SE(3)-DiffusionFields:
Learning smooth cost functions for joint grasp and motion optimization through diffusion

ICRA 2023

박지호

2024.03.12

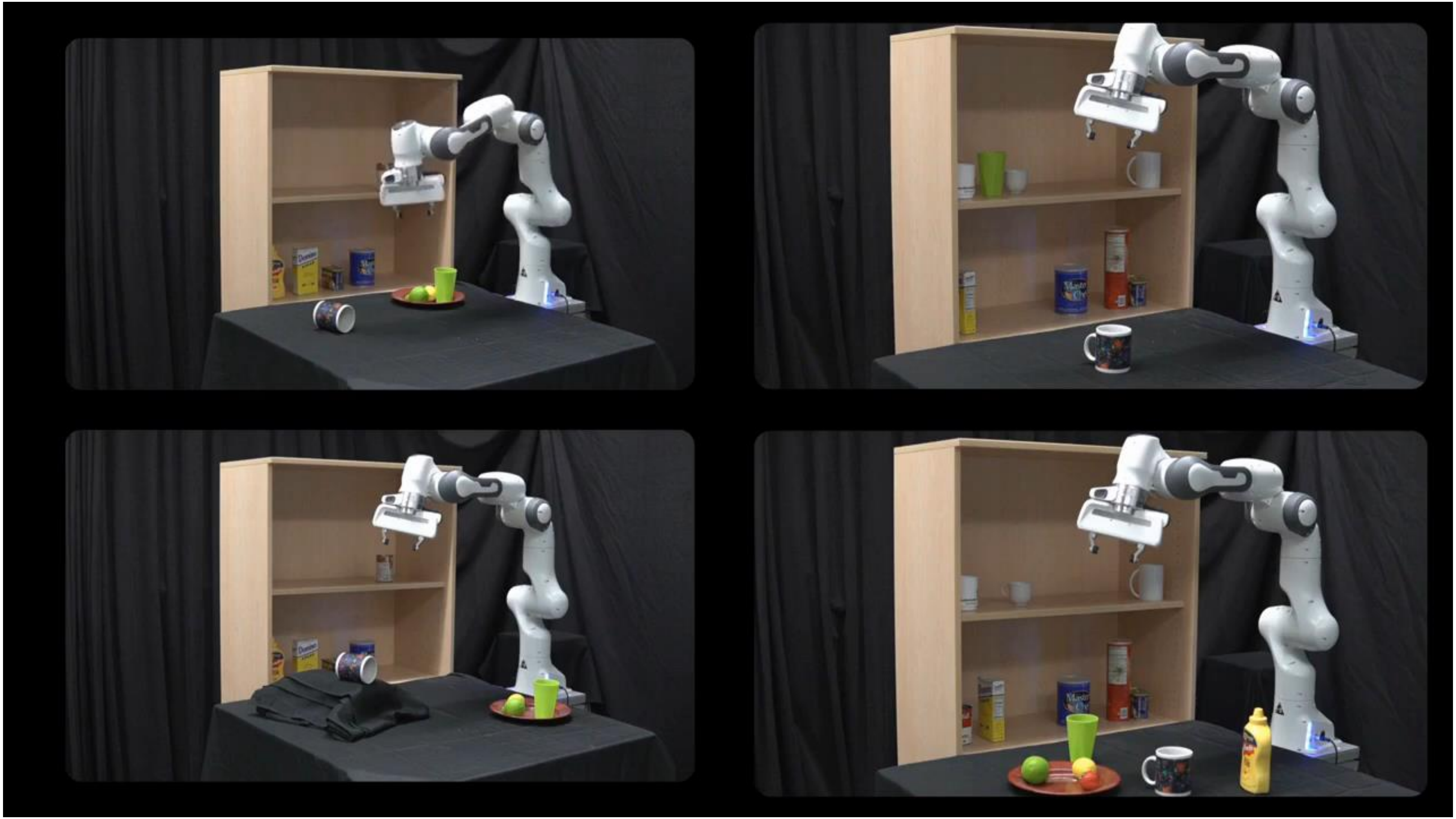
SE(3)-DiffusionFields

Task

- Pick and Place: Grasp + Motion Planning

Contributions

- Diffusion for Grasp(SE(3))
- *Energy Prediction (not Score)* → Grasp & Motion Joint Optimization



Contents

Preliminaries

1. Motion Optimization
2. SE(3) Lie Group
3. Score-Based Generative Model

1. Motion Planning

Methods

1) Data-Driven:

- Train Motion Generator → Sample top-k → eval & select

2) Optimization:

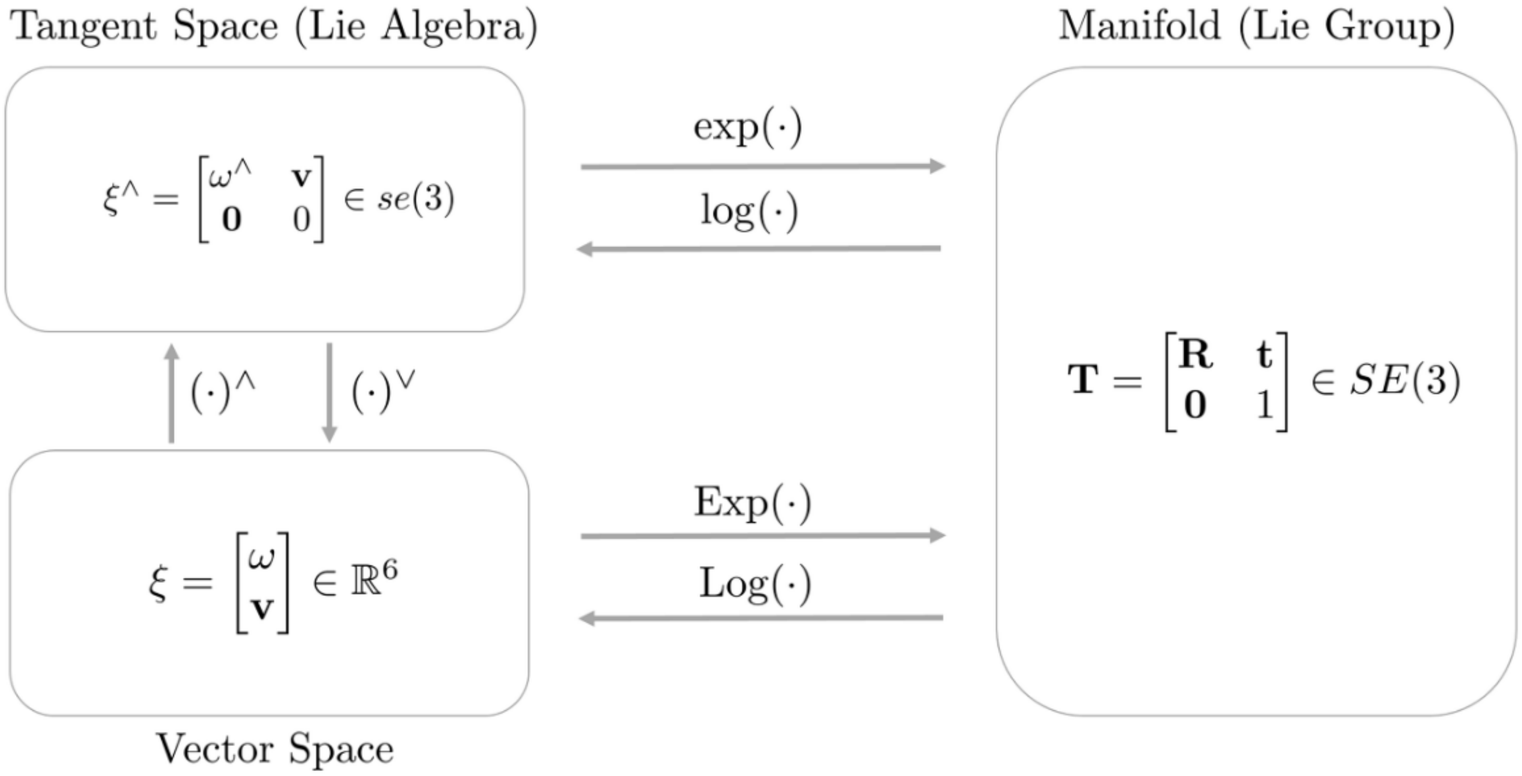
- Objective Function(ex. collision, target distance) → Trajectory Optimization

Common Strategy: *1) Data Driven*

- Advantages: Fast Inference, High performance,
- Disadvantages: Train Cost(Rely on Task Specific Generator!)

2. SE(3) LieGroup

"Lie 군 중 하나인 Special Euclidean 3 (SE(3))군은 3차원 공간 상에서 강체의 변환과 관련된 행렬과 이에 닫혀있는 연산들로 구성된 군을 의미한다."



https://alida.tistory.com/9#org3029dbb

2. SE(3) LieGroup

- Rotation & Translation Representations

$$SE(3) = \left\{ \mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \in \mathbb{R}^{4\times4} \mid \mathbf{R} \in SO(3), \mathbf{t} \in \mathbb{R}^3 \right\} \Longleftrightarrow \mathrm{se}(3) = \left\{ \xi^{\wedge} = \begin{bmatrix} \omega^{\wedge} & \mathbf{v} \\ \mathbf{0} & 0 \end{bmatrix} \in \mathbb{R}^{4\times4} \middle| \xi = \begin{bmatrix} \omega \\ \mathbf{v} \end{bmatrix} \in \mathbb{R}^6 \right\}$$

Rotation(9-dim) + Translation(3-dim)

=> 12-dim

Rotation(3-dim) + Translation(3-dim)

=> 6-dim

https://alida.tistory.com/9#org3029dbb

2. SE(3) LieGroup

- Rotation & Translation Representations

$$SE(3) = \left\{ \mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \in \mathbb{R}^{4\times4} \mid \mathbf{R} \in SO(3), \mathbf{t} \in \mathbb{R}^3 \right\}$$

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{bmatrix} = \begin{bmatrix} R_{11} & R_{12} & R_{13} & T_x \\ R_{21} & R_{22} & R_{23} & T_y \\ R_{31} & R_{32} & R_{33} & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{bmatrix}$$

Rotation(9-dim) + Translation(3-dim)

=> 16-dim

https://alida.tistory.com/9#org3029dbb

3. Score-Based Generative Model

- Probability Model: Gibbs(Boltzman) Distribution
- Score : Gradient of Log probability = Gradient of Energy
- Diffusion Model: Langevin Dynamics

3. Score-Based Generative Model

Gibbs(Boltzman) Distribution

- Probability Model: Energy(ε) of i → Probability of i

Physics

$$p_i = \frac{1}{Q} e^{-\varepsilon_i/(kT)} = \frac{e^{-\varepsilon_i/(kT)}}{\sum_{j=1}^{M} e^{-\varepsilon_j/(kT)}}$$

$$\Downarrow$$

Generative Model

$$p_\theta(\mathbf{x}) = \frac{\exp(-E_\theta(\mathbf{x}))}{Z(\theta)}$$

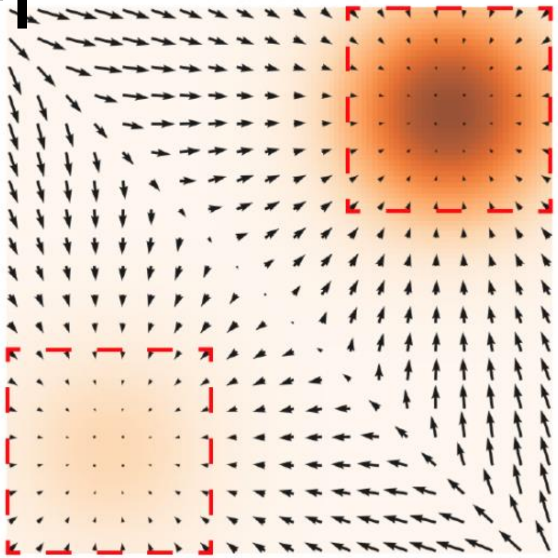
$$Z(\theta) = \int \exp(-E_\theta(\mathbf{x}))d\mathbf{x}$$

3. Score-Based Generative Model

Score: **Gradient of log Probability = Gradient of Energy**

$$\boldsymbol{Score} = \boldsymbol{\nabla_x \log p(x)}$$

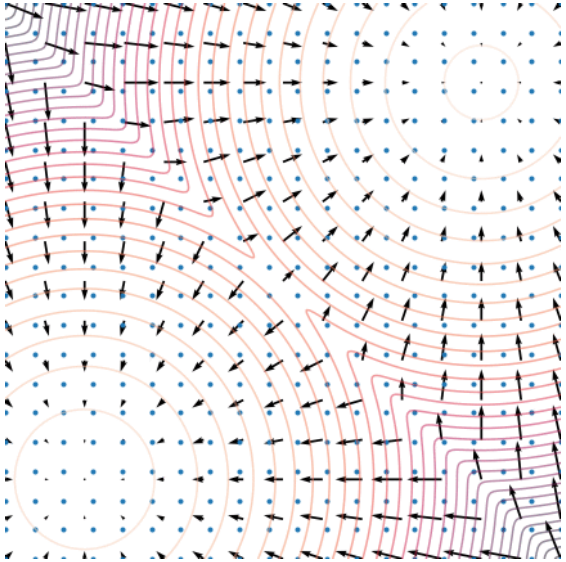
$$= \nabla_x \log \frac{1}{Z} e^{-E_\theta(x)} = -\nabla_x E_\theta(x) - \nabla_x \log Z = -\boldsymbol{\nabla_x E_\theta(x)}$$



Diffusion Model: Score Predictor $s_\theta(x)$

Loss :
$$\frac{1}{2}\mathbb{E}_{q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})p_{\text{data}}(\mathbf{x})}\left[\|\mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}) - \nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x})\|_2^2\right].$$

(Denoising Score Matching)



3. Score-Based Generative Model

Langevin Dynamics → Diffusion Sampling

- mathematical modeling of the dynamics of molecular systems
- Nature prefers to go to Low Energy (Low Energy = High Probability)

Physics

$$M\,\ddot{\mathbf{X}} = -\nabla U(\mathbf{X}) - \gamma\, M\,\dot{\mathbf{X}} + \sqrt{2\,M\,\gamma\,k_{\mathrm{B}}T}\,\mathbf{R}(t)\,,$$

Generative Model

$$\tilde{\mathbf{x}}_t = \tilde{\mathbf{x}}_{t-1} + \frac{\epsilon}{2}\nabla_{\mathbf{x}} \log p(\tilde{\mathbf{x}}_{t-1}) + \sqrt{\epsilon}\,\mathbf{z}_t,$$

Method

1. **Grasp**: SE(3)-DiffusionField

- Object Dependent Grasp(SE(3)) Prediction

2. **Grasp + Motion Joint Optimization** (via Diffusion Inverse Sampling)

- No training, Only Optimization (w/ pretrained SE(3)-Diff)

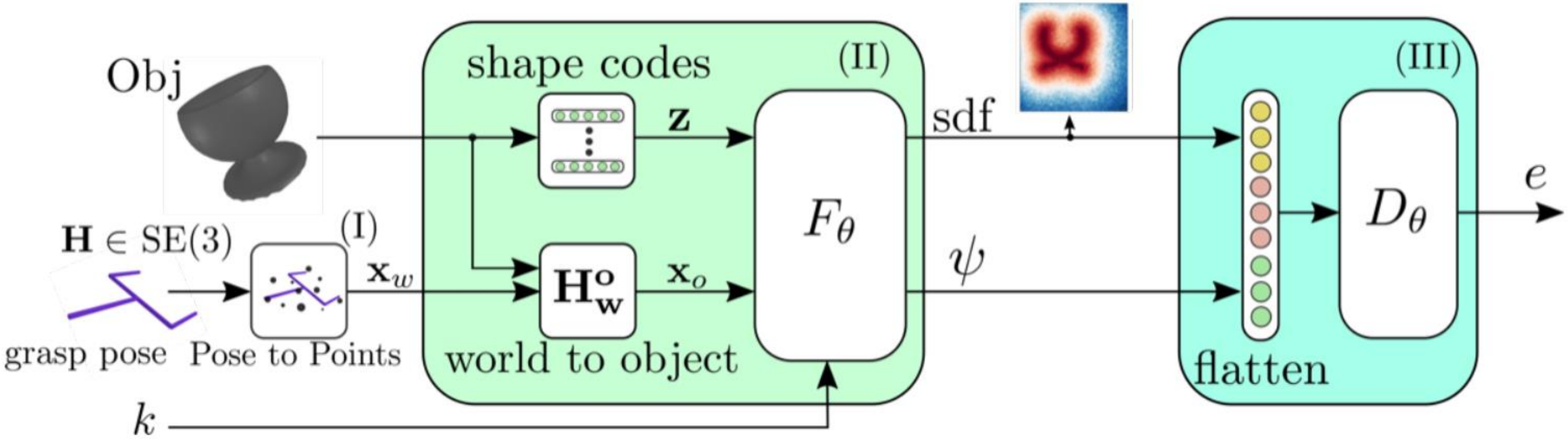
1. Grasp: SE(3)-DiffusionField

Train Dataset

- Acronym: Grasp Dataset(Simulation)

SE(3)-Diff

- Input: Object Pose & id, Grasp(6DoF SE(3)), Diffusion Timestep(k)
- Output: Energy of Grasp(H)



1. Grasp: SE(3)-DiffusionField

If Model Predicts Energy instead of Score...

How can we Train Diffusion Denoising?

→ $\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{H}, k) = -DE_{\boldsymbol{\theta}}(\boldsymbol{H}, k)/D\boldsymbol{H}$

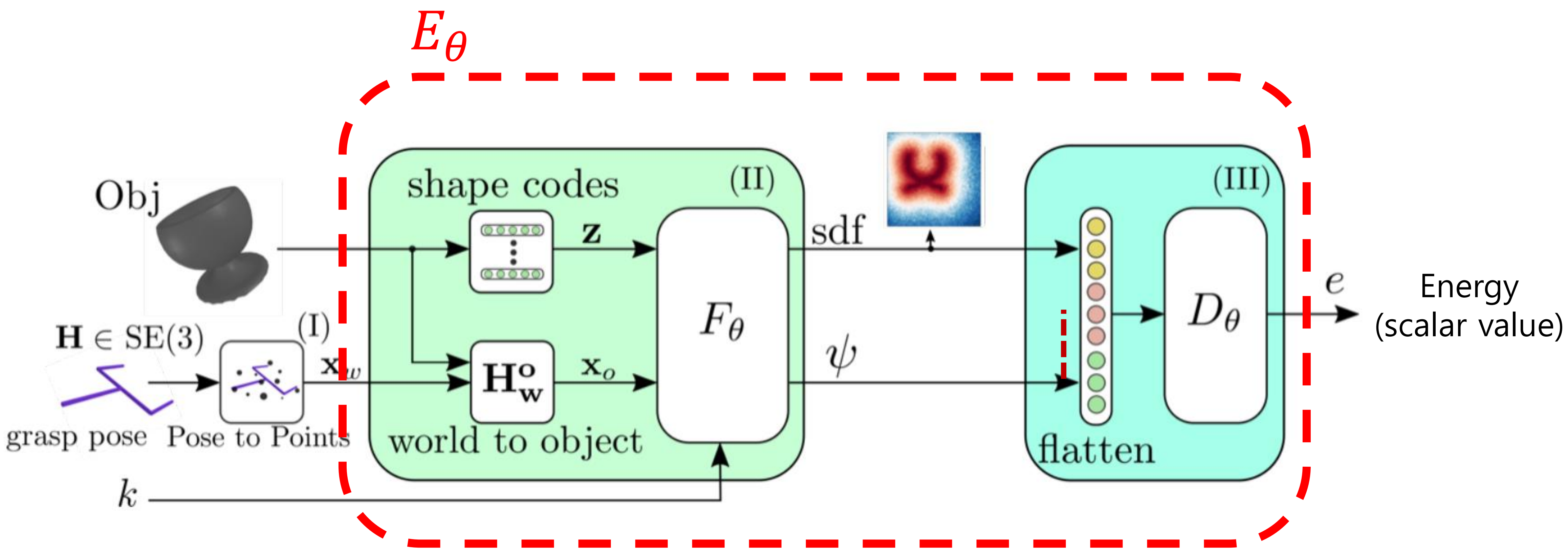
Score Matching Loss

$$\mathcal{L}_{\text{dsm}} = \frac{1}{L} \sum_{k=0}^{L} \mathbb{E}_{\boldsymbol{x}, \hat{\boldsymbol{x}}} \left[\left|\left| \boldsymbol{s}_{\boldsymbol{\theta}}(\hat{\boldsymbol{x}}, k) - \nabla_{\hat{\boldsymbol{x}}} \log \mathcal{N}(\hat{\boldsymbol{x}} | \boldsymbol{x}, \sigma_k^2 \boldsymbol{I}) \right|\right| \right],$$

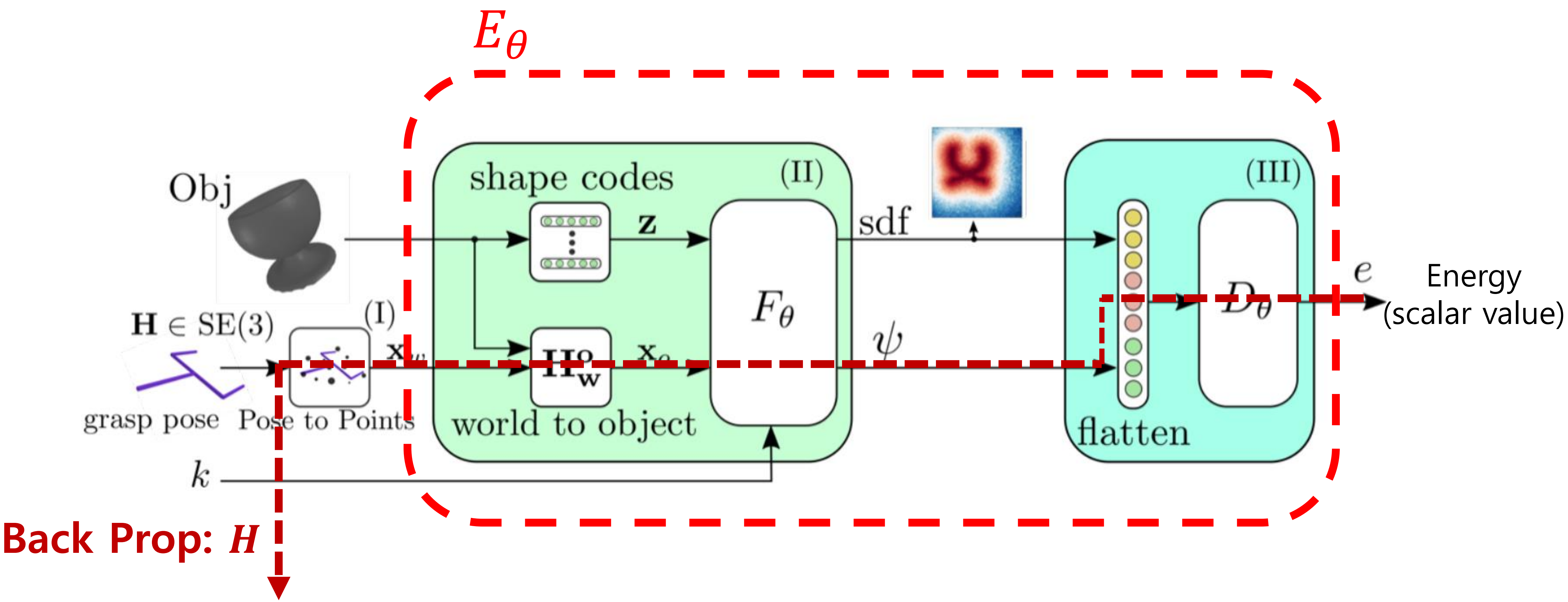
Diffusion Sampling(Denoising)

$$\boldsymbol{x}_{k-1} = \boldsymbol{x}_k + \frac{\alpha_k^2}{2} \boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{x}_k, k) + \alpha_k \boldsymbol{\epsilon}, \; \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}),$$

1. Grasp: SE(3)-DiffusionField

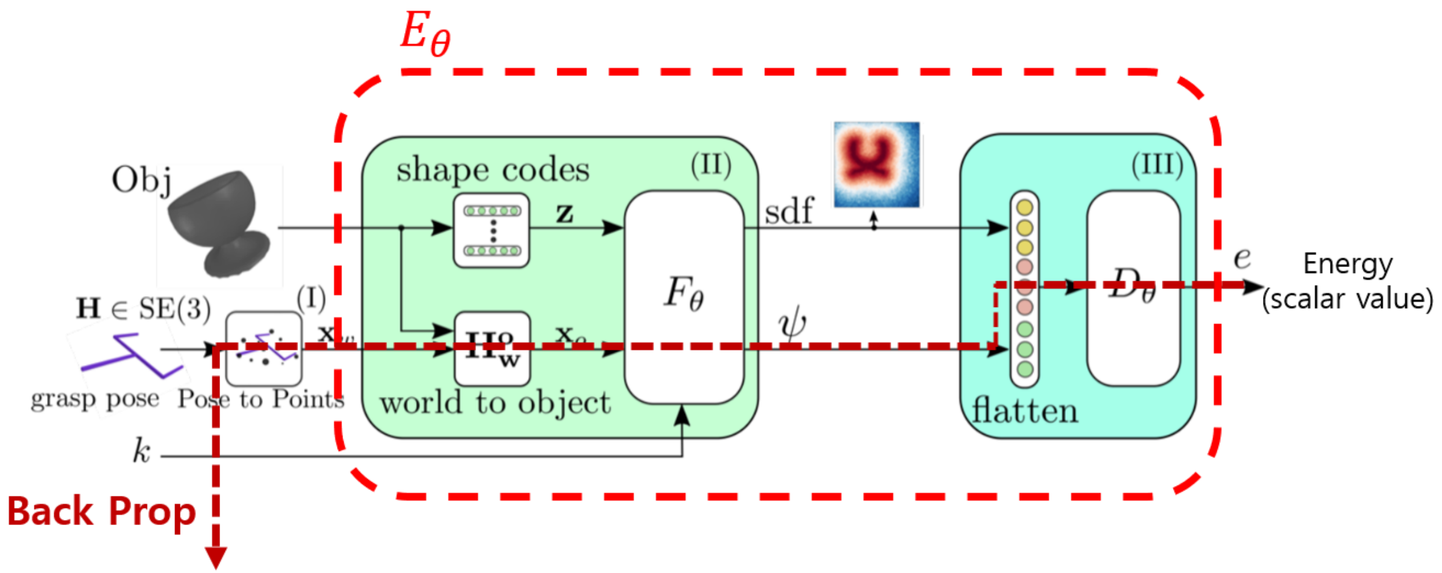


1. Grasp: SE(3)-DiffusionField



$$s_\theta(H, k) = -DE_\theta(H, k)/DH$$

1. Grasp: SE(3)-DiffusionField

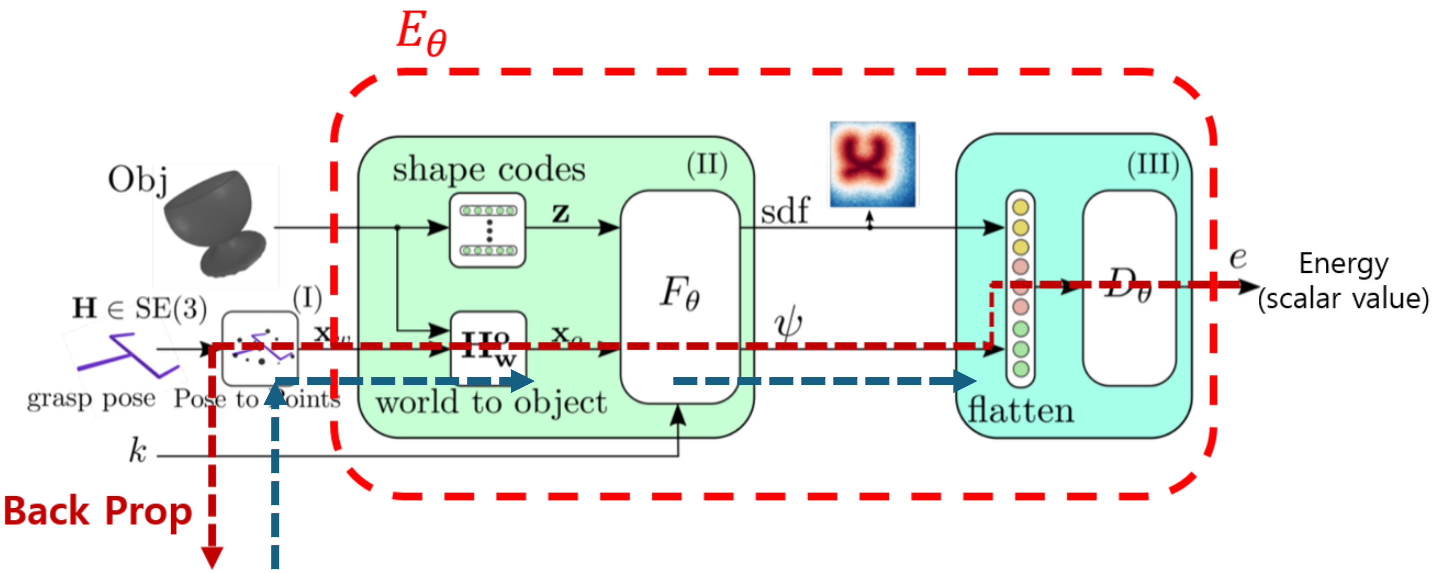


$$\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{H}, k) = -DE_{\boldsymbol{\theta}}(\boldsymbol{H}, k)/D\boldsymbol{H}$$

$$\mathcal{L}_{\text{dsm}} = \frac{1}{L}\sum_{k=0}^{L}\mathbb{E}_{\boldsymbol{H},\hat{\boldsymbol{H}}}\left[\left\|\boldsymbol{s}_{\boldsymbol{\theta}}(\hat{\boldsymbol{H}}, k) - \frac{D\log q(\hat{\boldsymbol{H}}|\boldsymbol{H}, \sigma_k \boldsymbol{I})}{D\hat{\boldsymbol{H}}}\right\|\right],$$

(Added Random Noise)

1. Grasp: SE(3)-DiffusionField



$$\boldsymbol{s_\theta}(\boldsymbol{H}, k) = -DE_{\boldsymbol{\theta}}(\boldsymbol{H}, k)/D\boldsymbol{H}$$

Back Prop & Update $\mathbf{E}_\theta$

$$\mathcal{L}_{\text{dsm}} = \frac{1}{L}\sum_{k=0}^{L}\mathbb{E}_{\boldsymbol{H},\hat{\boldsymbol{H}}}\left[\left\|\boldsymbol{s_\theta}(\hat{\boldsymbol{H}}, k) - \frac{D\log q(\hat{\boldsymbol{H}}|\boldsymbol{H}, \sigma_k \boldsymbol{I})}{D\hat{\boldsymbol{H}}}\right\|\right],$$

(Added Random Noise)

1. Grasp: SE(3)-DiffusionField

Train Loss

Diffusion Loss + SDF Loss

(3D Edge of Object)

Algorithm 1: Grasp SE(3)-DiF Training

Given: $\boldsymbol{\theta}_0$: initial params for $\boldsymbol{z}$, $F_{\boldsymbol{\theta}}$, $D_{\boldsymbol{\theta}}$;
Datasets: $\mathcal{D}_o : \{m, \boldsymbol{H}_w^o\}$, object ids and poses,
$\mathcal{D}_{sdf}^m : \{\boldsymbol{x}, \text{sdf}\}$, 3D positions $\boldsymbol{x}$ and sdf for object m,
$\mathcal{D}_g^m : \{\boldsymbol{H}\}$ succesful grasp poses for object m;

for $s \leftarrow 0$ **to** $S-1$ **do**
- $k, \sigma_k \leftarrow [0, \ldots, L]$; `// sample noise scale`
- $m, \boldsymbol{H}_w^o \in \mathcal{D}_o$; `// sample objects ids and poses`
- $\boldsymbol{z} = \text{shape codes}(m)$; `// get shape codes`
- **SDF train**
- $\boldsymbol{x}, \text{sdf} \in \mathcal{D}_{sdf}^m$; `// get 3D points and sdf for obj. m`
- $\hat{sdf}, \_ = F_{\boldsymbol{\theta}}(\boldsymbol{H}_w^o \boldsymbol{x}, \boldsymbol{z}, k)$; `// get predicted sdf`
- $L_{\text{sdf}} = \mathcal{L}_{\text{mse}}(\hat{sdf}, sdf)$; `// compute sdf error`
- **Grasp diffusion train**
- $\boldsymbol{H} \sim \mathcal{D}_g^m$; `// Sample success grasp poses for obj. m`
- $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma_k \boldsymbol{I})$; `// sample white noise on k scale`
- $\hat{\boldsymbol{H}} = \boldsymbol{H}\text{Expmap}(\boldsymbol{\epsilon})$; `// perturb grasp pose Eq. (4)`
- $\boldsymbol{x}_n^o = \hat{\boldsymbol{H}} \boldsymbol{x}_n$; `// Transform to N 3d points (see Figure 3)`
- $\hat{sdf}_n, \boldsymbol{\psi}_n = F_{\boldsymbol{\theta}}(\boldsymbol{x}_n^o, \boldsymbol{z}_b, k)$; `// get features`
- $\Psi = \text{Flatten}(\hat{sdf}_n, \boldsymbol{\psi}_n)$; `// Flatten the features`
- $e = D_{\boldsymbol{\theta}}(\Psi)$; `// compute energy`
- $L_{\text{dsm}} = \mathcal{L}_{\text{dsm}}(e, \hat{\boldsymbol{H}}, \boldsymbol{H}, \sigma_k)$; `// Compute dsm loss Eq. (5)`
- **Parameter update**
- $L = L_{\text{dsm}} + L_{\text{sdf}}$; `// Sum losses`
- $\boldsymbol{\theta}_{s+1} = \boldsymbol{\theta}_s - \alpha \nabla_{\boldsymbol{\theta}} L$; `// Update parameters`

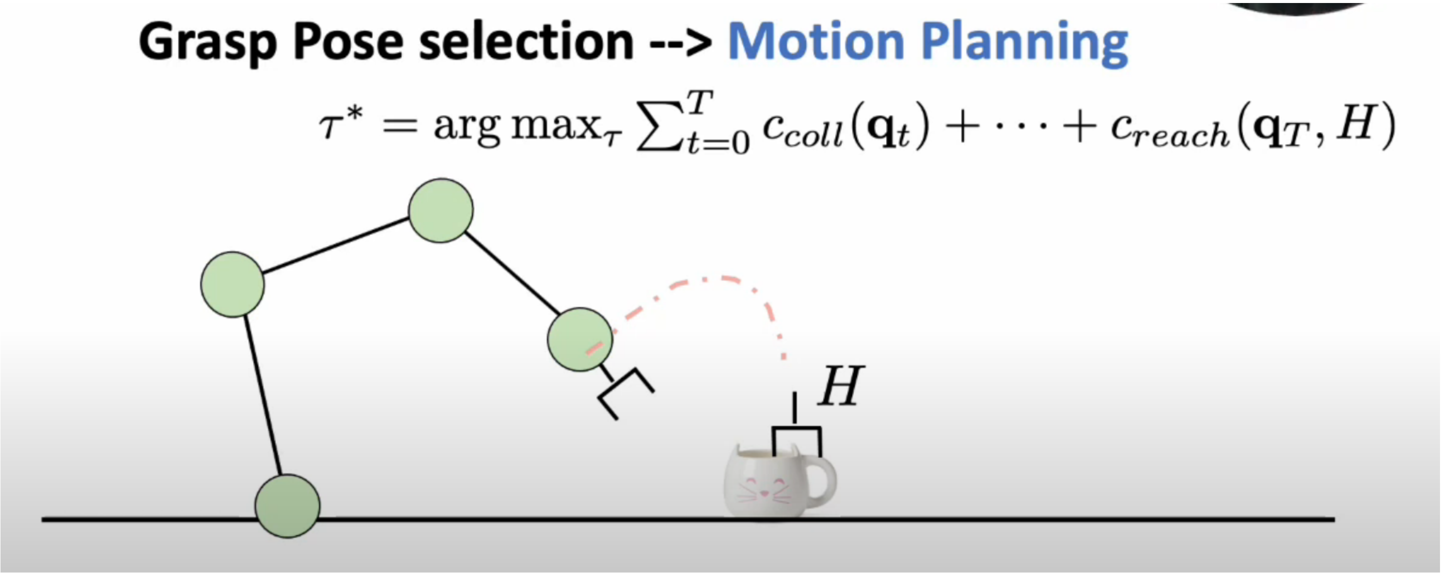
return $\boldsymbol{\theta}^*$;

2. Grasp + Motion Optimization

q_t : Robot Joints

τ: $\{q_t\}_{t=1}^{T}$: Trajectory

$\mathcal{J}$: Objective Function



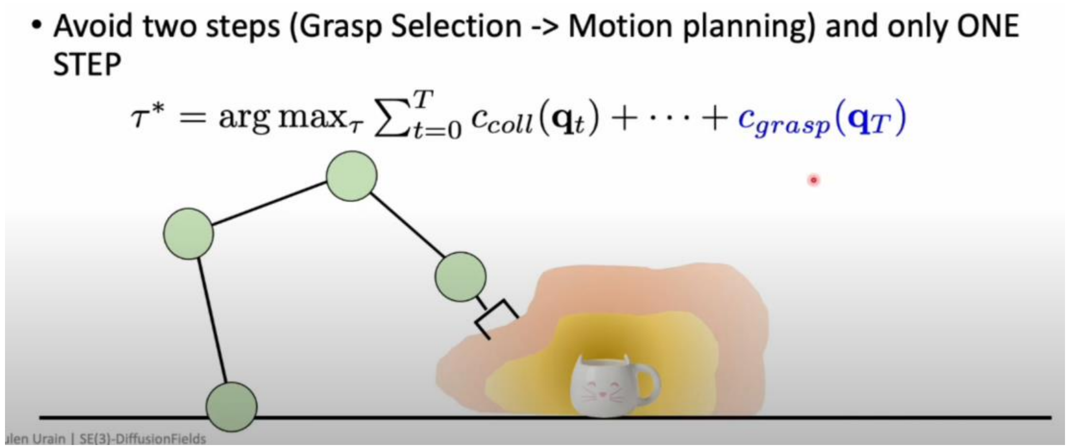
Motion Optimization

$$\boldsymbol{\tau}^* = \arg\min_{\boldsymbol{\tau}} \mathcal{J}(\boldsymbol{\tau}) = \arg\min_{\boldsymbol{\tau}} \sum_j \omega_j c_j(\boldsymbol{\tau})$$

2. Grasp + Motion Optimization

1) Add Energy in $\mathcal{J}$

- $c_{grasp} = E_\theta(q_t)$



2) Diffusion Sampling

- Diffusion Inverse Process
  (Instead of Gradient Descent)

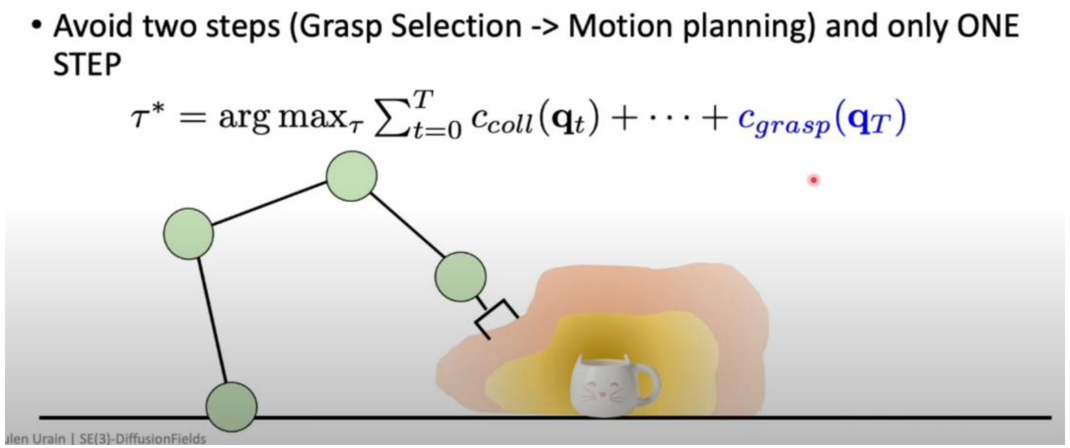
$$\boldsymbol{\tau}_{k-1} = \boldsymbol{\tau}_k + 0.5\ \alpha_k^2 \nabla_{\boldsymbol{\tau}_k} \log q(\boldsymbol{\tau}|k) + \alpha_k \boldsymbol{\epsilon}\,,\ \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}),$$

$$q(\boldsymbol{\tau}|k) \propto \exp(-\mathcal{J}(\boldsymbol{\tau}, k))$$

2. Grasp + Motion Optimization

1) Add Energy in $\mathcal{J}$

- $c_{grasp} = E_\theta(q_t)$



2) Diffusion Sampling

- Diffusion Inverse Process
  (Instead of Gradient Descent)

$$\boldsymbol{\tau}_{k-1} = \boldsymbol{\tau}_k + 0.5\ \alpha_k^2 \nabla_{\boldsymbol{\tau}_k} \log q(\boldsymbol{\tau}|k) + \alpha_k \boldsymbol{\epsilon}\,,\ \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}),$$

$$q(\boldsymbol{\tau}|k) \propto \exp(-\mathcal{J}(\boldsymbol{\tau}, k))$$

In Fact,
Adding Gradient of motion objectives, in Grasp Diffusion!

Experiments

1) Grasp Evaluation (Simulation)

2) Grasp + Motion Evaluation (Simulation)

- Comparing with other models

3) Real World

- 20 trials for each experiment(4)
- Results: 100%, 90%, 95%, 100%

Experiments

1) Grasp Evaluation

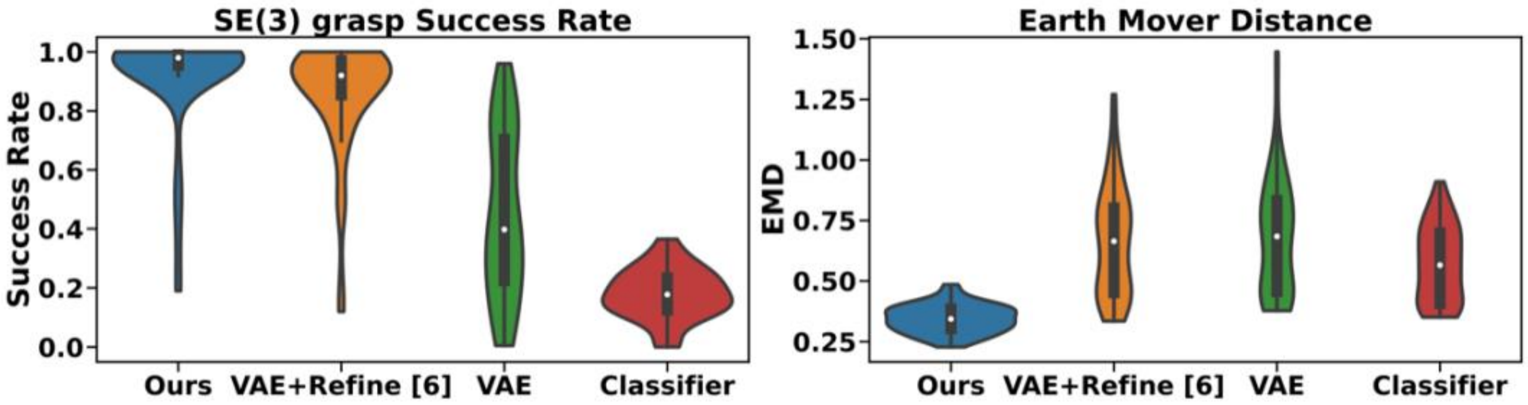


Fig. 4: 6D grasp pose generation experiment. Left: Success rate evaluation. Right: Earth Mover Distance (EMD) evaluation metrics (lower is better).

Experiments

2) Grasp + Motion Evaluation

- joint(class): Grasp Classifier + Motion Objective → Optimization

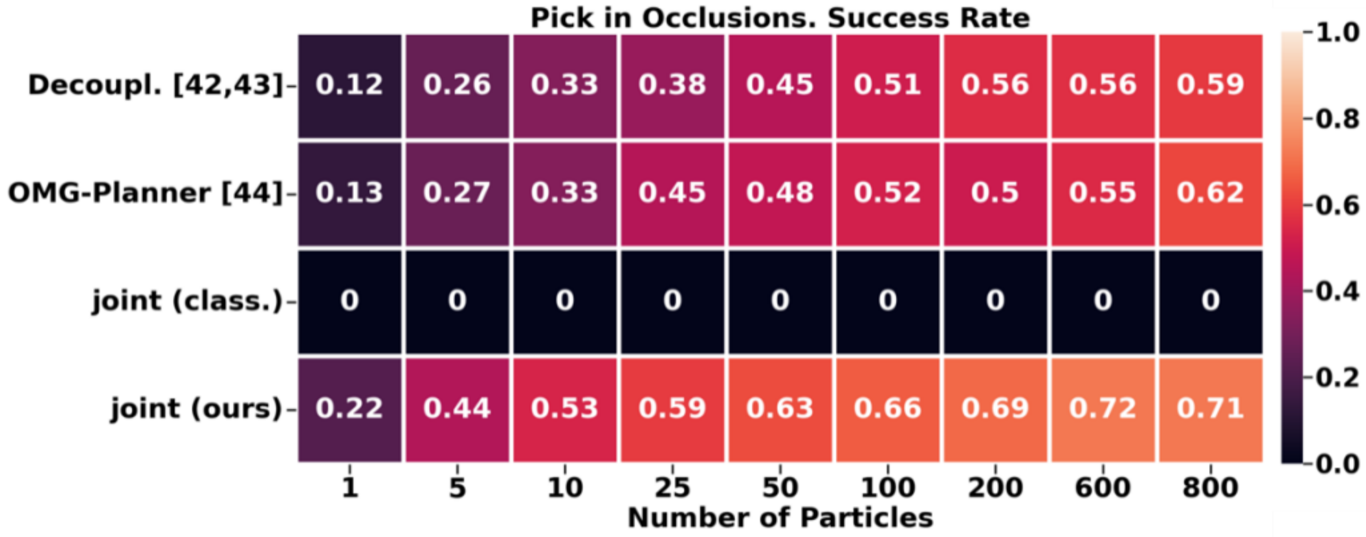
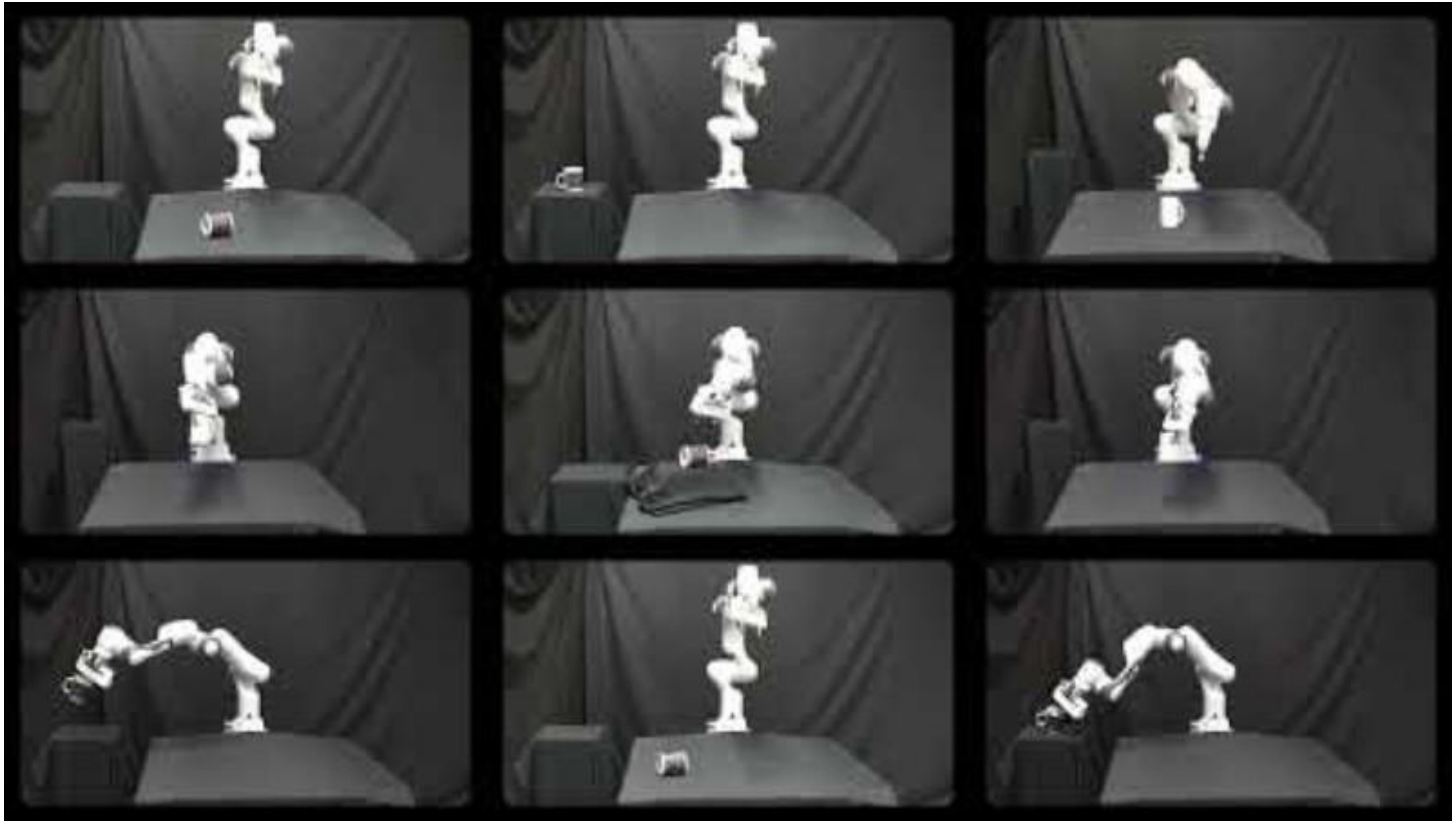
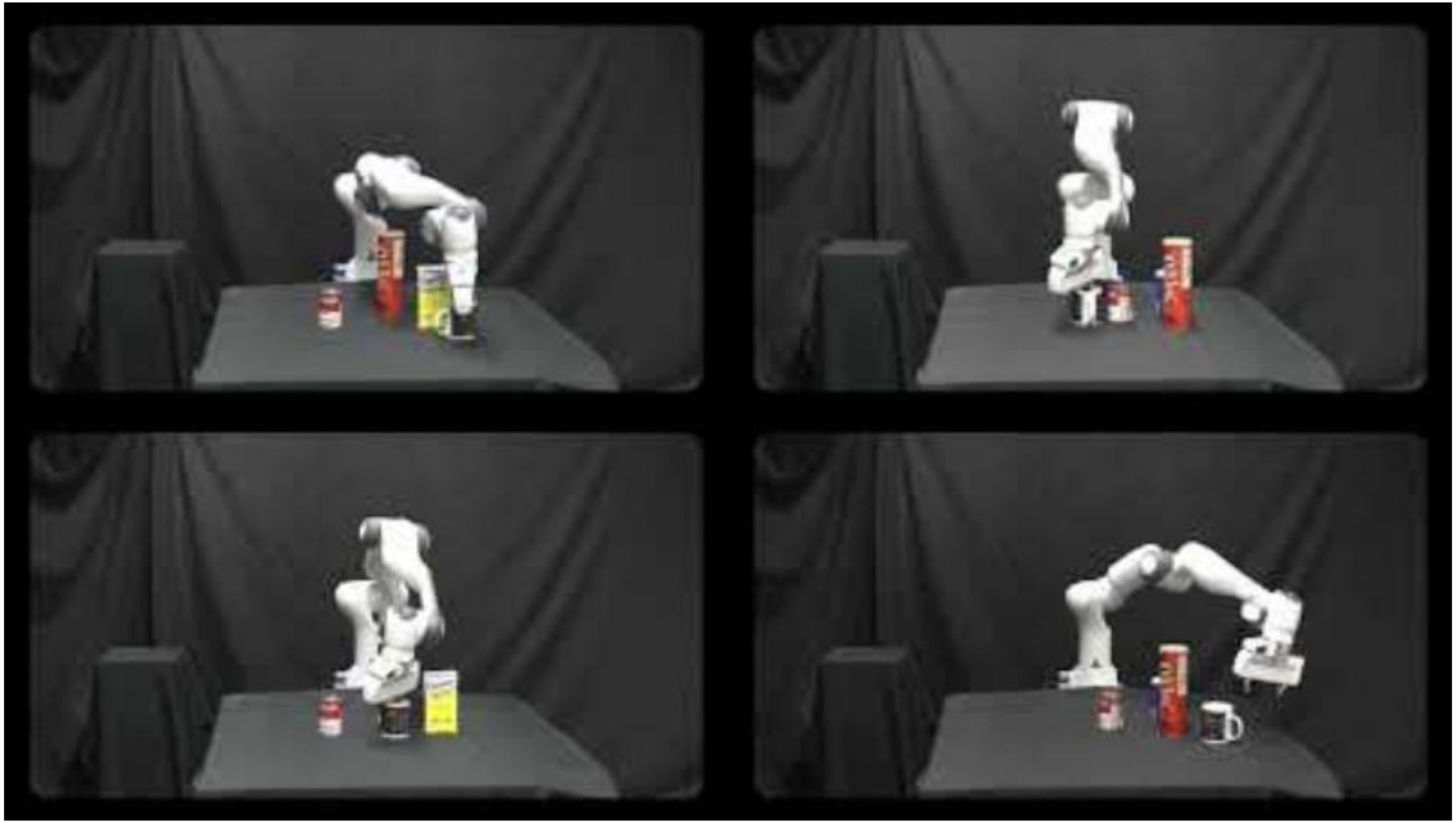


Fig. 5: Evaluation Pick in occlusion. We measure the success rate of 4 different methods based on different number of initializations.

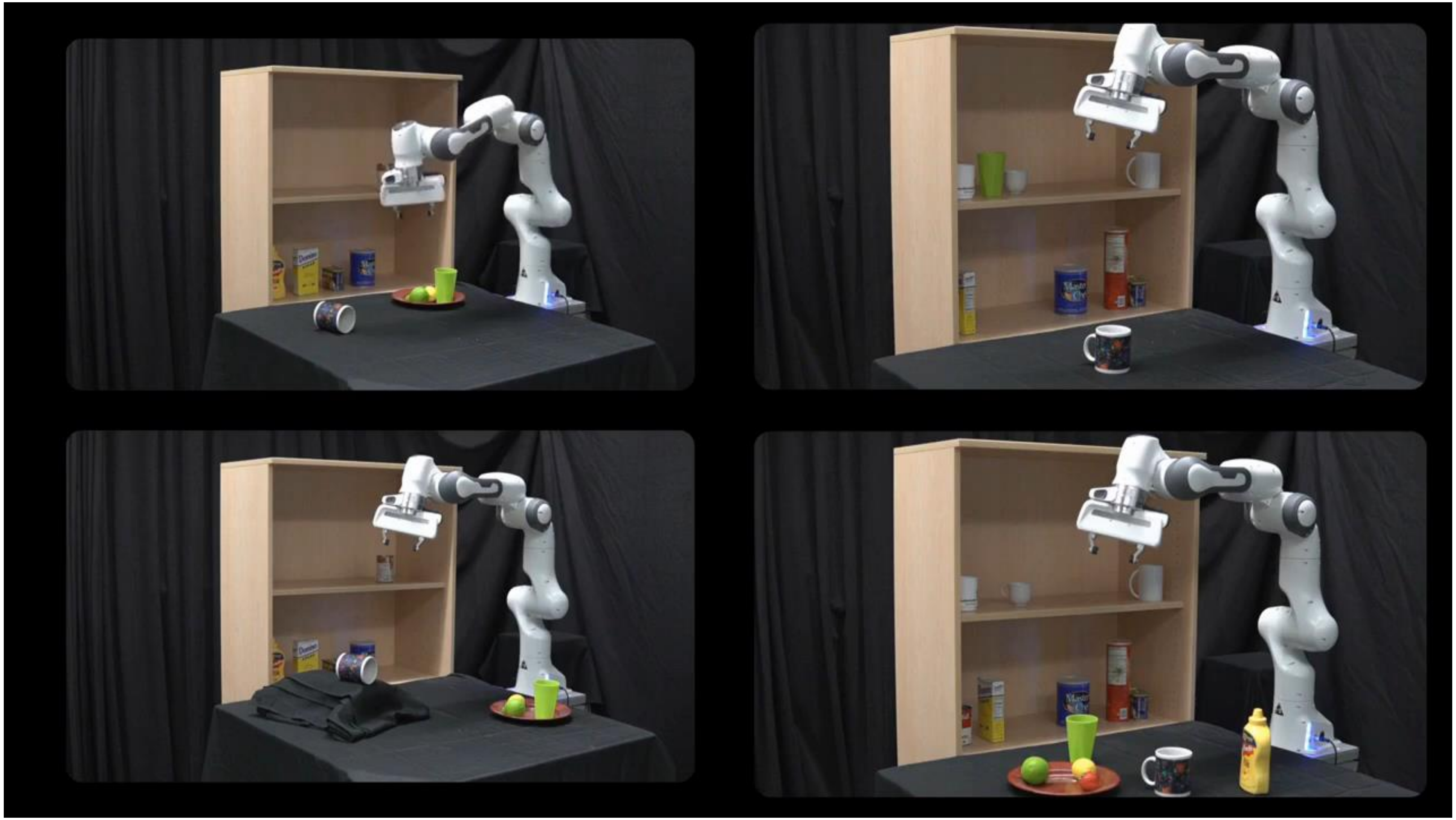
Experiments



Experiments



Experiments



Discussion

Is Predicting Energy Better?

- SE(3) dimension이 낮아서 energy-based modeling이 가능한 것일까?
- Score prediction 해도 Joint Optimization은 똑같이 적용이 가능한데, Unified Objective Function 스토리를 위한 것일까?

Thank You!