorithm will select a horizon that is at worst $\epsilon$-suboptimal assuming that DDP has converged to an optimal fixed-horizon solution. Note that in practice convergence is not necessary to select a new horizon, as shown in Figure 2, and we can in fact select a new horizon after every iteration of DDP.

### *D. Forwards Sweep*

Finally, we use our linear controller defined by $K_t, k_t$ to calculate a new nominal trajectory in the forwards pass, starting with our fixed $x_0$:

$$u_t = K(x - \bar{x}) + k + \bar{u}_t \qquad x_{t+1} = f(x, u_t) \tag{6}$$

These $x_t, u_t$ then become the $\bar{x}_t, \bar{u}_t$ for the next iteration. These two procedures repeat until convergence.

The value function and controller yielded by DDP are only valid near the nominal trajectory. To avoid taking large steps outside of this region of validity, it is common to use line search techniques on the forwards pass to ensure convergence. If the forwards pass does not yield a reduction in cost, it can be discarded and a new candidate solution can be computed by decreasing the deviation $k$ until the cost is reduced. In a similar vein, the value function approximation may be invalid for large $\bar{T} - T$. We can address this issue by only considering $T$ within a window of $\bar{T}$, and shrinking this window if cost increases.

Figure 2 shows the value-function approximation after a single iteration of DDP and a fully converged instance of fixed-horizon DDP. While convergence yields a better approximation for the value function, the single-iteration approximation often gives a good approximation in a neighborhood of the nominal horizon. This observation indicates that our algorithm is able to operate effectively without the need for a bilevel optimization approach.

### *E. Model-Predictive Control*

Model predictive control is a common approach to handling the dynamically changing environments in which robots often operate. In this scheme, at time $t$, an optimal plan $(x_k, u_k)$ is computed for $t \le k \le t + T$, with $x_t$ as the robot's current state. Then, the first action $u_t$ is applied and a state estimator calculates a new $x_{t+1}$. Finally, a new plan $(x_k, u_k)$ is computed for $t + 1 \le k \le t + T + 1$, and the cycle repeats.

Algorithm 3 describes a model-predictive scheme for control with optimal-horizon controllers. Unlike the receding-horizon context, the algorithm is also responsible for determining when to terminate the action, which is important for cases in which the robot must balance time-to-completion of the task with accuracy — for example if it must begin a separate task after completing a motion.

When the system dynamics and cost function are known exactly in advance by the controller, the MPC formulation is unnecessary (the update rule becomes $\pi \leftarrow \pi_{t+1:t+T}$, dropping the first timestep). However, the policy update

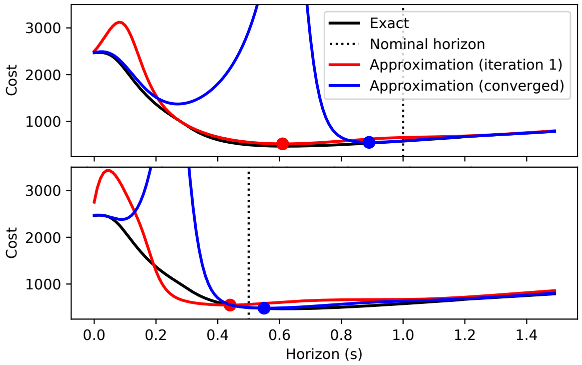


Fig. 2: True optimal cost (black) vs. approximated cost via value-function expansion $V^{t:\bar{T}}$ after a single iteration of DDP (red) and upon convergence (blue), shown for two different nominal horizons $\bar{T}$. Often, the initial approximation is good enough to select a new horizon (bottom), but occasionally it is necessary to run multiple iterations to achieve a better approximation (top). This is handled in the line-search procedure by rejecting a new horizon if it results in high cost.

will be nontrivial in the presence of any of the following disturbances:

- Noisy or incorrect dynamics model, such that the real value of $x_{t+1}$ is not identical to its predicted value
- Uncertain future cost landscape - for example, randomly moving obstacles

In real-world robotic systems, both of these conditions hold and simply tracking a pre-planned trajectory will fail, highlighting the importance of MPC controllers.

**Algorithm 3:** Model-Predictive Control

```
$(\bar{x}, \bar{u}), \bar{T} \leftarrow$ Initial computed trajectory;
while $\bar{T} > 1$ do
    $(\bar{x}, \bar{u}), \bar{T} \leftarrow$ OptimizeTrajectory($\bar{x}, \bar{u}, \bar{T}$);
    Apply action $\bar{u}_0$;
    Drop $(\bar{x}_0, \bar{u}_0)$;
    $\bar{T} \leftarrow \bar{T} - 1$;
end
Terminate task;
```

## V. Results

We compared our algorithm against several existing solution methods for optimal-horizon problems: Sun et. al [12], a continuous-time DDP approach, and the standard direct transcription approach using the an off-the-shelf interior-point NLP solver [17]. Table I gives a summary of the three solvers' performance on each of the following four problems.

We implemented the presented nonlinear optimal-horizon trajectory optimization algorithm in C++, with derivatives computed numerically.

| | **Ours** | | **Sun et. al** | | **IPOPT** | |
|---|---|---|---|---|---|---|
| **Problem** | **Time** | **Iter.** | **Time** | **Iter.** | **Time** | **Iter.** |
| Linear System | 1ms | 1 | 4.76s | 100 | 0.223s | 37 |
| Cartpole | 9ms | 35 | 213s | 3000 | 6.415s | 1223 |
| Quadrotor | 101ms | 25 | — | | — | |
| Navigation | 15ms | 26 | — | | 10.273s | 507 |
| Nav. (MPC) | 5ms | 9 | — | | 0.581s | 31 |

TABLE I: Comparison of computation times and iterations between our algorithm, the solver presented in Sun et. al [12], and IPOPT [17] with problem formulation described in Equation 2. Entries marked with a dash were not solved.

| $c_t$ | Ours | Exact | Cost % error |
|---|---|---|---|
| 1.0 | 3.31 | 3.31 | 0.00 |
| 3.0 | 2.53 | 2.54 | 0.00 |
| 10.0 | 1.79 | 1.8 | 0.01 |
| 30.0 | 1.65 | 1.65 | 0.00 |
| 100.0 | 1.47 | 1.41 | 0.27 |

TABLE II: Optimal horizons for cartpole on discrete-time problem with varying $c_t$

### A. Linear System

We validated the results of Section III by running our algorithm against a simple linear time-invariant double-integrator. As expected, only a single iteration was required to compute an optimal horizon and policy.

### B. Cartpole Swing-Up

The goal of the swing-up task is to cause a pendulum to arrive at the upright position $\theta = \pi$ with nearly zero velocity as quickly as possible, through linear motion of a cart to which its base is attached. The objective function consists of a quadratic running-cost on $\dot{x}$ and on $\dot{\theta}$ and time-penalization $c_t$ (a constant term added to $\ell$), as well as a large terminal cost on $(\theta - \pi)^2$ and terminal velocities.

We swept the parameter $c_t$ to find its effect on optimal horizon, as shown in Table II. The "exact" horizons in this table are computed exhaustively using a naive brute-force algorithm by applying the fixed-horizon DDP algorithm to each possible horizon. Our algorithm recovered a optimal or indistinguishable-from-optimal horizon in every case except $c_t = 100$ (where it still achieves nearly optimal cost).

Figure 3 shows one solution for cartpole with $c_t = 30$. This solution took roughly 8.6 milliseconds to compute. In general, our implementation converges in substantially fewer iterations than both existing DDP-based and general NLP-solver approaches. Figure 4 shows the progression of the optimization problem for each algorithm.

### C. Quadrotor

We also applied the algorithm to generate optimal-horizon trajectories for a nonlinear 3D quadrotor model with 12 state dimensions and 4 control dimensions [18]. The cost function for this simulation is a simple quadratic function of state and control, with the goal of reaching a set final state. We quadratically penalize deviation from a nominal upward

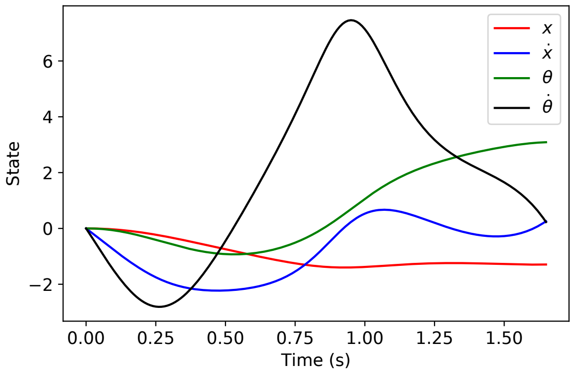


Fig. 3: Solution to cartpole problem ($c_t = 30$)

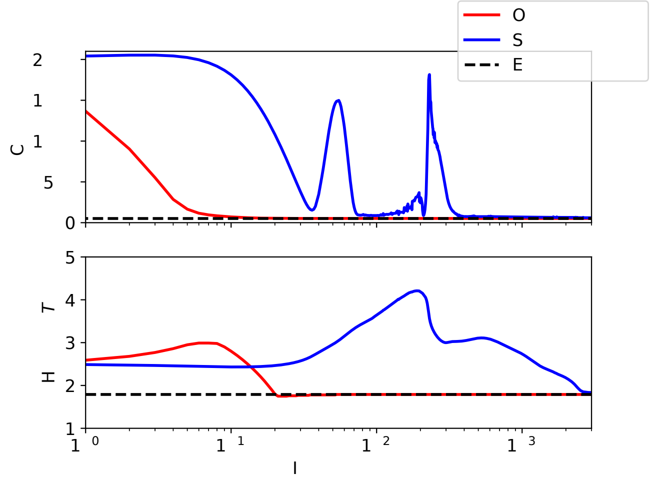
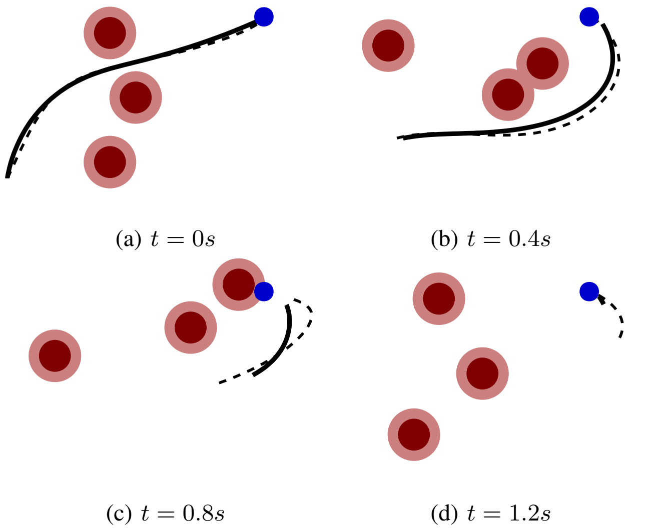


Fig. 4: Comparison between optimizer progress between our method and previous work [12] on the cartpole problem. IPOPT is not shown due to infeasible intermediate results.

thrust equal to the force of gravity and penalize the total time. This problem has high dimensionality and unstable dynamics and was not solved by the min-time IPOPT formulation or the continuous-time formulation from Sun et. al [12].

### D. Point-Mass Robot MPC

Finally, we applied the MPC algorithm described in Section IV-E to the obstacle-avoidance problem for a point-mass mobile robot. In this problem, the robot must reach a pre-specified goal position while avoiding obstacles whose motion is unknown to the controller. We used a double-integrator dynamics model in which acceleration is directly controlled and circular obstacles with cost $\exp(-\frac{\|x-o\|^2}{2r^2})$, where the obstacle has position $o$ and radius $r$ with motion patterns unknown a priori to the planner. The terminal cost was taken to be a quadratic penalizing deviation from the goal. Again, the $c_t$ term incentivizes the robot to arrive at the goal more quickly.


(a) $t = 0s$

(b) $t = 0.4s$

(c) $t = 0.8s$

(d) $t = 1.2s$

Fig. 5: Execution of optimal-horizon MPC (solid) vs receding-horizon (dashed). Our approach reaches the goal (blue) in finite time and terminates the task, while the receding-horizon approach only converges exponentially.

Figure 5 shows four snapshots of a particular experiment with MPC. The planner is able to dynamically adjust the horizon when the obstacle locations change, and ends the episode once it is close to the goal. As the two trajectories approach the goal the receding-horizon implementation only converges exponentially while the optimal-horizon implementation is able to complete in finite time. The MPC step takes on average 5 milliseconds to recompute for each stage when warm-started, easily fast enough for real-time use.

The supplementary material contains a short video of this MPC formulation operating, with the target changing to a randomly-selected position whenever the controller determines that the goal is reached as in Algorithm 3.

## VI. Conclusion

The problem of optimal-horizon planning and control is fundamental in robotic systems and is motivated by common practical problems in the robotics literature. By explicitly including the time horizon in the optimization problem, it is possible to more naturally represent problems in robotics often approximated by infinite-horizon controllers or approximations thereof. Our algorithm is able to generate solutions to the optimal-horizon control problem substantially faster then either of the existing approaches tested.

Another key application of the optimal-horizon MPC problem in robotics comes in the form of explicitly handling stochastic dynamics. By extending this framework to stochastic problems, it will be possible to tackle a much wider array of real-world problems. Additionally, it may be possible to extend a similar approach to derive fast optimal-horizon variants of other commonly-used MPC solvers such as model-predictive path integral [19] for use in model-based reinforcement learning.

# REFERENCES

[1] J. T. Betts, "Survey of numerical methods for trajectory optimization," *Journal of Guidance, Control, and Dynamics*, vol. 21, no. 2, pp. 193–207, Mar. 1998. [Online]. Available: https://doi.org/10.2514/2.4231

[2] Y. Wang and S. Boyd, "Fast model predictive control using online optimization," *IEEE Transactions on Control Systems Technology*, vol. 18, no. 2, pp. 267–278, 2010.

[3] Y. Tassa, T. Erez, and W. Smart, "Receding horizon differential dynamic programming," in *Advances in Neural Information Processing Systems*, J. Platt, D. Koller, Y. Singer, and S. Roweis, Eds., vol. 20. Curran Associates, Inc., 2008. [Online]. Available: https://proceedings.neurips.cc/paper/2007/file/c6bff625bdb0393992c9d4db0c6bbe45-Paper.pdf

[4] W. Li and E. Todorov, "Iterative linear quadratic regulator design for nonlinear biological movement systems." vol. 1, 01 2004, pp. 222–229.

[5] D. M. Murray and S. J. Yakowitz, "Differential dynamic programming and newton's method for discrete optimal control problems," *Journal of Optimization Theory and Applications*, vol. 43, no. 3, pp. 395–414, Jul 1984. [Online]. Available: https://doi.org/10.1007/BF00934463

[6] E. Theodorou, Y. Tassa, and E. Todorov, "Stochastic differential dynamic programming," in *Proceedings of the 2010 American Control Conference*, 2010, pp. 1125–1132.

[7] J. van den Berg, S. Patil, and R. Alterovitz, *Motion Planning Under Uncertainty Using Differential Dynamic Programming in Belief Space*. Cham: Springer International Publishing, 2017, pp. 473–490. [Online]. Available: https://doi.org/10.1007/978-3-319-29363-9_27

[8] Y. Tassa, N. Mansard, and E. Todorov, "Control-limited differential dynamic programming," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 1168–1175.

[9] Z. Xie, C. K. Liu, and K. Hauser, "Differential dynamic programming with nonlinear constraints," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 695–702.

[10] Y. Aoyama, G. Boutselis, A. Patel, and E. Theodorou, "Constrained differential dynamic programming revisited," 05 2020.

[11] D. Jacobson and D. Mayne, *Differential Dynamic Programming*, ser. Modern analytic and computational methods in science and mathematics. American Elsevier Publishing Company, 1970. [Online]. Available: https://books.google.com/books?id=tA-oAAAAIAAJ

[12] W. Sun, E. Theodorou, and P. Tsiotras, "Model based reinforcement learning with final time horizon optimization," 2015.

[13] J. van den Berg, *Extended LQR: Locally-Optimal Feedback Control for Systems with Non-Linear Dynamics and Non-Quadratic Cost*. Cham: Springer International Publishing, 2016, pp. 39–56. [Online]. Available: https://doi.org/10.1007/978-3-319-28872-7_3

[14] T. A. Howell, B. E. Jackson, and Z. Manchester, "ALTRO: A fast solver for constrained trajectory optimization," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 7674–7679.

[15] A. De Marchi and M. Gerdts, "Free finite horizon LQR: A bilevel perspective and its application to model predictive control," *Automatica*, vol. 100, pp. 299–311, 02 2019.

[16] C. Rösmann, F. Hoffmann, and T. Bertram, "Timed-elastic-bands for time-optimal point-to-point nonlinear model predictive control," in *2015 European Control Conference (ECC)*, 2015, pp. 3352–3357.

[17] A. Wächter, "An interior point algorithm for large-scale nonlinear optimization with applications in process engineering," Ph.D. dissertation, 2002.

[18] F. Sabatino, "Quadrotor control: modeling, nonlinear control design, and simulation," 2015.

[19] G. Williams, N. Wagener, B. Goldfain, P. Drews, J. M. Rehg, B. Boots, and E. A. Theodorou, "Information theoretic mpc for model-based reinforcement learning," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 1714–1721.