AC4MPC: Actor–Critic Reinforcement Learning for Guiding Model Predictive Control

Combining globally-informed learning with locally-accurate optimization

Rudolf Reiter[1], Andrea Ghezzi, Katrin Baumgärtner, Jasper Hoffmann, Robert D. McAllister, Moritz Diehl

[1]Robotics and Perception Group, University of Zurich
Work done at: Systems Control and Optimization Laboratory, University of Freiburg
Published: IEEE Transactions on Control System Technologies, 2025

Talk roadmap (20 min)

1. Why combine MPC and RL?
2. AC4MPC: idea + formulation
3. Guarantees: what is (and isn't) promised
4. Real-time version: AC4MPC-RTI + evaluation of infeasible rollouts
5. Experiments: Snow-Hill + Autonomous Driving
6. Takeaways

Motivation: complementary strengths (and pain points)

MPC (online optimization)

- Handles constraints naturally
- High local accuracy, interpretable objective
- Challenges:
  - local minima (nonconvex NLP)
  - computation limits $\Rightarrow$ short horizons
  - terminal cost/constraints often hard to design (no steady state)

RL (actor–critic)

- Global exploration via interaction
- Learns *policy* (actor) and *value* (critic)
- Challenges:
  - limited accuracy / approximation error
  - safety + constraint satisfaction not guaranteed
  - training cost / sim-to-real concerns

Goal: Use RL to provide *global guidance* (warm-start + terminal value), while MPC refines actions *locally and constraint-aware*.

[1] Reiter et al. (2025). Synthesis of Model Predictive Control and Reinforcement Learning: Survey and Classification. arXiv:2502.02133.

Core idea: AC4MPC in one slide

What AC4MPC does
At each control step, solve a finite-horizon MPC problem where

- the actor $\hat{\pi}(s)$ provides a **rollout warm-start** (escape local minima / speed convergence),
- the critic $\hat{J}(s)$ (or $\hat{Q}(s, u)$) provides a **terminal cost** (approx. infinite horizon),
- optionally, an extra **actor rollout of length** R improves robustness to critic error.

Why the "A" and "C" both matter

- **Actor-only:** better initialization, but horizon still short-sighted.
- **Critic-only:** better terminal shaping, but solver may get trapped without good initial guess.
- **AC4MPC:** terminal cost approximation *and* better (local) optimum.

Problem setup and MPC objective (discounted)

Discrete-time deterministic dynamics: $s_{k+1} = F(s_k, u_k), \qquad c(s,u) \geq 0, \qquad \gamma \in (0,1]$.

Standard discounted MPC objective (horizon N):

$$V_N(s, \mathbf{u}) = \sum_{k=0}^{N-1} \gamma^k c(s_k, u_k) + \gamma^N V_f(s_N).$$

AC4MPC terminal cost with rollout R:

$$V_f(s) := \sum_{i=0}^{R-1} \gamma^i \, c(s_i, \hat{\pi}(s_i)) \; + \; \gamma^R \hat{J}(s_R), \quad s_{i+1} = F(s_i, \hat{\pi}(s_i)).$$

Resulting objective (free controls for $0..N-1$, then actor fixed):

$$V_{N,R}(s, \mathbf{u}) = \sum_{k=0}^{N-1} \gamma^k c(s_k, u_k) + \sum_{k=N}^{N+R-1} \gamma^k c(s_k, \hat{\pi}(s_k)) + \gamma^{N+R} \hat{J}(s_{N+R}).$$

Key knobs: N (optimization hor.) vs. R (mitigate critic errors without adding decision vars).

Closed-loop guarantee: what the theory says (high-level)

Critic assumptions (informal)
The critic approximately satisfies a one-sided Bellman inequality for the actor:

$$\gamma \hat{J}(F(s, \hat{\pi}(s))) \leq \hat{J}(s) - c(s, \hat{\pi}(s)) + \delta, \quad \hat{J}(s) \leq J_{\hat{\pi}}(s) + \varepsilon.$$

Performance bound (simplified)
For $\gamma \in (0, 1)$, the AC4MPC closed-loop cost is bounded by

$$\mathcal{J}^{\text{AC4MPC}}(s_0) \lesssim J_{\hat{\pi}}(s_0) - \underbrace{\sigma_{N,R}(s_0)}_{\text{MPC improvement over actor rollout}} + \gamma^{N+R}\varepsilon + \frac{\gamma^{N+R}}{1-\gamma}\delta.$$

Interpretation:

- If actor is suboptimal ($\sigma_{N,R} > 0$), AC4MPC can improve it.
- Critic errors enter as γ^{N+R}: longer N and/or R suppress them.
- Guarantee does *not* require global optimality in the real-time (suboptimal) version.

From concept to real-time: AC4MPC-RTI

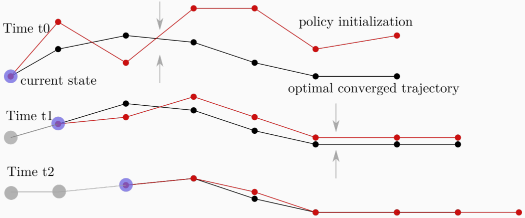


Figure 1: One time initialization converges over iterations.

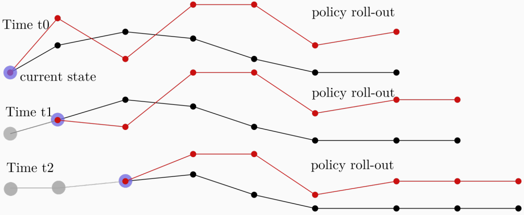


Figure 2: Reinitialization at each time step does not converge over iterations.

Problem: RTI (one/few SQP steps per closed-loop iter.) means trajectories are *not fully converged* but re-initialized with policy at each time step.

From concept to real-time: AC4MPC-RTI overview

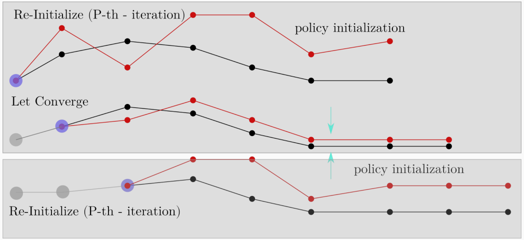


Figure 3: RTI (one/few SQP steps per closed-loop iter.) means trajectories are *not fully converged*.

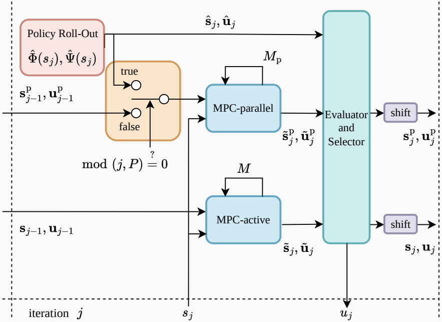


Figure 4: AC4MPC-RTI: policy rollout (red), periodic re-init of parallel MPC (yellow), trajectory evaluation (green), shift update (purple).

Why this is needed: RTI (one/few SQP steps per closed-loop iter.) means trajectories are *not fully converged*.

AC4MPC-RTI mechanics: parallelization + selection

Two MPC instances + actor rollout:

- **Active MPC:** warm-start from shifted previous best trajectory
- **Parallel MPC:** reinitialized every P steps with actor rollout
- **Actor rollout:** always available candidate

Selection rule (each step): choose the candidate with lowest predicted cost (via `ac4eval`).

Effect: preserves RTI's "tracking a local optimum" *but* periodically injects a globally-informed initialization to escape poor basins.

Key parameters

- P: re-init period (exploration frequency)
- M, M_p: SQP steps per tick (active / parallel)
- R: evaluation rollout length
- $\alpha \in [0, 1]$: feasibility correction strength

Evaluating infeasible multiple-shooting trajectories: `ac4eval`

Issue: RTI + multiple shooting $\Rightarrow$ candidate $(\mathbf{s}, \mathbf{u})$ may violate dynamics at shooting nodes.
Fix: simulate a corrected rollout:

$$\bar{u}_k = u_k + \alpha\big(\hat{\pi}(\bar{s}_k) - \hat{\pi}(s_k)\big), \qquad \bar{s}_{k+1} = F(\bar{s}_k, \bar{u}_k).$$

- $\alpha = 0$: pure open-loop forward sim
- $\alpha = 1$: full actor-based correction (close gaps)

Then append an actor rollout of length R and add critic at the end.

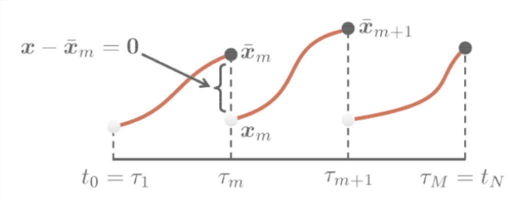


Figure 5: Cost evaluation with multiple shooting requires feasibility projection.

Dynamics: 1D point mass with position p, velocity v:

$$\dot{p} = v, \qquad \dot{v} = u + a_{\text{res}}(p), \qquad |u| \le 1.$$

Key property: must first move away to gain speed, otherwise gets stuck.

Purpose of this experiment:

- show local minima in short-horizon MPC
- show why actor+critic together matters
- compare to DP baseline in low dimension

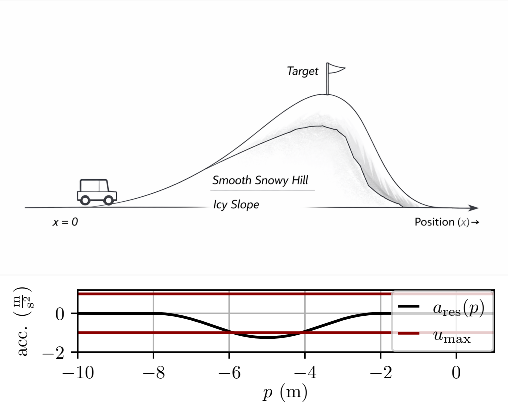
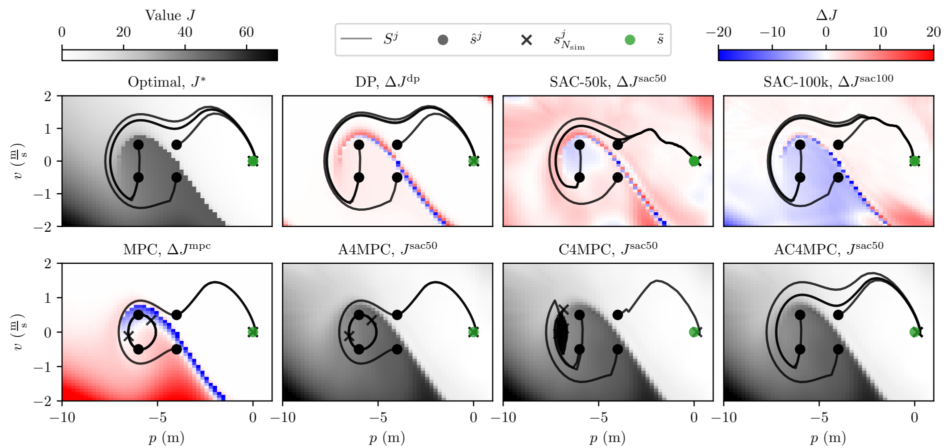


Figure 6: Snowy slope disturbance and control bounds.

Snow-Hill: qualitative behavior (value/cost maps)

Figure 7: Value fun. and traj.: nominal MPC suffers local minima; AC4MPC escapes via warm-start + terminal shaping.



Takeaway: In a nonconvex landscape, better terminal shaping *and* a better initial guess are both needed to consistently reach good solutions.

Snow-Hill: RTI behavior

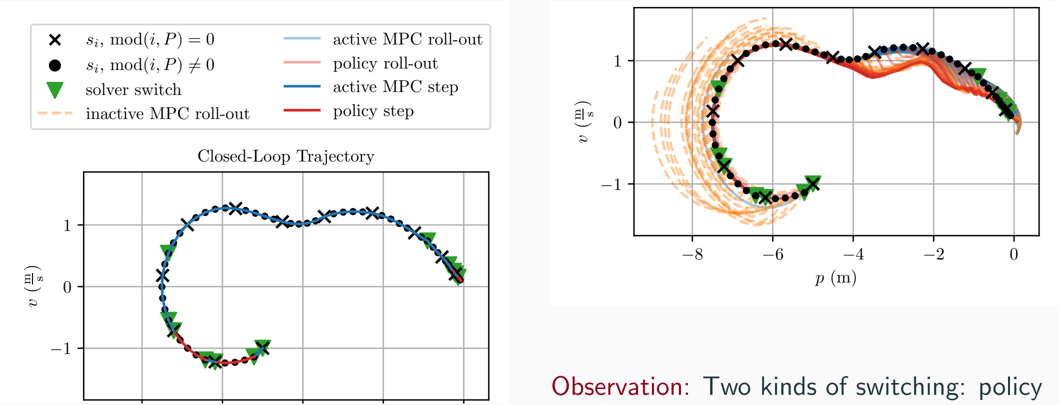


Figure 8: AC4MPC-RTI switches between candidates; parallel MPC reinit every P steps.

Observation: Two kinds of switching: policy step vs. MPC step and initialization within MPC.

Snow-Hill: quantitative results + RTI behavior

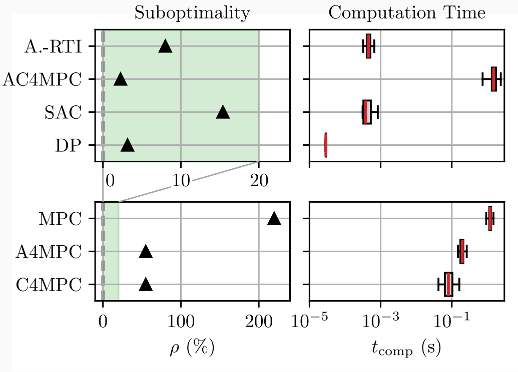


Figure 9: Closed-loop cost + compute time: AC4MPC best cost; AC4MPC-RTI near-best at much lower compute.

Takeaway: RTI variant keeps most of the performance gain while enabling real-time operation.

Snow-Hill: robustness to model mismatch

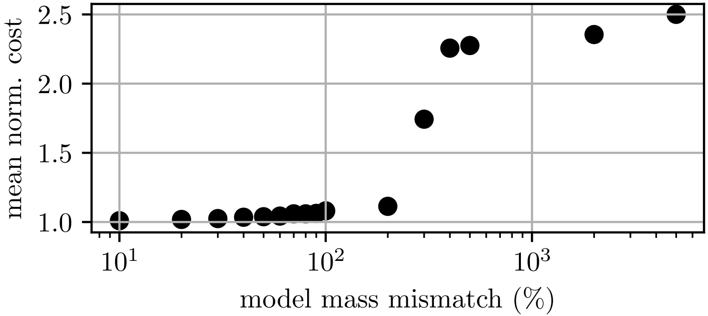


Figure 10: Mean performance vs. model-plant mismatch (mass scaling). AC4MPC tolerates moderate mismatch before degrading.

Message: benefits rely on a reasonably accurate model (as with MPC), but AC4MPC inherits MPC-like robustness for moderate mismatch.

Experiment 2: Autonomous driving overtaking (setup & result)

Task: overtake slower vehicles on randomized road curvature, with constraints:

- speed limit, accel limits
- collision avoidance via obstacle constraints
- no meaningful steady state $\Rightarrow$ terminal design is hard

Observation from paper:

- large-horizon MPC becomes more sensitive to initialization (local minima)
- AC4MPC-RTI leverages actor to escape, critic as terminal shaping

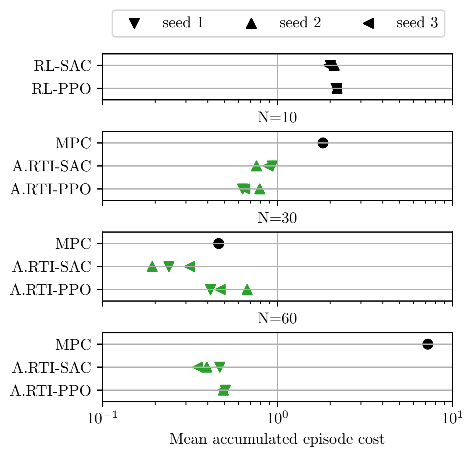


Figure 11: Mean episode cost vs. horizon N: AC4MPC-RTI improves over MPC and RL across horizons.

Autonomous driving: what “escaping a local minimum” looks like

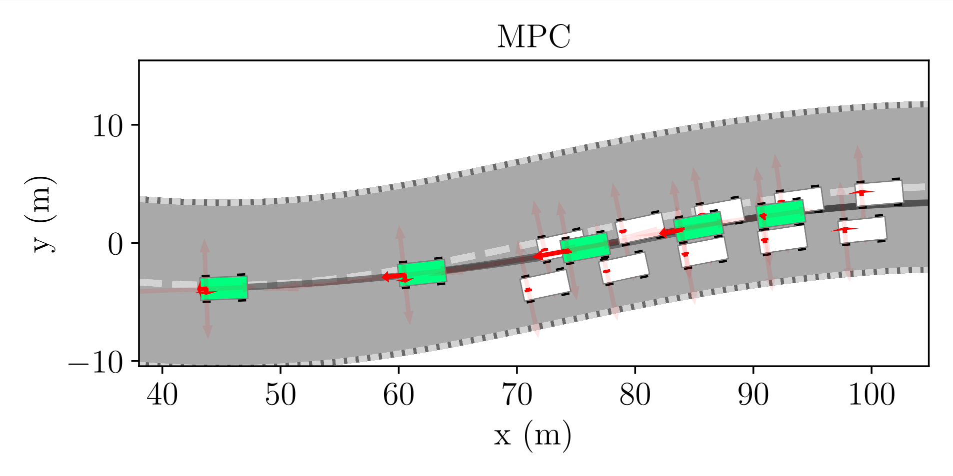


Figure 12: Snapshots: MPC may get stuck.

Autonomous driving: what “escaping a local minimum” looks like

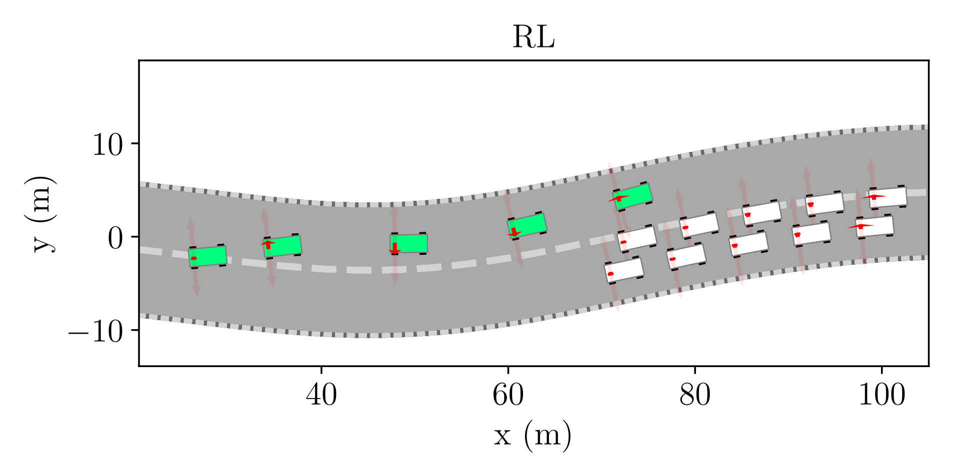


Figure 13: Snapshots: RL is conservative.

Autonomous driving: what “escaping a local minimum” looks like

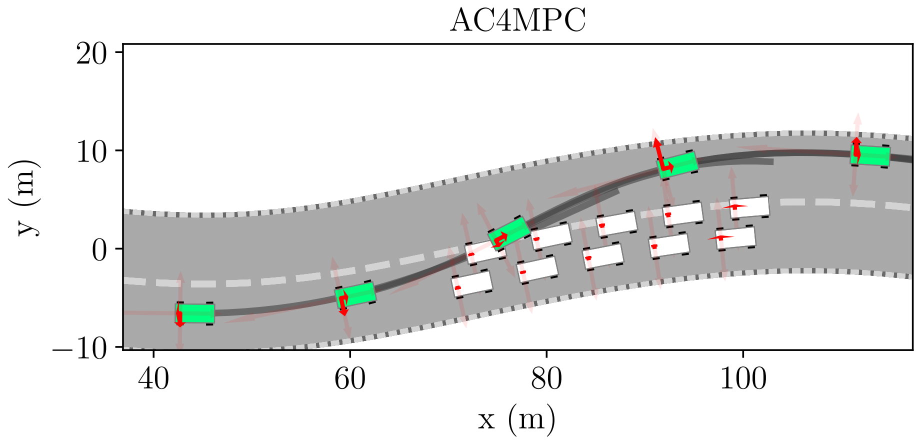


Figure 14: Snapshots: AC4MPC-RTI uses policy rollout + terminal guidance to pass.

Practical notes (what you'd tell someone implementing this)

Numerics that matter (from the paper)

- Use smooth networks (e.g., `tanh`); ReLU harms SQP smoothness.
- Critic inside MPC can destabilize solver: scale it (paper used a factor β in AD).
- Prefer Gauss–Newton / first-order handling for NN terminal term if needed.

Tuning intuition

- increase N when actor is clearly suboptimal (more room to improve)
- increase R when critic is noisy (mitigate terminal error cheaply)
- increase α when gaps/infeasibility are significant (more correction in evaluation)
- decrease P to inject actor warm-starts more frequently (more "globalization")

Conclusions (what to remember)

Main takeaways

- AC4MPC combines: **actor warm-start** + **critic terminal shaping** + **selection among candidates**.
- Theoretical bounds explain the trade-off:

$$\text{gain over actor} \approx \sigma_{N,R} \quad \text{vs. critic errors suppressed by } \gamma^{N+R}.$$

- AC4MPC-RTI makes it practical: parallel RTI + evaluation of infeasible rollouts.
- Empirically: avoids local minima where long-horizon MPC struggles; improves cost vs RL with modest overhead.

Questions?