Michael Psenka[1] · Mike Rabbat[2] · Aditi Krishnapriyan[1] · Yann LeCun[2,3]* · Amir Bar[2]*

[1]UC Berkeley · [2]Meta AI (FAIR) · [3]NYU *equal advisorship

PARALLEL STOCHASTIC GRADIENT-BASED PLANNING

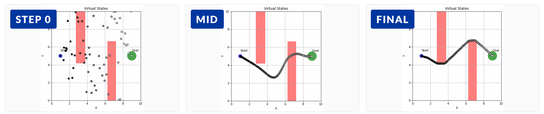
GRASP: Gradient-Based Planning for World Models at Long Horizons

Gradient **R**el**A**xed **S**tochastic **P**lanner: a first-order, lifted-state planner that only backpropagates through *action* gradients, not brittle state gradients.

TL;DR

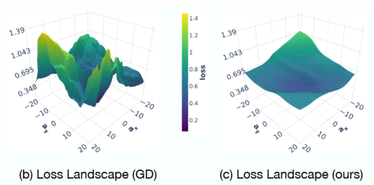
GRASP.

A gradient-based planner for learned world models that makes long-horizon planning practical by **(1) lifting trajectories into virtual states** for parallel-in-time optimization, **(2) injecting noise into state iterates** for exploration, and **(3) reshaping gradients** so only action signals flow, sidestepping the adversarial fragility of state-input gradients in deep world models.



BallNav · collocation exploration. *Scattered virtual states collapse onto a feasible trajectory that routes around obstacles: possible because states may temporarily pass through unphysical regions.*

Push-T · long-horizon planning. *GRASP remains tractable at H = 60–80, where serial gradient descent collapses.*



Naive GD's loss surface (left) is rough and adversarial; GRASP's reshaped landscape (right) is smooth and well-conditioned — see Method.

01 · PROBLEM

Why long-horizon planning is hard

Given start s_0 and goal g, the naïve planner rolls the world model forward and minimizes terminal error:

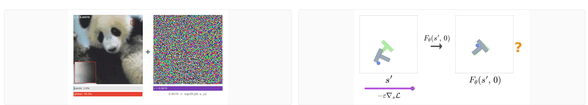
$$min_{\boldsymbol{a}} \| s_T(\boldsymbol{a}) - g \|^2$$

At short horizons, this works. As T grows, two central failure modes compound:

① **Exploding / vanishing gradients.** Backprop through T compositions of F_θ gives a product of Jacobians whose conditioning scales as $\sigma(D_S F)^{T-1}$.

Similarity-loss along the optimal path is non-monotonic; the GD loss surface is rough.

② **Adversarial state gradients.** Because training trajectories lie on a low-dim manifold $\mathcal{M}_S$, F_θ has high Lipschitz in normal directions: it's trivial to find adversarial states that fool the model into thinking it reached the goal.



***Left:** classic adversarial example — a small, normal-direction perturbation changes a classifier's output entirely. **Right:** the same fragility inside a world model — base points can be nudged to pretend the local goal was already reached.*

02 · METHOD

GRASP, in three ingredients

Start from the **collocation** objective: lift states into variables, relax dynamics into a soft penalty. Every term is local, so all T evaluate in parallel: long, non-greedy detours (going around walls, backing up) no longer require a deep serial rollout to discover, and gradients no longer flow through a deep T-step composition. The action space still grows as $T\dim(\mathcal{A})$, but the optimizer now explores it all at once.

INGREDIENT 1

Noise the state iterates

Inject Gaussian noise into virtual state updates so the optimizer hops between basins; keep actions on pure gradient descent.

$$s_t \leftarrow s_t - \eta_s \nabla_s \mathscr{L} + \sigma \cdot \xi$$

INGREDIENT 2

Reshape gradients

Stop gradients into the *state* input of F_θ (the fragile pathway). Actions still receive clean signal. Add a **dense goal term** to prevent trivial minima.

$D_s F_\theta$ ✗ → $D_a F_\theta$ ✓

$$\mathscr{L}(\boldsymbol{s},\boldsymbol{a}) = \Sigma_t \| F_\theta(s_t, a_t) - s_{t+1} \|^2_2 + \gamma \cdot \Sigma_t \| F_\theta(s_t, a_t) - g \|^2_2$$

INGREDIENT 3

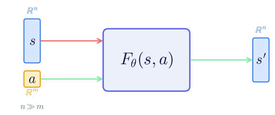
Periodic sync to true rollouts

Every K_{sync} iterations, refine on the original serial objective to ground lifted states onto feasible trajectories.

03 · WHY IT WORKS

Action gradients are trustworthy; state gradients aren't

The action Jacobian $D_a F_\theta$ is well-behaved because the action space is **low-dimensional and exhaustively covered** during training. GRASP is built to use only that pathway.



Why: actions $a \in \mathbb{R}^m$ are low-dimensional and densely trained; states $s \in \mathbb{R}^n$ aren't ($n \gg m$). GRASP only trusts gradients through the low-dim path.

04 · NEXT

What's next

- **Diffusion world models.** Deeper latent timesteps ≈ smoothed versions of F_θ itself: a natural extension of the noising ingredient.
- **Closed-loop and RL.** Fold GRASP into adaptive replanning and policy-learning loops.
- **Better optimizers.** Momentum- and second-order-aware variants of the lifted objective.

05 · RESULTS

Higher success, shorter wall-clock

***Left:** Push-T success % / median time-to-success. **Right:** success % across world-model environments at fixed planning horizon. **Bold = best in row.** *GRASP, † CEM sync instead of GD, ‡ with noise & goal-coefficient annealing.*

PUSH-T · LONG-HORIZON PLANNING

Horizon	CEM	GD	LatCo	GRASP
H = 40	**61.4%** / 35.3s	51.0% / 18.0s	15.0% / 598.0s	59.0% / **8.5s**
H = 50	30.2% / 96.2s	37.6% / 76.3s	4.2% / 1114.7s	**43.4% / 15.2s**
H = 60	7.2% / 83.1s	16.4% / 146.5s	2.0% / 231.5s	**26.2% / 49.1s**
H = 70	7.8% / 156.1s	12.0% / 103.1s	0.0% / n/a	**16.0% / 79.9s**
H = 80	2.8% / 132.2s	6.4% / 161.3s	0.0% / n/a	**10.4% / 58.9s**

OTHER ENVIRONMENTS

Environment	Horizon	CEM	GD	GRASP
Reacher (le-wm)	15 (75)	38%	42%	**70%***
Push-T (le-wm)	15 (75)	22%	26%	**46%*†**
Cube (le-wm)	10 (50)	58%	56%	**66%*‡**
Cube (le-wm)	15 (75)	64%	50%	**68%*‡**
MW Reach (jepa-wm)	6 (30)	12%	6%	**19%*‡**

SCAN FOR PAPER

Paper & code

michaelpsenka.io/grasp