Endpoint-Explicit Differential Dynamic Programming via Exact Resolution

Maria Parillli[1] Sergi Martinez[2] Carlos Mastalli[2]
[1] *Mechatronics Research Group, Universidad Simón Bolıvar, Venezuela.*
[2] *Robot Motor Intelligence (RoMI) Lab, Heriot-Watt University, U.K.*

Introduction

Various approaches have emerged in recent years to address the challenge of parallelizing optimal control (OC) algorithms. To achieve this, it is critical to handle endpoint constraints efficiently. The **contribution of our work** are the follows

- A novel **endpoint-explicit DDP**, which is both exact and computationally efficient, offering **quadratic convergence**.
- Our method **effectively handles rank deficiencies** without increasing computational cost.
- Highly suitable for **MPC applications** with both **forward and inverse dynamics** formulations.

OC problem abstraction

OC problems subject to stagewise and endpoint constraints can be locally resolved by solving the following quadratic program:

$$\min_{\mathbf{w}} \ \frac{1}{2}\mathbf{w}^\intercal\mathbf{A}\mathbf{w} - \mathbf{w}^\intercal\mathbf{a} \quad \text{subject to} \quad \mathbf{B}\mathbf{w} = \mathbf{b}. \tag{1}$$

In this abstraction, $\mathbf{w} \in \mathbb{R}^{n_w}$ represents the **primal and dual decision variables** encountered in an OC problem without endpoint constraints (e.g., search direction for states, controls, and multipliers of the dynamics constraints). Moreover, $\mathbf{B} \in \mathbb{R}^{n_b \times n_w}$ represents the **Jacobian of the endpoint constraint**, while $\mathbf{A} \in S^{n_w}_{++}$ is a large banded matrix that describes the OC problem without endpoint constraints.

The solution to this system is:

$$\mathbf{y} = -\mathbf{S}^{-1}(\mathbf{b} - \mathbf{B}\mathbf{A}^{-1}\mathbf{a}), \tag{2a}$$
$$\mathbf{w} = \mathbf{A}^{-1}\mathbf{a} - \mathbf{A}^{-1}\mathbf{B}^\intercal\mathbf{y}, \tag{2b}$$

where $\mathbf{S} = \mathbf{B}\mathbf{A}^{-1}\mathbf{B}^\intercal$ denotes the Schur complement, and its inversion can be efficiently computed via a sparse Cholesky decomposition. Here, Equation (2a) calculates the Lagrange multiplier associated with the **endpoint constraint**. In contrast, Equation (2b) computes the **primal and dual variables**. This is achieved through two **Riccati recursions** with different initializations. This abstract solution results in a highly efficient algorithm capable of handling arbitrary stagewise constraints.

Endpoint-explicit DDP

Nonlinear OC problems with endpoint constraints can be formulated using either forward or inverse dynamics:

$$\begin{aligned} \min_{\mathbf{x}_s,\mathbf{u}_s} \ & \ell_N(\mathbf{x}_N) + \sum_{k=0}^{N-1} \ell_k(\mathbf{x}_k,\mathbf{u}_k) \\ \text{s.t.} \ & \tilde{\mathbf{x}}_0 \ominus \mathbf{x}_0 = \mathbf{0}, \quad \text{(initial condition)} \quad \mathbf{f}_k(\mathbf{x}_k,\mathbf{u}_k) \ominus \mathbf{x}_{k+1} = \mathbf{0}, \quad \text{(integrator / forward dyn.)} \\ & \mathbf{h}_k(\mathbf{x}_k,\mathbf{u}_k) = \mathbf{0}, \quad \text{(inverse dyn.)} \quad \mathbf{r}(\mathbf{x}_N) = \mathbf{0}. \quad \text{(endpoint constraint)} \end{aligned} \tag{3}$$

In general, our algorithm can be divided into three main steps:

1. We compute the search direction for the **endpoint-independent problem** ($\hat{\mathbf{w}} = \mathbf{A}^{-1}\mathbf{a}$) through Riccati recursion, performing the unconstrained DDP algorithm:

$$\begin{aligned} &\delta\hat{\mathbf{u}} = -\hat{\boldsymbol{\pi}} - \boldsymbol{\Pi}\delta\hat{\mathbf{x}}, \\ &\delta\hat{\mathbf{x}}_{k+1} = \mathbf{f}_{\mathbf{x}_k}\delta\hat{\mathbf{x}}_k + \mathbf{f}_{\mathbf{u}_k}\delta\hat{\mathbf{u}}_k + \bar{\mathbf{f}}_{k+1}, \quad \delta\hat{\mathbf{x}}_0 = \bar{\mathbf{f}}_0, \end{aligned} \qquad \begin{aligned} &\hat{\boldsymbol{\pi}}_u := \mathbf{Q}_{\mathbf{uu}}^{-1}\mathbf{Q}_{\mathbf{u}}, \ \boldsymbol{\Pi}_u := \mathbf{Q}_{\mathbf{uu}}^{-1}\mathbf{Q}_{\mathbf{ux}} \ \text{(forward dyn.)} \\ &\hat{\boldsymbol{\pi}}_s := \mathbf{k} + (\mathbf{k}_s^{\intercal\sim}{}_{\mathbf{uu}}\Psi_s^\intercal)^\intercal, \ \boldsymbol{\Pi}_s := \mathbf{K} + (\mathbf{K}_s^{\intercal\sim}{}_{\mathbf{uu}}\Psi_s^\intercal)^\intercal \\ &\text{(inverse dyn.)} \end{aligned} \tag{4}$$

2. We run another Riccati recursion to compute the **endpoint-dependent search direction** ($\widetilde{\mathbf{W}} = \mathbf{A}^{-1}\mathbf{B}^\intercal$). This step reuses the factorization computed in **(1.)**, and replaces $\mathbf{a}$ by $\mathbf{B}^\intercal$ in Equation (1):

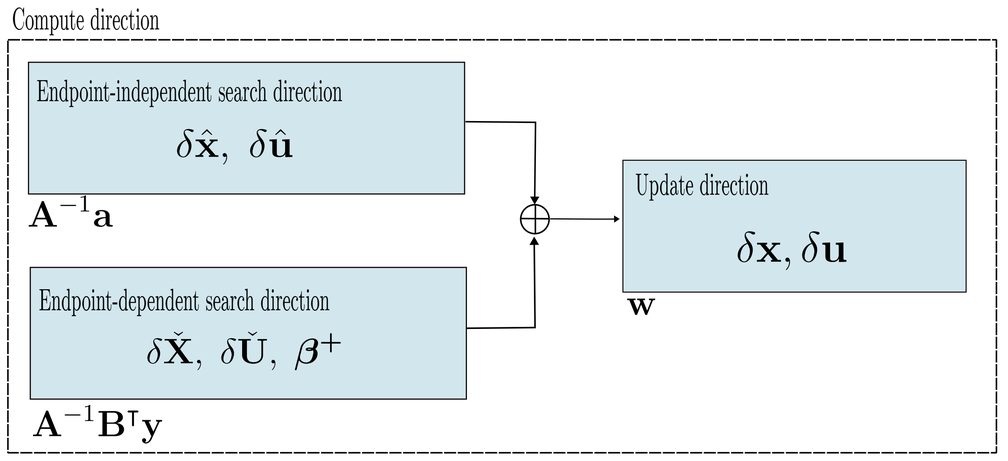
$$\begin{aligned} &\delta\breve{\mathbf{U}}_{\mathbf{c}} = -\breve{\boldsymbol{\pi}}_{\mathbf{c}} - \boldsymbol{\Pi}\delta\breve{\mathbf{X}}_{\mathbf{c}}, \\ &\delta\breve{\mathbf{X}}'_{\mathbf{c}} = \mathbf{f}_{\mathbf{x}}\delta\breve{\mathbf{X}}_{\mathbf{c}} + \mathbf{f}_{\mathbf{u}}\delta\breve{\mathbf{U}}_{\mathbf{c}}, \quad \delta\breve{\mathbf{X}}_{\mathbf{c}_0} = \mathbf{0}, \end{aligned} \qquad \begin{aligned} &\breve{\boldsymbol{\pi}}_{\mathbf{c}_u} := \mathbf{Q}_{\mathbf{uu}}^{-1}\mathbf{Q}_{\mathbf{uc}} \ \text{(forward dyn.)} \\ &\breve{\boldsymbol{\pi}}_{\mathbf{c}_s} := \mathbf{k}_{\mathbf{c}} + (\mathbf{k}_{sc}^\intercal\tilde{\mathbf{Q}}_{\mathbf{uu}}\boldsymbol{\Psi}_s^\intercal)^\intercal \ \text{(inverse dyn.)} \end{aligned} \tag{5}$$

3. From Equation (2a) we calculate the **endpoint multiplier** as

$$\boldsymbol{\beta}^+ = (\mathbf{r}_{\mathbf{x}_N}\delta\breve{\mathbf{X}}_{\mathbf{c}_N})^{-1}(\bar{\mathbf{r}} + \mathbf{r}_{\mathbf{x}_N}\delta\hat{\mathbf{x}}_N). \tag{6}$$

Finally, we update the search direction as follows:

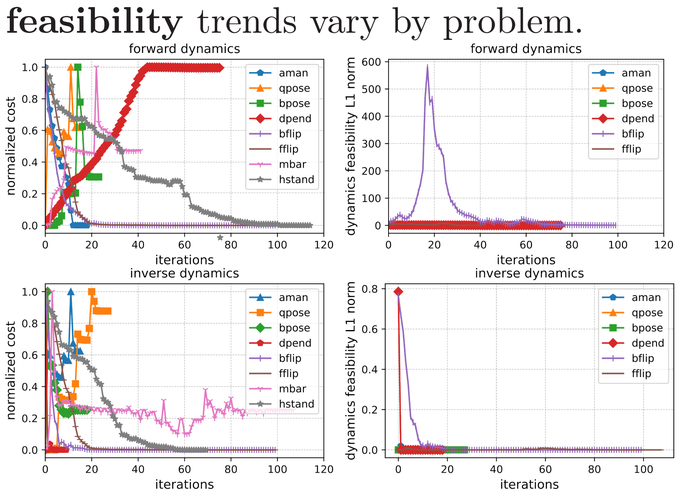
$$\begin{aligned} \delta\mathbf{u} &= \delta\hat{\mathbf{u}} - \delta\breve{\mathbf{U}}_{\mathbf{c}}\boldsymbol{\beta}^+, \\ \delta\mathbf{x}' &= \delta\hat{\mathbf{x}}' - \delta\breve{\mathbf{X}}'_{\mathbf{c}}\boldsymbol{\beta}^+. \end{aligned} \tag{7}$$



Results: Cost and feasibility, Endpoint factorization, Gymnastic Maneuvers and MPC trials

Costs and endpoint feasibility

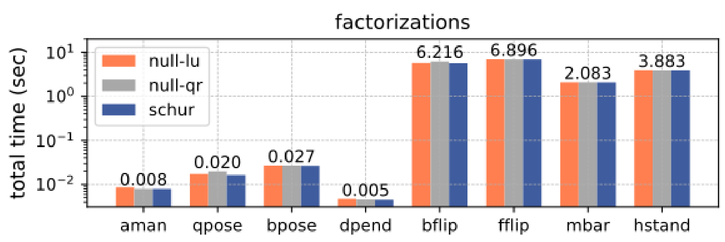
We compared **forward** and **inverse dynamics formulations** to evaluate our endpoint-explicit strategy on problems with different stagewise constraints. Here, inverse dynamics generally converges faster, while **cost** and **feasibility** trends vary by problem.



Endpoint factorization

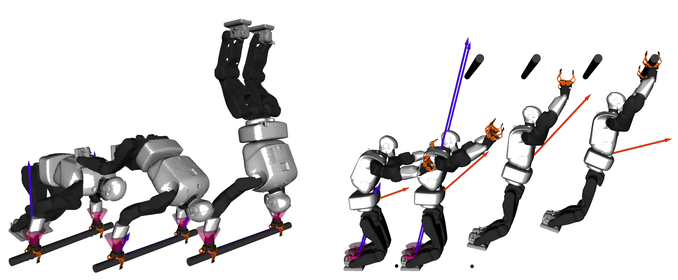
On the other hand, we evaluated our **nullspace factorization**'s ability to handle rank-deficient constraints efficiently.

Compared to the Schur complement method, it achieves **similar computation times using LU or QR with pivoting**.



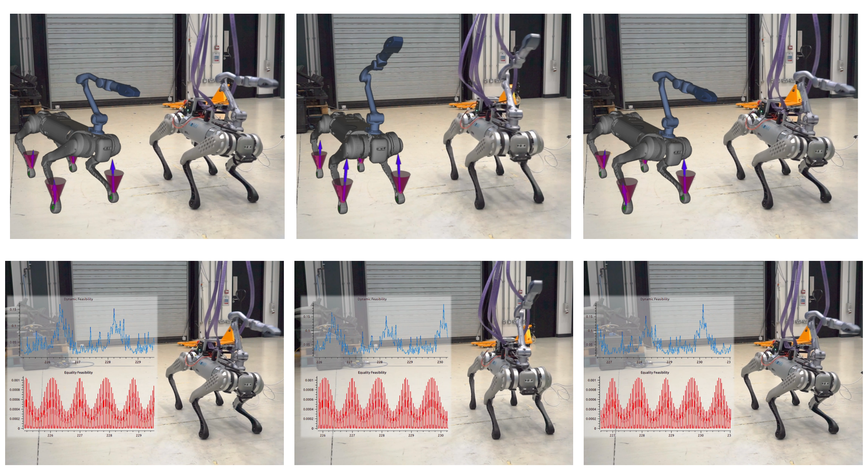
Gymnastic maneuvers

Also, our endpoint-explicit DDP algorithm **efficiently solved complex gymnastic maneuvers** for the Talos humanoid, achieving faster convergence, shorter runtimes, and better constraint satisfaction.



MPC trials

To demonstrate its relevance for control tasks, we experimentally validated our **MPC controller** using our endpoint-explicit method on scenarios like: a *B1 quadruped with a Z1 manipulator*.



We compared tracking error between the traditional cost-based penalty method and our approach with an explicit terminal constraint. In both the simulated and real-world scenarios, the **terminal constraint consistently improved tracking performance**.