DeepSimHO: Stable Pose Estimation for Hand-Object Interaction via Physics Simulation —Supplementary Materials—

Rong Wang **Wei Mao** **Hongdong Li**
The Australian National University
{rong.wang, wei.mao, hongdong.li}@anu.edu.au

1 Datasets

Both DexYCB [3] and HO3D v2 [6] datasets are for non-commercial purpose only. The licenses for them can be found in the below links.

- DexYCB: `https://dex-ycb.github.io/`.
- HO3D v2: `https://files.icg.tugraz.at/d/5224b9b16422474892d7/files/?p=%2FREADME.txt`.

2 More Implementation Details

2.1 DeepSim Architecture

MLP. We flatten the initial hand mesh $\hat{\mathbf{M}}^h$ and object mesh $\hat{\mathbf{M}}^o = \hat{\mathbf{R}}^o\mathbf{M}^o + \hat{\mathbf{t}}^o$, both normalized by the hand root joint, into a 1D vector. We then concatenate them with the two signed distance vectors $\hat{\mathbf{c}}^h$ and $\hat{\mathbf{c}}^o$ as an input feature $\mathbf{F} \in \mathbb{R}^{7172}$ for the DeepSim. The DeepSim is a 6 layer MLP, where the linear layer dimensions are [1024, 1024, 1024, 512, 256, 128] and the activation function is ReLU [1]. Its output is the scalar replicated stability loss $\hat{\mathcal{L}}_s$. We observe this simple MLP is sufficiently accurate for stability regression and has numerically stable gradient, which benefits the back-propagation and refinement.

LSTM. In the ablation study, we adopt a LSTM [7] as an alternative architecture, which auto-regressively estimates the residual object center displacements at every $S = 10$ simulation steps

$$\begin{aligned} \mathbf{h}_1 &= \text{LSTM}(\mathbf{F}, \mathbf{0}) \\ \mathbf{h}_i &= \text{LSTM}(\text{MLP}_e(\mathbf{h}_{i-1}), \mathbf{h}_{i-1}), \quad i = 2, 3, \cdots, N \\ \Delta\hat{\mathbf{t}}_i &= \text{MLP}_d(\mathbf{h}_i), \quad i = 1, 2, \cdots, N \ , \end{aligned} \tag{1}$$

where $N = T/S = 10$, MLP_e and MLP_d represent learnable MLP encoder and decoder, with linear layer dimensions [1024, 1024, 1024] and [1024, 512, 256, 128] respectively, and $\mathbf{h}_i \in \mathbb{R}^{1024}$ represents the hidden state. The output $\Delta\hat{\mathbf{t}}_i \in \mathbb{R}^3$ represents the residual object center displacement at each step, so that the final replicated stability loss can be written as

$$\hat{\mathcal{L}}_s = ||\sum_{i=1}^{N} \Delta\hat{\mathbf{t}}_i|| \ . \tag{2}$$

To train such DeepSim network, we impose an approximation loss as

$$\mathcal{L}_a = ||\sum_{i=1}^{N} \Delta\hat{\mathbf{t}}_i - \Delta\mathbf{t}_i|| \ , \tag{3}$$

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

where $\Delta \mathbf{t}_i$ represents the residual object center displacement returned by the simulator. In practice, we observe that the success rate inferred by the LSTM DeepSim is roughly three times of the actual success rate evaluated by the simulator, which indicates an overfit that the LSTM DeepSim tends to predict small displacements. Hence this architecture does not outperforms the MLP counterpart.

2.2 Direct Gradient Calculation

Finite Difference. We use the MuJoCo [9] official implementation `mjd_transitionFD` to calculate the state gradient $\partial \mathbf{s}_{t+1}/\partial \mathbf{s}_t$, where $\mathbf{s}_t = [\mathbf{q}_t, \dot{\mathbf{q}}_t]$. The gradient of the final state with respect to the initial state is therefore the multiplication of each state gradient matrix. We set the perturbation at each component of the state as $\epsilon = 1 \times 10^{-4}$. Since the hand is static, we only perturb object states.

NimblePhysics. NimblePhysics [10] provides slightly different functions compared to the MuJoCo simulator. In particular, for physics modeling, we adopt the MANO [8] URDF model from [4]. We further set the object mass and inertia following the previous annotations[1]. Note that NimblePhysics does not provide analytical gradient for penetration resolving and implementation for adhesion force. Hence it favors inter-penetration between the hand and the object to achieve stability. In addition, NimblePhysics takes around 120 hours for an epoch on the DexYCB training set, while MuJoCo takes only 2 hours thanks to the underlying concurrent implementation. In consequence it is both numerically unstable and computationally intractable to apply NimblePhysics in our method.

3 Joint Refinement for Base and DeepSim Network

Akin to the discriminator in GAN [5], the DeepSim network determines the stability of the initial estimation from the base network (acts as a generator) by learning from the simulator. However, as there is no adversary relationship between the base and DeepSim network, joint training both networks can be made much simpler. Specifically, we train the base and DeepSim network in an alternating order as shown in Algorithm 1 (for one sample). To avoid overfitting and misleading by incorrect regression, we additionally perform data augmentation and loss masking as described in below. For clarification, $\mathbf{t}_t \in \mathbf{q}_t$ represents the object center position at time t (as a component of the full configuration), where $\mathbf{q}_0$ is the initial configuration and $\mathbf{q}_T = f_s(\mathbf{q}_0)$ is the final configuration after the simulation. We set the threshold $S = 0.01$, which classifies if the sample is stable or not.

Algorithm 1 Joint training of the base and DeepSim network.

for number of training iterations **do**
  for sample in minibatch **do**
    // Train DeepSim network
    Evaluate initial configuration: $\mathbf{q}_0 = f_b(\mathbf{I})$
    Perform random augmentation: $\mathbf{q}_0^r = \mathbf{q}_0 + \Delta\mathbf{q}, \quad \Delta\mathbf{q} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
    Merge all data: $\mathbf{q}_0^a = \{\mathbf{q}_0, \mathbf{q}_0^r, \mathbf{q}_0^{gt}\}$ ▷ $\mathbf{q}_0^{gt}$ represents ground truth pose for $\mathbf{I}$
    Evaluate stability loss: $\mathcal{L}_s^a = ||\mathbf{t}_T^a - \mathbf{t}_0^a||$
    Regress replicated stability loss: $\hat{\mathcal{L}}_s^a = f_d(\mathbf{F}^a)$ ▷ $\mathbf{F}^a$ represents the feature for $\mathbf{q}_0^a$ (Section 2.1)
    Update DeepSim network with approximation loss: $\mathcal{L}_a = ||\mathcal{L}_s^a - \hat{\mathcal{L}}_s^a||$
    // Train base network
    Re-evaluate replicated stability loss: $\hat{\mathcal{L}}_s = f_d(\mathbf{F})$ ▷ $\mathbf{F}$ is the feature for $\mathbf{q}_0$
    Extract relevant stability loss: $\mathcal{L}_s = ||\mathbf{t}_T - \mathbf{t}_0||, \quad \mathcal{L}_s^{gt} = ||\mathbf{t}_T^{gt} - \mathbf{t}_0^{gt}||$
    if ($\hat{\mathcal{L}}_s < S$ and $\mathcal{L}_s < S$ **or** $\hat{\mathcal{L}}_s \geq S$ and $\mathcal{L}_s \geq S$) **and** ($\mathcal{L}_s^{gt} < S$) **then**
      Update base network with replicated stability loss $\hat{\mathcal{L}}_s$
    end if
  end for
end for

4 More Results

In this section, we show more results in Figure 1 of the refined pose estimations and the corresponding simulation results. We observe that our method can produce physically stable results covering various

[1]We obtain physics properties of YCB objects from two open source projects (*i*) and (*ii*).

Figure 1: **More qualitative results.** We render with object textures[2] in the YCB [2] dataset and compare the resulting estimation (transparent, zoom in) and its corresponding configuration after simulation in the last column. The results are physically stable as the simulation displacements are small. More samples with complete simulation processes can be found in the supplementary video.

object categories and grasping approaches. When forwarding the estimations to the simulator, we observe minor simulation displacement (in the last column), indicating desired pose stability. Rotated views also show that the estimated poses are physically plausible. We additionally include a supplementary video that records the entire simulation processes for selected samples.

In addition, we show in Figure 2 that our method can generalize to various types of hand-object interaction, such as holding the object on the palm. Furthermore, our method can be applied to non-grasping samples without biasing towards predicting firm grasping, thanks to the joint accuracy-stability training.

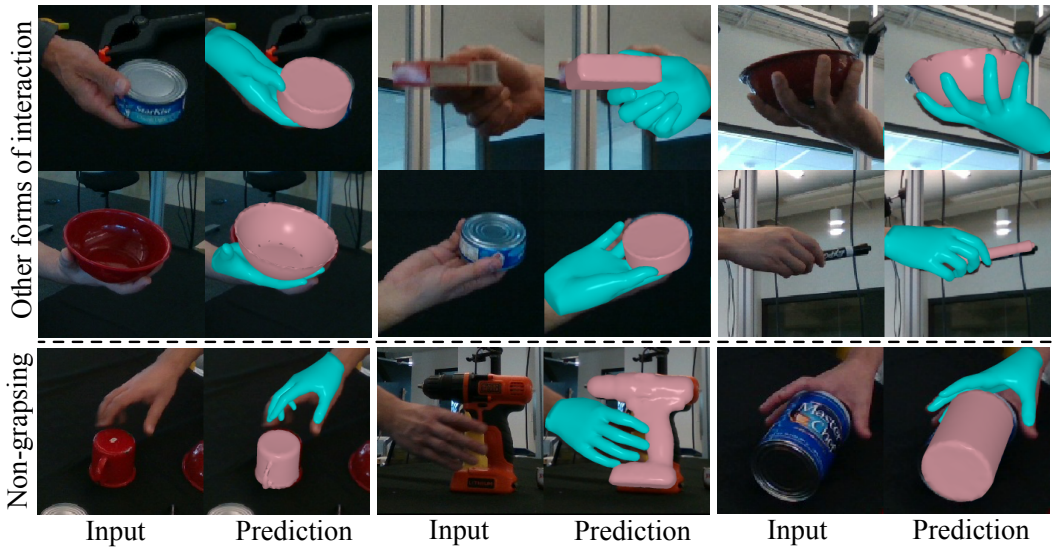

Figure 2: **More qualitative results.** Our methods can generalize to various forms of interaction, *e.g.* holding, and even non-grasping samples, without biasing towards firm grasping.

5 Failure Cases

We show in Figure 3 to illustrate the example of failure cases. Due to the occlusion ambiguity, our method may reach sub-optimum where the hand-object pose is stable but do not align with the ground truth on the contact points. Such cases can affect the estimation accuracy. Furthermore, due to imperfect simulation, especially in the collision detection, some outcomes reported to be stable by the simulator may not appear to be perfect stable in real-world. We encourage future researches to explore on enhanced physics simulation to tackle this limitation.

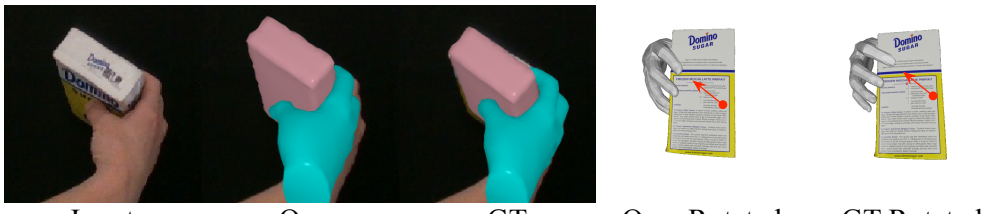

Figure 3: **Failure case of our method.** While our method can produce stable grasping, due to the occlusion ambiguity the grasping may not align with the ground truth (GT). The generated grasping may also not be the most natural way to grasp the target object.

[2]We find the HO3D dataset using a different object texture to the official YCB dataset (*e.g.* the second last sample). However, this does not affect the pose estimation (see in the Camera View).

References

[1] Abien Fred Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018. 1

[2] Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. The ycb object and model set: Towards common benchmarks for manipulation research. In *2015 international conference on advanced robotics (ICAR)*, pages 510–517. IEEE, 2015. 3

[3] Yu-Wei Chao, Wei Yang, Yu Xiang, Pavlo Molchanov, Ankur Handa, Jonathan Tremblay, Yashraj S Narang, Karl Van Wyk, Umar Iqbal, Stan Birchfield, et al. Dexycb: A benchmark for capturing hand grasping of objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9044–9053, 2021. 1

[4] Sammy Christen, Muhammed Kocabas, Emre Aksan, Jemin Hwangbo, Jie Song, and Otmar Hilliges. D-grasp: Physically plausible dynamic grasp synthesis for hand-object interactions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[5] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020. 2

[6] Shreyas Hampali, Mahdi Rad, Markus Oberweger, and Vincent Lepetit. Honnotate: A method for 3d annotation of hand and object poses. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3196–3206, 2020. 1

[7] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 1

[8] Javier Romero, Dimitrios Tzionas, and Michael J Black. Embodied hands: Modeling and capturing hands and bodies together. *arXiv preprint arXiv:2201.02610*, 2022. 2

[9] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012. 2

[10] Keenon Werling, Dalton Omens, Jeongseok Lee, Ioannis Exarchos, and C Karen Liu. Fast and feature-complete differentiable physics for articulated rigid bodies with contact. *arXiv preprint arXiv:2103.16021*, 2021. 2