

# **MOTION PLANNING DIFFUSION: LEARNING AND ADAPTING ROBOT MOTION PLANNING WITH DIFFUSION MODELS**

J. Carvalho, A. Le, P. Kicki, D. Koert, J. Peters

---

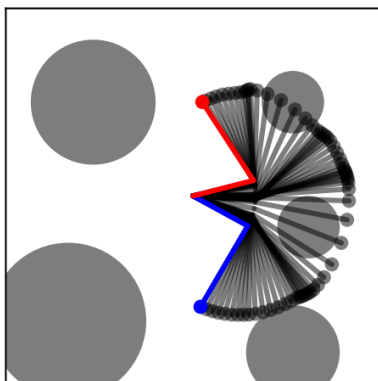
Presented By: Aayush Fadia

- Diffusion for optimization-based planning.
  - Prior trajectory distribution learned from dataset.
  - Adaptation to specific environment done at planning time.

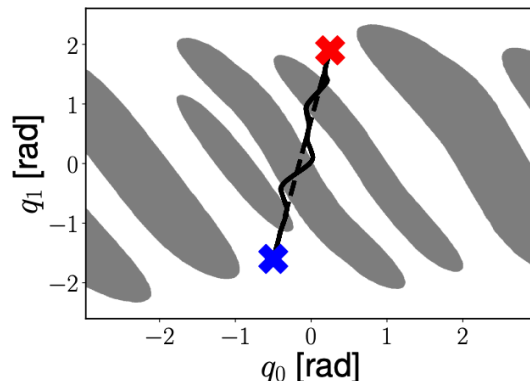
## **Publication**

- Motion Planning Diffusion: Learning and Planning of Robot Motions with Diffusion Models - Published in IROS 2023:
  - 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
- Updated Work (This): Motion Planning Diffusion: Learning and Adapting Robot Motion Planning with Diffusion Models:
  - IEEE TRANSACTIONS ON ROBOTICS, VOL. 41, 2025

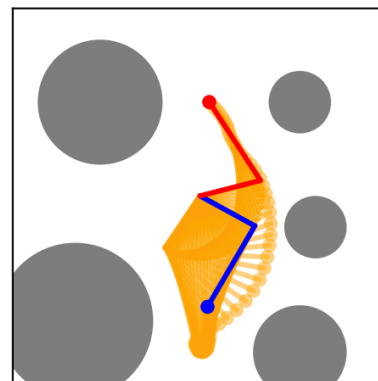
- Optimization-based motion planning:
  - CHOMP and family: Initial trajectory (straight-line or sampling planner) optimized using cost function gradient for smoothness.
- Diffusion Policy:
  - Not a planner - it sees observations and provides a small number of actions to be executed in the near term only.



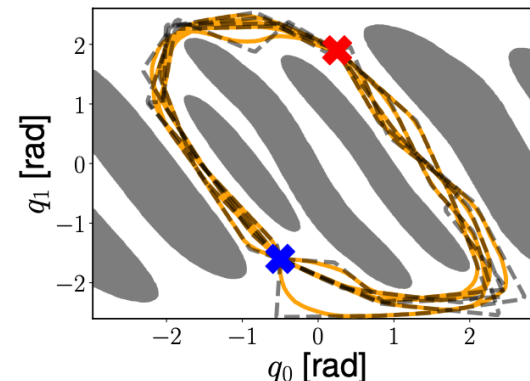
(a) CHOMP (task space)



(b) CHOMP (joint space)



(c) RRT-Connect + CHOMP (task space)



(d) RRT-Connect + CHOMP (joint space)

- **Aim/Motivation**

- Need to encode prior trajectories in a multimodal manner.
- Need to generalize outside the training environment.
- Both at the same time.

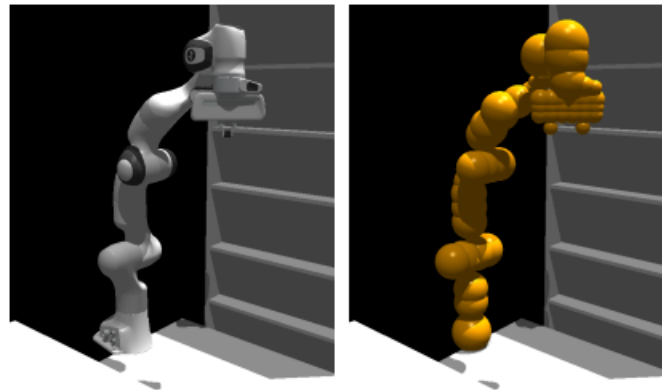
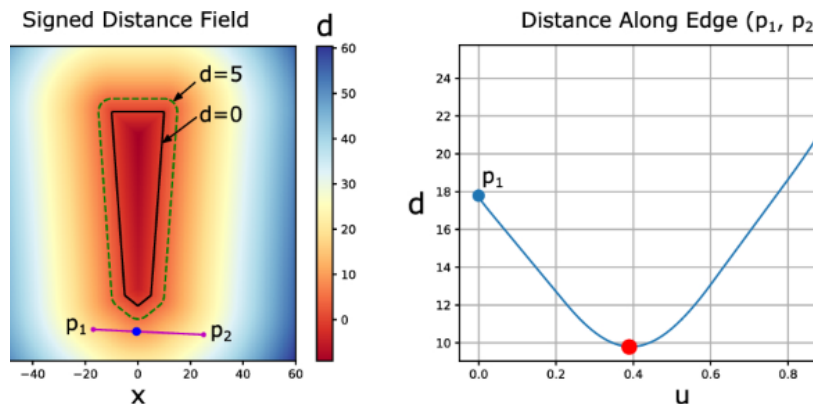
- **Assumptions**

- Fixed-time trajectory planning - no attempt to generate fast trajectories.

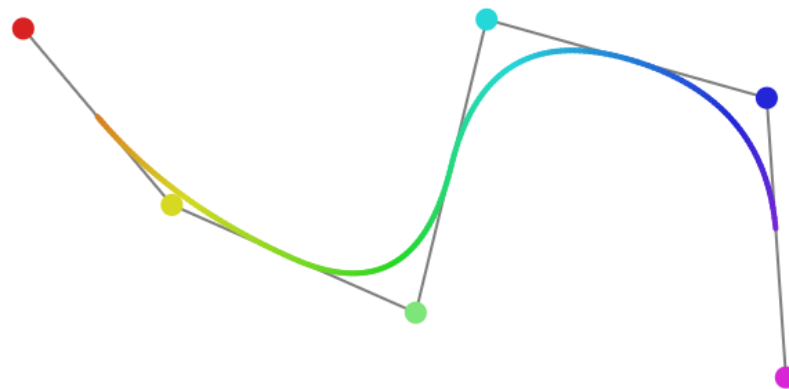
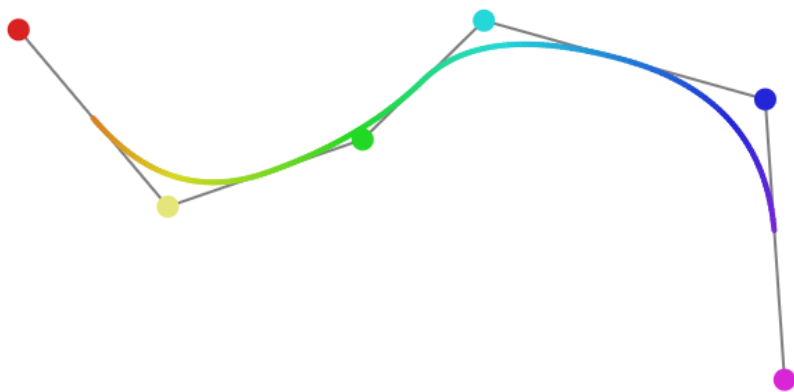
# Core concepts

- Difficult to optimize against boolean (Y/N) constraints:
  - Is colliding? Is within limits?
- Need to make constraints soft  $C(\tau) \in \mathbb{R}$ , real and differentiable (less is better).
- Represent total cost as  $C(\tau) = \sum_j \lambda_j C_j(\tau)$
- Find trajectory  $\tau = \arg \min_{\tau} (C(\tau))$

- Robot  $\approx S$  spheres on, radii  $r_m$  and centres  $x_m$  in configuration  $q_t$ .
- $$C_{\text{env}} = \sum_{m=0}^{S-1} \text{ReLU}(-\text{SDF}(x_m) + r_m + \varepsilon)$$
  - $>0$  and high when in collision,  $=0$  when not in collision.
- $$C_{\text{self}} = \max_{i,j \in S_c} (\text{ReLU}(-\|x_i - x_j\|) + r_i + r_j + \varepsilon)$$
- $C_{\text{coll}} = C_{\text{env}} + C_{\text{self}}$  differentiable w.r.t trajectory:
  - Can find  $\nabla_{x_m} C_{\text{coll}}$  from above and  $\nabla_q x_m$  from jacobian.
  - Chain rule!



- Trajectory (**joint space**) is parameterized by **B-Splines**, not waypoints.
- Fewer parameters, smooth by definition, locality (changing a control point changes only neighboring segments).
- Differentiable - can calculate  $\nabla_b q_t$ , where  $b$  are B-spline parameters, and  $q_t$  is robot configuration at time  $t$ .



# The Idea

Quality	Image	Trajectory
<b>Objective</b>	Give me a picture of a cat	Plan trajectory from $q_0$ to $g$ ( $q_g$ or $H_g$ )
<b>Objective (math)</b>	Sample image $X \in \mathbb{R}^{h \cdot w \cdot c} \sim p(X   \text{"cat"})$ - training distribution of images on the internet	Sample parameters $X \in \mathbb{R}^{n \cdot d} \sim p(X   q_0, g)$ - training distribution of solved trajectories
<b>Process</b>	<pre> 1: <math>\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})</math> 2: <b>for</b> <math>t = T, \dots, 1</math> <b>do</b> 3:   <math>\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})</math> if <math>t &gt; 1</math>, else <math>\mathbf{z} = \mathbf{0}</math> 4:   <math>\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\alpha_t}} \epsilon_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}</math> 5: <b>end for</b> 6: <b>return</b> <math>\mathbf{x}_0</math> </pre>	+ conditioning("cat", $q_0, g$ ) added to noise model $\epsilon_0$

- In this scheme, all the conditioning ("cat",  $g$ ) is present at training time.
- For planning environment information (cost function) is available **only** at inference time.
- How do we change the regime outlined above to achieve this?

- At inference time, at every denoising step, we:
  - Update the mean using the diffusion model.
  - Update this new mean using gradient descent to minimize the cost function.
- In english:
  - Take a diffusion step, i.e. make the trajectory look like the ones in the training data (for this  $q_0, g$ ).
  - Shift the trajectory a little to minimize the environment-dependent cost function (i.e. don't collide in this current planning environment).
- This is a principled approach i.e. it provably samples from the optimal distribution (proof left out for brevity).

**Input:** Pre-trained denoising model  $\epsilon_\theta$ , noise schedule terms  $(\alpha_i, \bar{\alpha}_i, \sigma_i)$ , start joint position  $\mathbf{q}_{\text{start}}$  and goal end-effector pose  ${}^W \mathbf{H}_{\text{goal}}^{EE}$ , B-spline basis matrix  $\mathbf{B}$ , motion planning costs  $C_j$  and temperatures  $\lambda_j$ , prior temperature  $\lambda_{\text{prior}}$ , cost gradient start index  $i_{\text{cost}}$ , inner gradient parameters (steps  $M$ , step size  $\gamma$ , maximum step  $\delta$ )

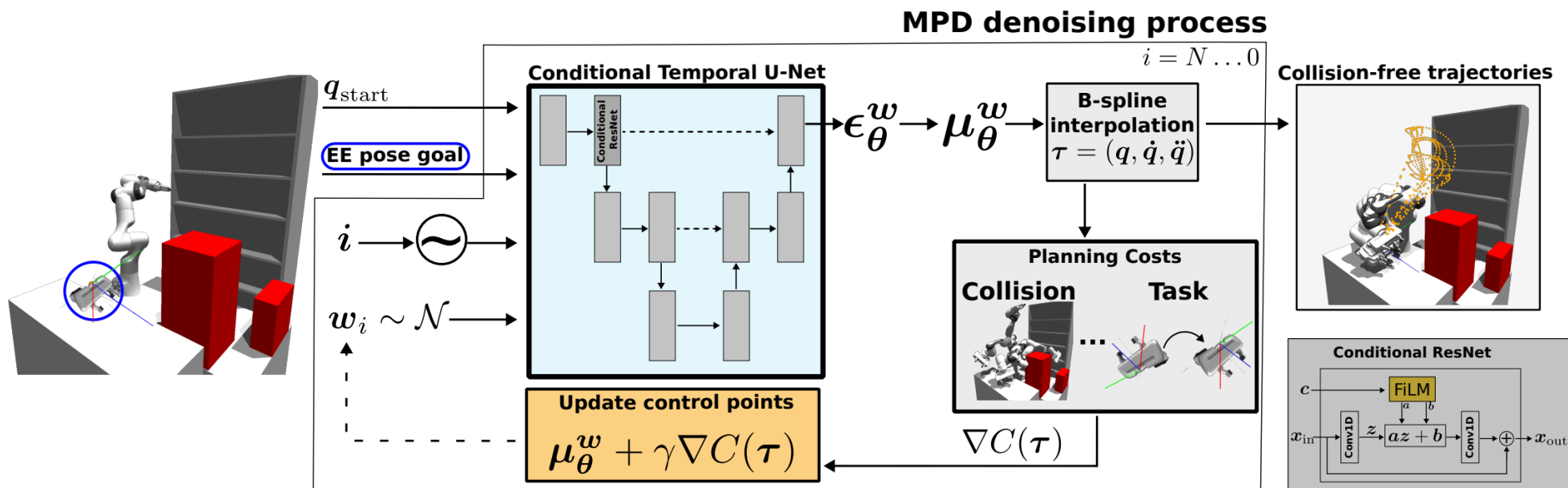
## Diffusion (DDPM) Algorithm

## Cost Guidance (MPD Addition)

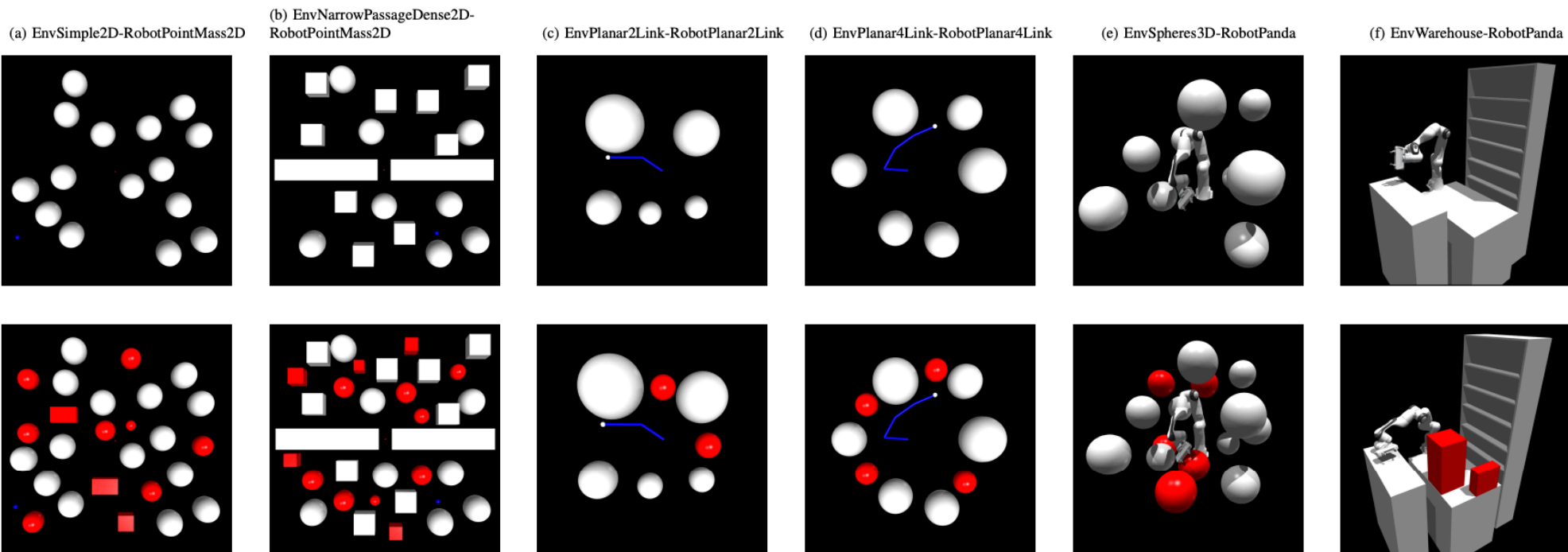
- Cost-based guidance skipped initially when trajectory is noisy.
- $M$  gradient descent steps taken, cost-guidance update clipped to  $\delta$  to prevent OOD.
- Diffusion update weighted by  $\lambda_{\text{prior}}$  to prioritize cost-guidance.

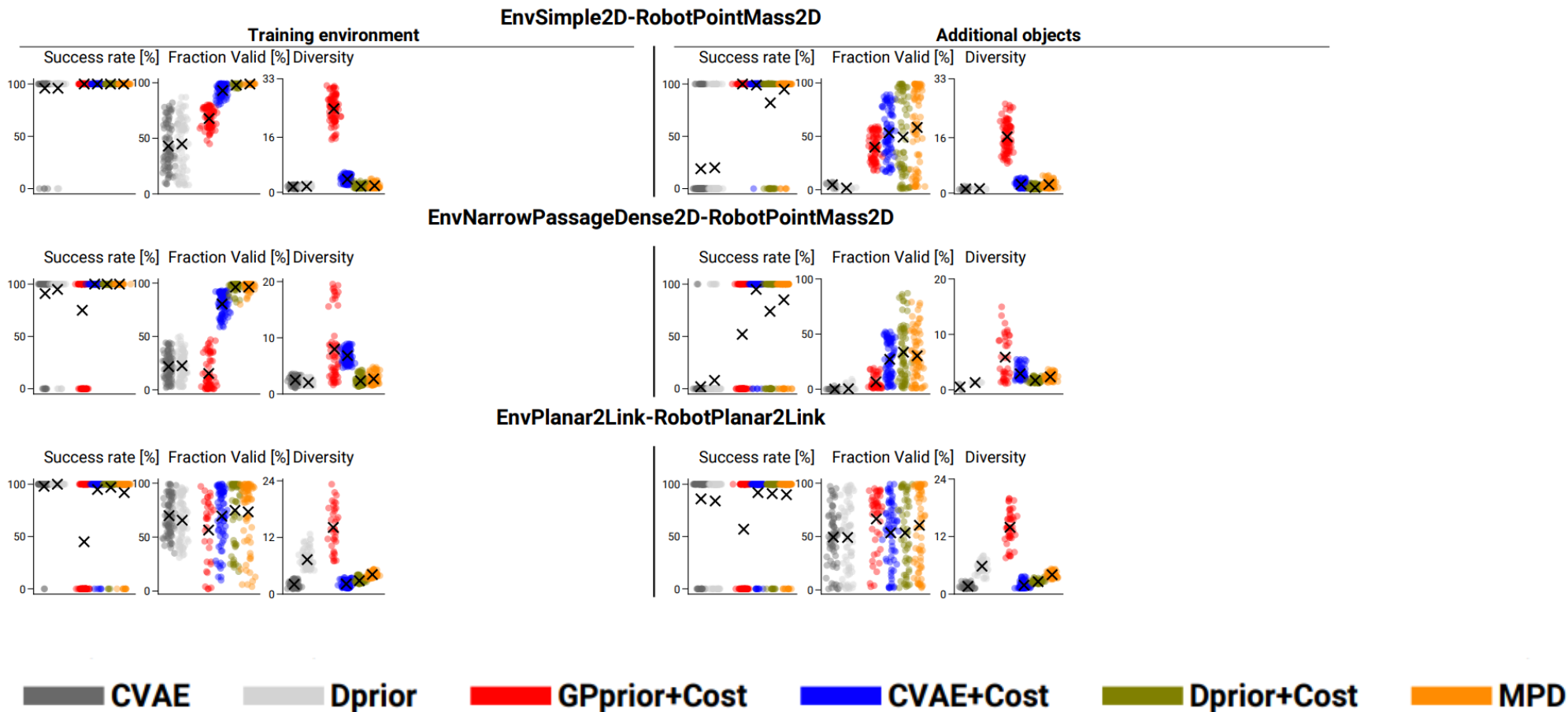
```

 $\mathbf{c} = [\mathbf{q}_{\text{start}}, {}^W \mathbf{H}_{\text{goal}}^{EE}]$  ▷ build the conditioning variable
 $\mathbf{w}_N \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$  ▷ sample noisy control points
for  $i = N, \dots, 1$  do
    if  $i > i_{\text{cost}}$  then  $\lambda_{\text{prior}} = 1$ 
        ▷ compute the diffusion prior mean eq. (6)
         $\boldsymbol{\mu}_i(\mathbf{w}_i, i, \mathbf{c}) = \frac{1}{\sqrt{\alpha_i}} \left( \mathbf{w}_i - \frac{1 - \alpha_i}{\sqrt{1 - \bar{\alpha}_i}} \lambda_{\text{prior}} \epsilon_\theta(\mathbf{w}_i, i, \mathbf{c}) \right)$ 
    if  $i < i_{\text{cost}}$  then
        ▷ inner gradient steps eq. (15)
         $\boldsymbol{\mu}_i^0 = \boldsymbol{\mu}_i$ 
        for  $k = 1, \dots, M$  do
            ▷ compute B-spline trajectories eqs. (19), (24) and (25)
             $\boldsymbol{\tau} = (\mathbf{q}, \mathbf{q}', \mathbf{q}'') = (\mathbf{B}\boldsymbol{\mu}_i^k, \mathbf{B}'\boldsymbol{\mu}_i^k, \mathbf{B}''\boldsymbol{\mu}_i^k)$ 
            ▷ compute costs gradient eqs. (27) and (28)
             $\mathbf{g} = -\sum_j \lambda_j \nabla_{\boldsymbol{\mu}_i^k} C_j(\boldsymbol{\tau})$ 
            ▷ Clip and apply the gradient
             $\boldsymbol{\mu}_i^k = \boldsymbol{\mu}_i^{k-1} + \gamma \mathbf{g}$ 
             $\Delta \boldsymbol{\mu} = \text{clip}(|\boldsymbol{\mu}_i^k - \boldsymbol{\mu}_i^0|, -\delta, \delta)$ 
             $\boldsymbol{\mu}_i^k = \boldsymbol{\mu}_i^0 + \Delta \boldsymbol{\mu}$ 
        else
             $\boldsymbol{\mu}_i^M = \boldsymbol{\mu}_i$ 
            ▷ sample from the posterior distribution
             $\mathbf{w}_{i-1} = \boldsymbol{\mu}_i^M + \boldsymbol{\Sigma}_i \mathbf{z}, \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
            ▷ compute B-spline trajectories
             $\boldsymbol{\tau}_0 = (\mathbf{q}_0, \mathbf{q}'_0, \mathbf{q}''_0) = (\mathbf{B}\mathbf{w}_0, \mathbf{B}'\mathbf{w}_0, \mathbf{B}''\mathbf{w}_0)$ 
Output: batch of trajectories  $\boldsymbol{\tau}_0$ 
    
```



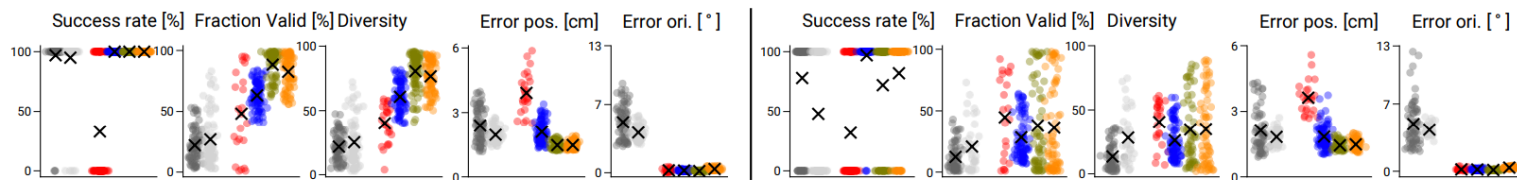
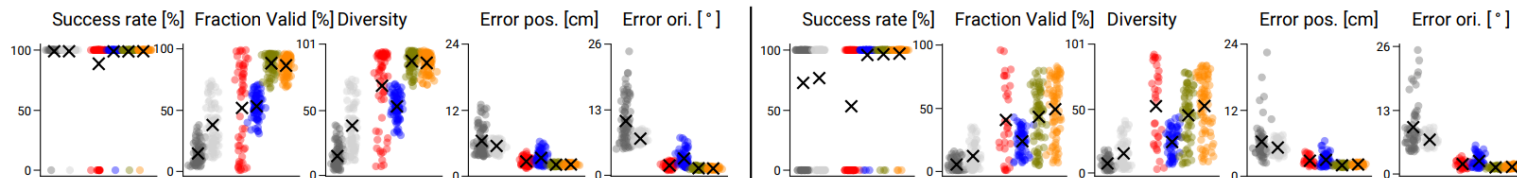
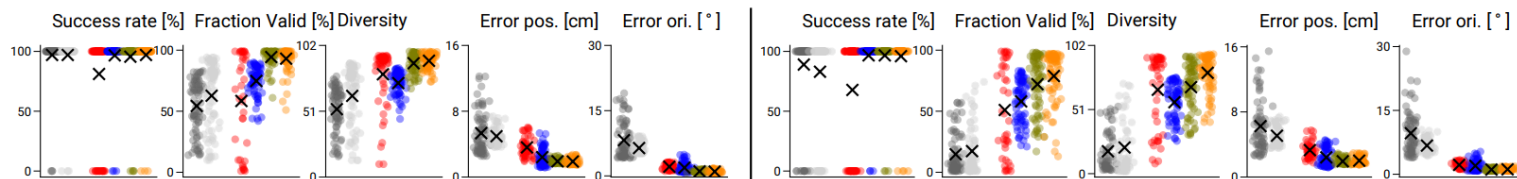
# Results





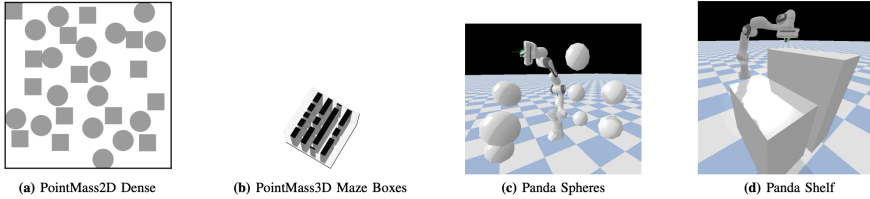
Training environment

Additional objects

**EnvPlanar4Link-RobotPlanar4Link**

**EnvSpheres3D-RobotPanda**

**EnvWarehouse-RobotPanda**


■ CVAE  
 ■ Dprior  
 ■ GPprior+Cost  
 ■ CVAE+Cost  
 ■ Dprior+Cost  
 ■ MPD

“When comparing CVAE and Dprior, their success rate and validity fraction are close, but the **diversity score is higher for the diffusion-based models**, in particular for higher-dimensional environments (using RobotPanda). For instance, in the EnvWarehouse-RobotPanda, both CVAE and Dprior achieve 97% mean success rate, but Dprior shows more mean variability (50.9 vs. 60.3). **Moreover, Dprior achieves smaller mean errors in the desired end-effector goal position and orientation** (7.4cm vs. 5.2cm, 10.5deg vs. 6.7 deg), which due to model approximations are not 0. These errors are improved during cost optimization.”



**Fig. 2:** The environments considered in our experiments include robot navigation tasks of a point mass in 2D and 3D, and a 7-dof Franka Emika Panda manipulator. In (a) and (b), the green and red dots are the initial and goal states, respectively, and the blue line is the result of RRT Connect.

**TABLE I:** Motion planning generation benchmarks in the environments of the training set, and additional environments with extra obstacles.  $\uparrow$  means higher is better.  $\downarrow$  means lower is better. Values highlighted in bold in the main text. Legend: RRTC – RRTConnect, CPrior – CVAE Prior, DPrior – Diffusion Prior, CPost – CVAE Posterior Optimization, MPD – Motion Planning Diffusion

	PointMass2D Dense					PointMass2D Dense - Extra Obstacles				
	T[s] $\downarrow$	S[%] $\uparrow$	I[%] $\downarrow$	PL $\downarrow$	VAR $\uparrow$	T[s] $\downarrow$	S[%] $\uparrow$	I[%] $\downarrow$	PL $\downarrow$	VAR $\uparrow$
RRTC	4.7 $\pm$ 2.5	100.0 $\pm$ 0.0	.0 $\pm$ 0.0	1.9 $\pm$ .5		8.0 $\pm$ 3.5	100.0 $\pm$ 0.0	.0 $\pm$ 0.0	2.2 $\pm$ .6	
CPrior	.02 $\pm$ .2	<b>46.0</b> $\pm$ 49.5	11.8 $\pm$ 12.5	1.5 $\pm$ .4	.0 $\pm$ 0.0	.02 $\pm$ .2	<b>14.0</b> $\pm$ 34.7	24.3 $\pm$ 14.7	1.6 $\pm$ .4	.0 $\pm$ 0.0
DPrior	.3 $\pm$ 0.0	<b>95.0</b> $\pm$ 14.0	5.5 $\pm$ 05.2	1.6 $\pm$ .5	<b>1.3</b> $\pm$ 1.0	.3 $\pm$ 0.0	<b>60.0</b> $\pm$ 49.0	16.0 $\pm$ 10.5	1.5 $\pm$ .5	1.3 $\pm$ 1.0
GPMP	26.5 $\pm$ 2	<b>52.0</b> $\pm$ 49.7	1.7 $\pm$ 1.3	1.4 $\pm$ .4	.03 $\pm$ .03	26.5 $\pm$ 2	<b>49.0</b> $\pm$ 50.0	1.1 $\pm$ 1.5	1.4 $\pm$ .4	.03 $\pm$ .04
RRTC-GPMP	38.43 $\pm$ 30.9	100.0 $\pm$ 0.0	.0 $\pm$ 0.0	1.94 $\pm$ .5	2.2 $\pm$ .9	42.5 $\pm$ 7.2	100.0 $\pm$ 0.0	0.0 $\pm$ 0.1	2.0 $\pm$ .5	2.7 $\pm$ .9
CPrior-GPMP	25.3 $\pm$ 2	71.0 $\pm$ 45.4	0.9 $\pm$ 01.8	1.4 $\pm$ .5	.01 $\pm$ .02	26.0 $\pm$ 2	53.0 $\pm$ 49.9	01.1 $\pm$ 01.7	1.5 $\pm$ .5	.02 $\pm$ .03
DPrior-GPMP	26.1 $\pm$ .3	<b>99.0</b> $\pm$ 10.0	.5 $\pm$ .6	1.4 $\pm$ 1.1	1.6 $\pm$ .5	27.0 $\pm$ .3	<b>92.0</b> $\pm$ 27.1	.7 $\pm$ .8	1.6 $\pm$ .5	1.4 $\pm$ .9
CPost	.1 $\pm$ .1	88.0 $\pm$ 32.5	.3 $\pm$ .7	1.5 $\pm$ .4	.01 $\pm$ .02	.1 $\pm$ .1	<b>78.0</b> $\pm$ 41.4	.5 $\pm$ .7	1.62 $\pm$ .5	.01 $\pm$ .02
MPD	.3 $\pm$ 0.0	99.0 $\pm$ 10.0	.6 $\pm$ 1.2	1.7 $\pm$ .5	1.4 $\pm$ 1.0	.3 $\pm$ 0.0	<b>79.0</b> $\pm$ 40.7	10.3 $\pm$ 8.5	1.7 $\pm$ .4	<b>1.4</b> $\pm$ 1.0

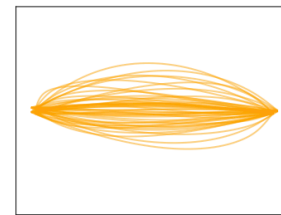
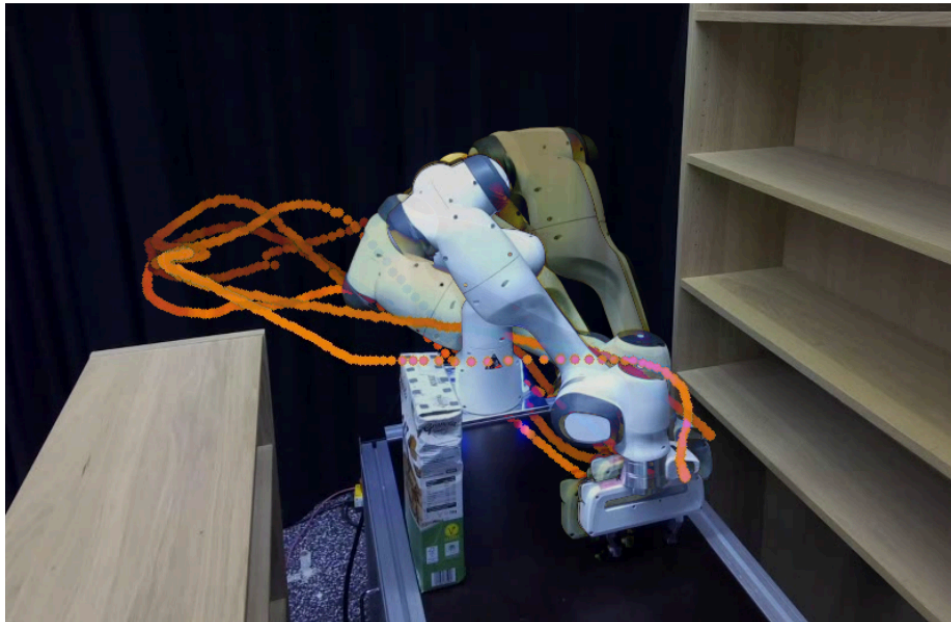
	PointMass3D Maze Boxes					PointMass3D Maze Boxes - Extra Obstacles				
	T[s] $\downarrow$	S[%] $\uparrow$	I[%] $\downarrow$	PL $\downarrow$	VAR $\uparrow$	T[s] $\downarrow$	S[%] $\uparrow$	I[%] $\downarrow$	PL $\downarrow$	VAR $\uparrow$
RRTC	27.4 $\pm$ 26.1	100.0 $\pm$ 0.0	.0 $\pm$ 0.0	3.7 $\pm$ 1.8		32.0 $\pm$ 27.4	100.0 $\pm$ 0.0	.0 $\pm$ 0.0	4.0 $\pm$ 1.7	
CPrior	.02 $\pm$ .1	<b>8.0</b> $\pm$ 27.1	29.1 $\pm$ 15.1	2.3 $\pm$ .8	.01 $\pm$ .02	.02 $\pm$ .1	<b>4.0</b> $\pm$ 19.6	27.9 $\pm$ 14.4	2.2 $\pm$ .8	.01 $\pm$ .02
DPrior	.3 $\pm$ 0.0	<b>54.0</b> $\pm$ 49.8	15.5 $\pm$ 11.7	2.1 $\pm$ .7	<b>4.1</b> $\pm$ 3.9	.3 $\pm$ 0.0	<b>51.0</b> $\pm$ 50.0	14.8 $\pm$ 8.4	2.2 $\pm$ .8	4.1 $\pm$ 4.1
GPMP	47.6 $\pm$ 1	<b>16.0</b> $\pm$ 36.7	32.3 $\pm$ 16.6	1.5 $\pm$ .5	.01 $\pm$ 0.0	48.5 $\pm$ 1	<b>19.0</b> $\pm$ 39.2	31.2 $\pm$ 17.9	1.4 $\pm$ .4	.01 $\pm$ 0.0
RRTC-GPMP	114.6 $\pm$ 56.9	100.0 $\pm$ 0.0	2.2 $\pm$ 1.3	4.1 $\pm$ 1.6	5.5 $\pm$ 3.0	111.2 $\pm$ 52.4	100.0 $\pm$ 0.0	2.4 $\pm$ 1.1	3.9 $\pm$ 1.4	5.59 $\pm$ 3.1
CPrior-GPMP	47.5 $\pm$ .3	<b>8.0</b> $\pm$ 27.1	27.7 $\pm$ 15.3	2.2 $\pm$ .8	.01 $\pm$ .03	47.9 $\pm$ .3	11.0 $\pm$ 31.3	29.0 $\pm$ 16.3	2.1 $\pm$ .7	.01 $\pm$ .03
DPrior-GPMP	48.3 $\pm$ 2	<b>39.0</b> $\pm$ 48.8	14.4 $\pm$ 7.7	2.2 $\pm$ .8	4.3 $\pm$ 4.1	49.0 $\pm$ .2	<b>43.0</b> $\pm$ 49.5	15.1 $\pm$ 8.5	2.1 $\pm$ .8	4.2 $\pm$ 3.9
CPost	.1 $\pm$ 2	50.0 $\pm$ 50.0	01.4 $\pm$ 01.4	2.3 $\pm$ .7	.02 $\pm$ .03	.1 $\pm$ 2	<b>52.0</b> $\pm$ 50.0	01.3 $\pm$ 01.5	2.2 $\pm$ .7	.02 $\pm$ .03
MPD	.3 $\pm$ .01	85.0 $\pm$ 35.7	2.0 $\pm$ 1.8	2.4 $\pm$ .7	4.2 $\pm$ 4.0	.3 $\pm$ .01	<b>82.0</b> $\pm$ 38.4	3.1 $\pm$ 2.1	2.4 $\pm$ .8	<b>4.2</b> $\pm$ 4.1

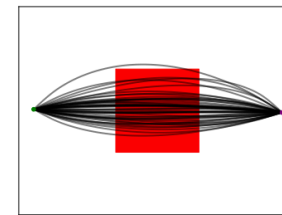
	Panda Spheres					Panda Spheres - Extra Obstacles				
	T[s] $\downarrow$	S[%] $\uparrow$	I[%] $\downarrow$	PL $\downarrow$	VAR $\uparrow$	T[s] $\downarrow$	S[%] $\uparrow$	I[%] $\downarrow$	PL $\downarrow$	VAR $\uparrow$
RRTC	42.9 $\pm$ 18.8	100.0 $\pm$ 0.0	.04 $\pm$ .08	12.1 $\pm$ 2.8		90.2 $\pm$ 103.6	100.0 $\pm$ 0.0	0.05 $\pm$ 0.05	13.2 $\pm$ 3.8	
CPrior	.02 $\pm$ .1	<b>36.0</b> $\pm$ 48.0	14.9 $\pm$ 15.0	5.4 $\pm$ 1.0	.02 $\pm$ .00	.02 $\pm$ .1	<b>10.0</b> $\pm$ 30.0	35.2 $\pm$ 24.6	5.4 $\pm$ 1.2	.01 $\pm$ .01
DPrior	.3 $\pm$ 0.0	<b>88.0</b> $\pm$ 32.5	11.5 $\pm$ 6.0	7.8 $\pm$ 1.4	<b>17.3</b> $\pm$ 5.0	.3 $\pm$ 0.0	<b>78.0</b> $\pm$ 41.4	23.9 $\pm$ 7.4	7.5 $\pm$ 1.6	18.0 $\pm$ 4.2
GPMP	194.4 $\pm$ 1	<b>42.0</b> $\pm$ 49.4	4.1 $\pm$ 4.8	5.1 $\pm$ 1.2	.02 $\pm$ .05	194.5 $\pm$ 2	<b>28.0</b> $\pm$ 44.9	9.6 $\pm$ 8.2	5.1 $\pm$ 1.3	.03 $\pm$ .12
RRTC-GPMP	230.4 $\pm$ 14.4	100.0 $\pm$ 0.0	02.4 $\pm$ 02.7	7.9 $\pm$ 1.4	18.7 $\pm$ 5.6	253.6 $\pm$ 58.7	96.0 $\pm$ 19.6	4.1 $\pm$ 2.8	8.0 $\pm$ 1.4	20.6 $\pm$ 7.2
CPrior-GPMP	191.6 $\pm$ .3	34.0 $\pm$ 47.4	5.2 $\pm$ 5.7	5.1 $\pm$ .9	.03 $\pm$ .06	194.2 $\pm$ 1	18.0 $\pm$ 38.4	12.3 $\pm$ 10.3	5.1 $\pm$ 1.1	.04 $\pm$ .06
DPrior-GPMP	192.3 $\pm$ .1	<b>100.0</b> $\pm$ 0.0	4.0 $\pm$ 2.6	6.8 $\pm$ 1.2	15.6 $\pm$ 3.5	192.1 $\pm$ .1	<b>82.0</b> $\pm$ 38.4	7.8 $\pm$ 3.8	7.1 $\pm$ 1.0	16.0 $\pm$ 3.6
CPost	.8 $\pm$ .1	70.9 $\pm$ 45.8	.7 $\pm$ 1.0	8.0 $\pm$ 1.3	.03 $\pm$ .04	.8 $\pm$ .1	<b>45.0</b> $\pm$ 49.8	1.5 $\pm$ 1.7	8.0 $\pm$ 1.2	1.1 $\pm$ 1.1
MPD	1.1 $\pm$ .01	100.0 $\pm$ 0.0	1.2 $\pm$ .8	9.9 $\pm$ 1.2	17.4 $\pm$ 3.9	1.1 $\pm$ .01	<b>93.0</b> $\pm$ 25.5	12.5 $\pm$ 7.6	10.0 $\pm$ 1.2	<b>18.0</b> $\pm$ 4.4

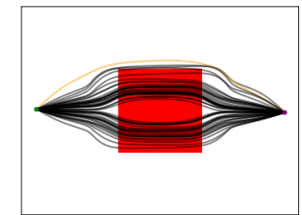
	Panda Shelf					Panda Shelf - Extra Obstacles				
	T[s] $\downarrow$	S[%] $\uparrow$	I[%] $\downarrow$	PL $\downarrow$	VAR $\uparrow$	T[s] $\downarrow$	S[%] $\uparrow$	I[%] $\downarrow$	PL $\downarrow$	VAR $\uparrow$
RRTC	29.9 $\pm$ 8.3	100.0 $\pm$ 0.0	0.04 $\pm$ 0.11	11.2 $\pm$ 2.4		31.5 $\pm$ 9.4	99.0 $\pm$ 10.0	2.2 $\pm$ 4.5	11.5 $\pm$ 2.9	
CPrior	.03 $\pm$ .2	<b>92.0</b> $\pm$ 27.1	4.3 $\pm$ 6.5	4.8 $\pm$ .9	.02 $\pm$ 0.0	.03 $\pm$ .1	<b>45.0</b> $\pm$ 45.7	8.3 $\pm$ 14.1	4.8 $\pm$ 1.8	.01 $\pm$ .00
DPrior	.3 $\pm$ 0.0	<b>100.0</b> $\pm$ 0.0	<b>3.6</b> $\pm$ 4.8	7.6 $\pm$ 1.2	<b>14.5</b> $\pm$ 3.4	.3 $\pm$ 0.0	<b>100.0</b> $\pm$ 0.0	<b>5.9</b> $\pm$ 7.3	7.6 $\pm$ 1.3	14.3 $\pm$ 3.7
GPMP	192.1 $\pm$ .1	<b>88.0</b> $\pm$ 32.5	.5 $\pm$ 1.6	5.1 $\pm$ 1.3	.01 $\pm$ .02	193.1 $\pm$ 13	<b>82.0</b> $\pm$ 38.42	2.0 $\pm$ 5.1	5.1 $\pm$ 1.4	.01 $\pm$ .01
RRTC-GPMP	218.3 $\pm$ 7.0	100.0 $\pm$ 0.0	.4 $\pm$ .9	8.1 $\pm$ 1.1	16.7 $\pm$ 3.8	228.2 $\pm$ 20.1	98.0 $\pm$ 14.0	1.5 $\pm$ 2.1	7.9 $\pm$ 1.5	18.3 $\pm$ 6.0
CPrior-GPMP	192.1 $\pm$ .1	94.0 $\pm$ 23.8	.6 $\pm$ 2.6	4.4 $\pm$ .9	.01 $\pm$ .00	192.1 $\pm$ .1	80.0 $\pm$ 49.0	3.3 $\pm$ 7.7	4.5 $\pm$ .8	.03 $\pm$ .04
DPrior-GPMP	192.3 $\pm$ .1	<b>100.0</b> $\pm$ 0.0	0.9 $\pm$ 1.1	7.1 $\pm$ 1.1	13.8 $\pm$ 2.9	192.6 $\pm$ .1	<b>98.0</b> $\pm$ 14.0	2.3 $\pm$ 2.4	6.9 $\pm$ 1.2	14.0 $\pm$ 3.1
CPost	.8 $\pm$ .1	93.0 $\pm$ 25.5	0.1 $\pm$ 0.4	7.4 $\pm$ 1.2	.02 $\pm$ .04	.8 $\pm$ .1	<b>83.0</b> $\pm$ 37.6	0.5 $\pm$ 1.1	10.8 $\pm$ 6.8	<b>2.4</b> $\pm$ 5.2
MPD	1.0 $\pm$ .01	100.0 $\pm$ 0.0	0.6 $\pm$ 0.9	9.3 $\pm$ 1.0	14.8 $\pm$ 3.2	1.0 $\pm$ .01	<b>99.0</b> $\pm$ 10.0	4.1 $\pm$ 4.9	9.4 $\pm$ 1.1	<b>15.1</b> $\pm$ 3.5



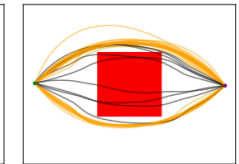
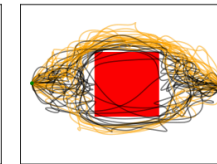
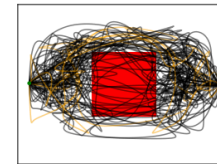
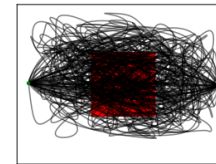
(a) Demonstrations



(b) Dprior

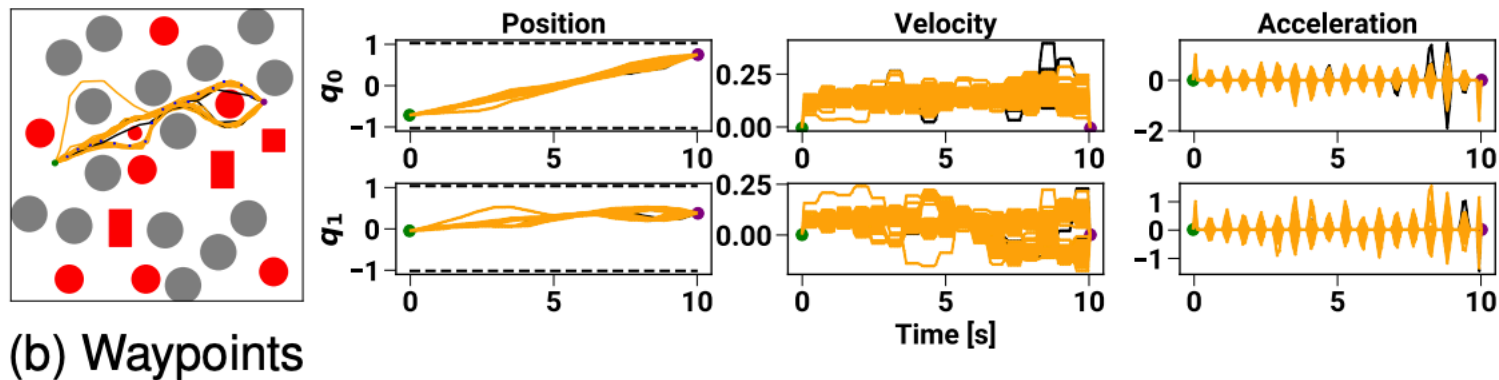
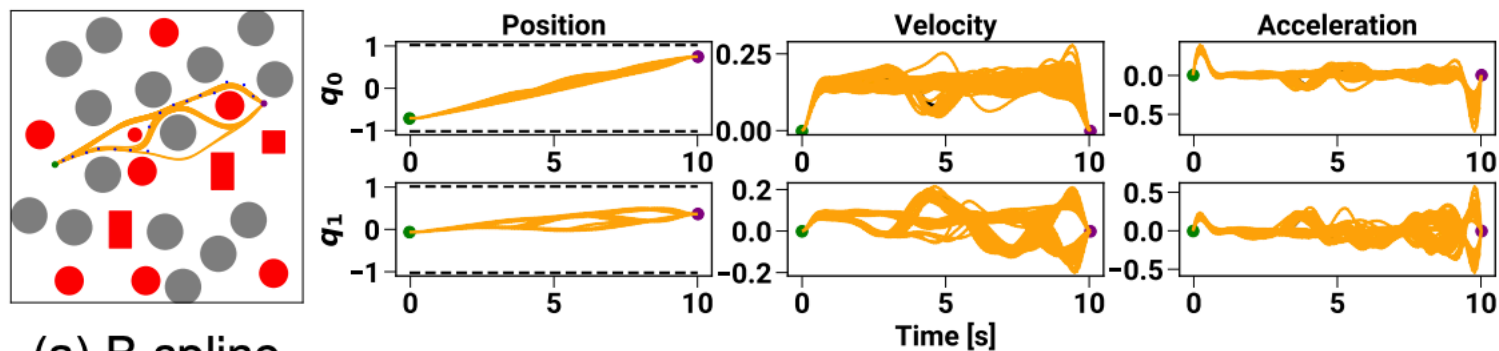


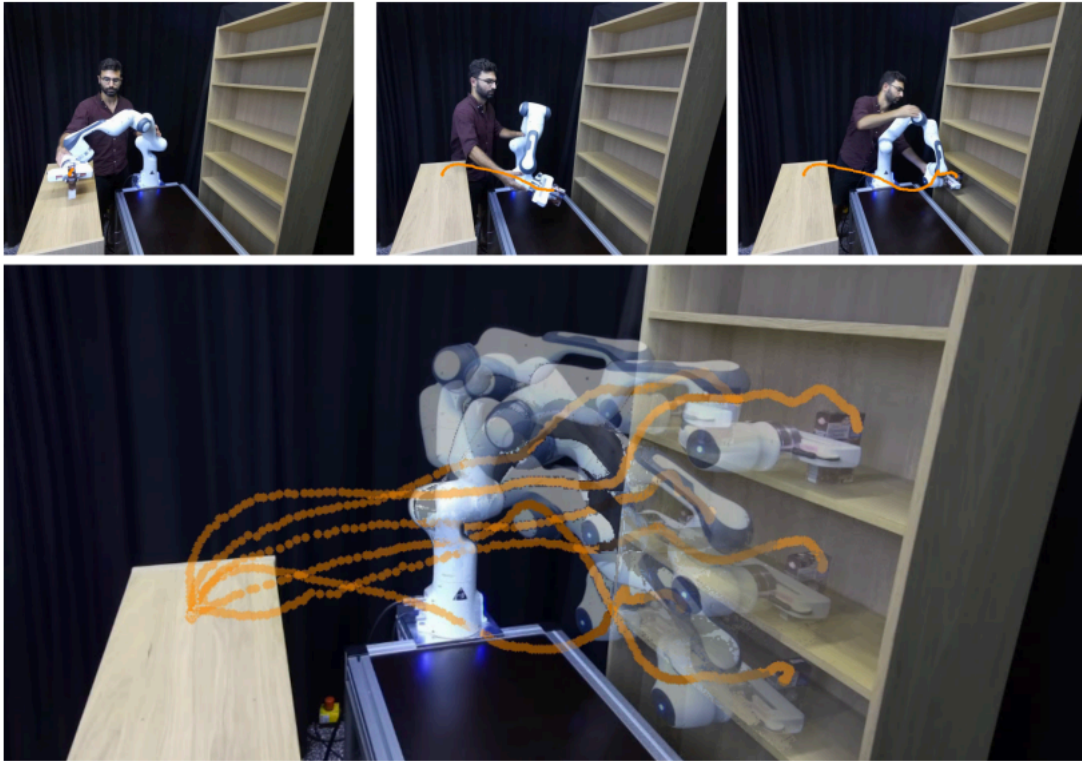
(c) Dprior + Cost

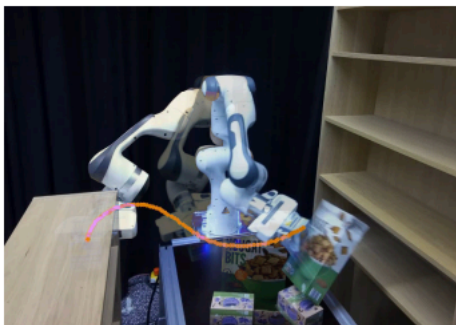


(d) MPD

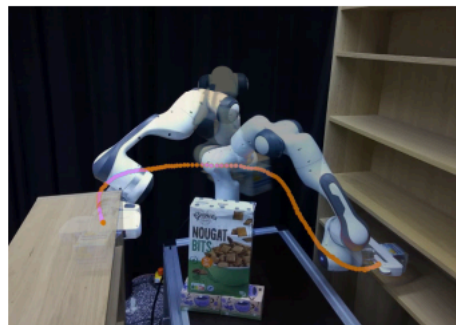
$i = N$  →  $i = 0$   
MPD denoising process



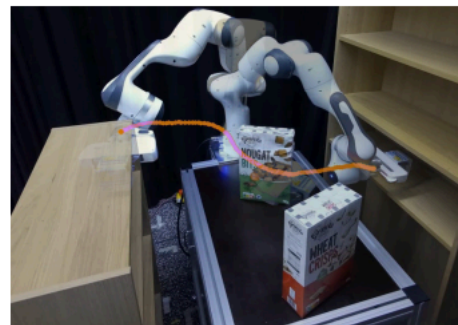




(a) Diffusion prior



(b) MPD



PERFORMANCE METRICS FOR A REAL-WORLD MOTION PLANNING TASK  
IN THE ENVWAREHOUSE-ROBOTPANDA ENVIRONMENT WITH  
ADDITIONAL OBJECTS USING DIFFUSION-BASED MODELS.

Algorithms	Success rate [%]	Fraction valid [%]	Diversity
Dprior	0.0	—	—
Dprior+Cost	100.0	37.0	27.95
MPD	100.0	78.0	59.50

- **Limitations:**

- ▶ Trajectory duration is fixed - no optimization for faster trajectories.
- ▶ Environments cannot undergo major structural changes.
- ▶ Have perfect knowledge of environment.
- ▶ Cost gradient computation is expensive (authors note that they faced issues while parallelizing this).
- ▶ No safety guarantees - the authors mention that they could project the sample at each step onto a (safe) manifold, but it is computationally expensive.
- ▶ No other guarantees provided, completeness or soundness are not guaranteed.