Manipulation as in Simulation: Enabling Accurate Geometry Perception in Robots

Minghuan Liu[1,*,†], **Zhengbang Zhu**[1,2,*], **Xiaoshen Han**[1,2,*], **Peng Hu**[1,*], **Haotong Lin**[1], **Xinyao Li**[2], **Jingxiao Chen**[1,2], **Jiafeng Xu**[1], **Yichu Yang**[1], **Yunfeng Lin**[2], **Xinghang Li**[3], **Yong Yu**[2], **Weinan Zhang**[2], **Tao Kong**[1], **Bingyi Kang**[1,†]

[1]ByteDance Seed, [2]Shanghai Jiao Tong University, [3]Tsinghua University

[*]Equal Contribution, [†]Corresponding authors

Abstract

Modern robotic manipulation primarily relies on visual observations in a 2D color space for skill learning, but suffers from poor generalization. In contrast, humans, living in a 3D world, depend more on physical properties—such as distance, size, and shape—than on texture when interacting with objects. Since such 3D geometric information can be acquired from widely available depth cameras, it appears feasible to endow robots with similar perceptual capabilities. Our pilot study found that using depth cameras for manipulation is challenging, primarily due to their limited accuracy and susceptibility to various types of noise. In this work, we propose Camera Depth Models (CDMs) as a simple plugin on daily-use depth cameras, which take RGB images and raw depth signals as input and output denoised, accurate metric depth. To achieve this, we develop a neural data engine that generates high-quality paired data from simulation by modeling a depth camera's noise pattern. Our results show that CDMs achieve nearly simulation-level accuracy in depth prediction, effectively bridging the sim-to-real gap for manipulation tasks. Notably, our experiments demonstrate, for the first time, that a depth-only policy trained exclusively in simulation generalizes seamlessly to real robots on two long-horizon tasks, with little to no performance degradation. Further analysis reveals that sim-to-real success rates are strongly correlated with the quality of depth perception. We hope our findings will inspire future research in utilizing simulation data and 3D information in general robot policies. We release the datasets and models for three depth sensors, along with an easy-to-use guidance at manipulation-as-in-simulation.github.io.

Date: August 19, 2025
Correspondence: Minghuan Liu at ericliuof97@gmail.com, Bingyi Kang at bingyikang@bytedance.com
Project Page: xxx
Note: Minghuan did this work while at ByteDance Seed.

1 Introduction

Manipulation is a fundamental capability expected of robots, primarily involving skilled interactions with diverse objects, and thus necessitating visual observations. Recent advances show that robots can perform various tasks using 2D color images from single or multiple viewpoints [1, 4, 9, 21, 24, 40, 50, 60]. While color images provide rich semantic information, humans operate in a 3D world and rely on geometric cues—such as

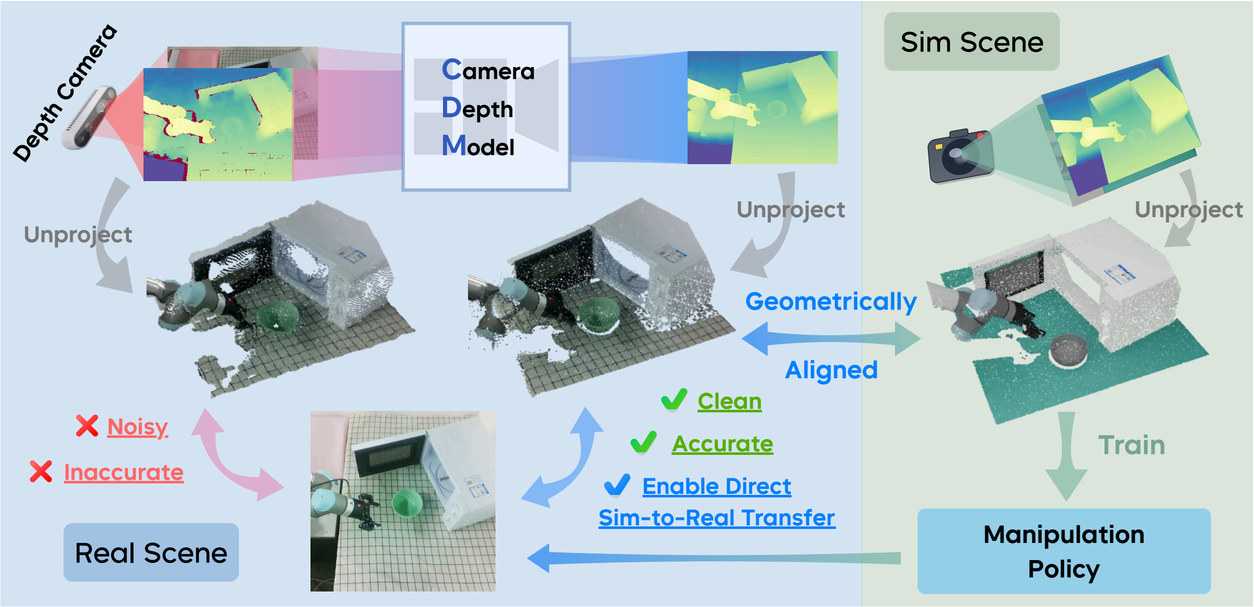


Figure 1 How the proposed camera depth model (CDMs) makes real-world manipulation as in simulation. The illustration is made on the RealSense D435 camera with CDM-435. With CDM, the manipulation policy learns from accurate geometric information, which is aligned between the simulation and the real world.

shape and spatial relationships—to distinguish objects (e.g., bottles versus bowls) and comprehend the skills required. This reliance on geometry, rather than texture, enables functional inference and precise interaction with objects.

With the widespread availability of depth cameras, acquiring 3D geometric information appears to be straightforward, suggesting that robots could be endowed with similar perceptual capabilities [8, 28, 53, 59]. However, their unreliable output, frequent mode failures, and sensitivity to noise pose significant challenges. Although recent studies have integrated 3D representations into robotic manipulation, performance remains limited by the poor quality of depth data produced by such devices. Consequently, evaluations are typically restricted to simulation environments [61, 62], where clean and accurate depth is available; or rely on downsampled point clouds [19, 58, 59] to mitigate noise in real-world scenarios. As illustrated in Fig. 1, real-world depth camera data often contains significant and characteristic noise artifacts, resulting in inaccurate perception of objects and environments by robots.

To mitigate the fundamental problem of depth perception and bring accurate geometry into robotic manipulation, this paper proposes camera depth models (CDMs), a plug-in solution for depth cameras that enhances geometric accuracy, as illustrated in Fig. 1. A CDM processes RGB images and noisy depth signals from a specific depth camera to produce high-quality, denoised metric depth. To train such models, we developed a multi-camera mount and collected a dataset of RGB-depth pairs from seven cameras across ten depth modes. Leveraging both this dataset and open-source simulated data, we designed a neural data engine that models the noise patterns of depth cameras to generate high-quality paired data in simulation. To address the scale mismatch in synthesized noise, we propose a novel guided filter approach for noise augmentation. CDMs achieve nearly simulation-level 3D accuracy, effectively bridging the sim-to-real geometry gap from the real-world perspective.

Our experiments evaluate CDMs in real-world imitation and sim-to-real manipulation tasks. We show that CDMs enable robot policies to learn generalizable skills from accurate geometric information. Notably, we demonstrate, for the first time, that a depth-only policy trained entirely in simulation without real-world fine-tuning can transfer seamlessly to real robots on two long-horizon tasks, with minimal to no performance degradation. These results highlight the potential of CDMs to leverage simulation and underscore the importance of accurate geometric data in developing robust, generalizable robot policies.

In a nutshell, the contributions of this paper are mainly threefold:

1. We introduce ByteDepth, a real-world multi-camera depth dataset comprising over 170,000 RGB-depth pairs from ten distinct configurations captured by seven depth cameras.
2. We proposed and released a family of camera depth models (CDMs), a plug-in solution that enhances depth perception accuracy for widely used depth cameras.
3. Through CDMs, we demonstrate how the sim-to-real geometry gap can be bridged, highlighting the critical role of accurate geometric information in robotic manipulation tasks.

2 Related Work

2.1 Metric Depth Prediction

Recent advances in depth-fundamental models, such as the Depth Anything (DA) series [54, 55], have substantially improved the estimation of scene geometry and high-resolution relative depths across diverse open-world images, demonstrating robust generalization. However, most real-world applications require accurate metric depth rather than relative depth. Simply fine-tuning DA models to predict metric depth [55] remains constrained by a fixed depth scale and is susceptible to scale ambiguities [18, 56]. Although recent approaches [44, 46] introduce affine-invariant techniques to train relative models on large-scale datasets and achieve improved metric depth estimation via post-processing with a prompt depth, the fundamental scale ambiguity in monocular images, especially for scenes with large depth ranges, remains unresolved. To address this, many recent approaches choose to incorporate explicit scale cues for predicting metric depths. For instance, Guizilini et al. [12] and Piccinelli et al. [34] introduce camera intrinsics into the model. Lin et al. [27] and Wang et al. [47] proposed a more straightforward way that directly integrates scale information by prompting paradigms, *i.e.*, low-quality depth images or sparse LiDAR signals, into the model's architecture of the pre-trained DA model, and finetuned the model on RGBD datasets with handcrafted prompt depth images on synthesized data. Nevertheless, they are limited in prompt images made with the style of handcrafted rules and are hard to work well on diverse sensor configurations for dynamic scenes. In addition to these solutions with a depth prompt, there are some works focused on recovering metric depth (disparity) from stereo RGB images [49], which provide implicit depth cues through disparity, but they often require careful calibration and are limited in diversity. Such methods are limited to working on stereo cameras (and with RGB only) and require the precise camera intrinsics (baseline distance, focal length, and so on) to obtain the depth.

2.2 Manipulation with 3D Representation

Robotic manipulation involves skillfully interacting with objects, which necessitates accurate perception of their states. Classical planning-based approaches typically depend on a calibrated perception module to identify the 3D positions of relevant objects, which are then used to plan feasible manipulation paths. For example, Fang et al. [8] predicts grasp poses based on point clouds captured by a depth camera.

Learning-based methods, on the contrary, focus on modeling autonomous robot policies using neural networks. Most recent works rely on RGB images, ranging from single-view [4, 5, 21] to multi-view setups [9, 60], and from single-task policies [4, 60] to multi-task generalist policies [1, 21, 24, 40, 50]. Although these approaches have achieved impressive progress on various tasks, they often struggle to generalize to various visual conditions. To better capture the 3D structure of the environment and leverage geometric information, some recent works have started incorporating 3D representations into robot policies. For example, Ze et al. [58, 59] and Hua et al. [19] use point clouds to improve policy generalization across objects with similar shapes but varying textures and backgrounds. However, these methods still require point cloud downsampling, calibration, and table-top cropping to mitigate noise from depth cameras. Liu et al. [28] segment objects and trains a depth-only policy for loco-manipulation tasks to address the sim-to-real gap. Zhen et al. [61] train a 3D-aware robot foundation model that accepts modalities such as depth, point clouds, and 3D bounding boxes, but their experiments are limited to simulation, where perfect depth is available. Similarly, Zhu et al. [62] compares different visual representations in the simulation and demonstrates the clear advantages of explicit 3D representations with perfect 3D perception. Furthermore, some works explore more complex representations, such as neural radiance fields [26, 53] and dense voxels [38, 57]; however, all of these 3D representations ultimately depend on transforming the original camera depth into real-world scenarios.

2.3 Visual Sim-to-Real

Sim-to-real transfer requires policies to overcome both the observation gap and the physics gap between simulated and real-world environments. Physics gaps, such as discrepancies in dynamics or friction, are often addressed by using domain randomization [33, 41], which trains policies to generalize across a range of physical parameters that encompass real-world variability. While this method is generally effective for locomotion tasks in legged robots [22, 52], manipulation tasks require more precise modeling because they depend heavily on accurate visual observations of objects, typically provided by RGB and/or depth images. Achieving robust sim-to-real policy transfer for manipulation tasks using RGB images requires high-fidelity simulation rendering to minimize the visual gap. Relying solely on simulator-generated images often demands extensive curriculum and augmentation design to ensure that the learned policy is effectively transferred to the real world [42]. Although advances in simulator rendering technologies [31, 51] can help, recent real-to-sim approaches using neural rendering techniques demonstrate that reconstructing photorealistic scenes from real-world data can further reduce the visual gap [13, 25, 29, 35].

In addition to RGB images, some works utilize colorless 3D representations, such as point clouds and depth images, to reduce visual discrepancies between simulation and reality. For example, He et al. [16] predicts the ray distances from simulated depth images to learn the reach-avoid value networks; Cheng et al. [3], Zhuang et al. [63], Zhuang et al. [64], and Lai et al. [23] employ depth images to train quadrupeds for collision avoidance and high dynamic locomotion; and Liu et al. [28] uses depth images from two camera views for mobile manipulation tasks on a quadruped robot, which requires object segmentation to further narrow the sim-to-real gap. These approaches typically add noise and augmentations to simulated depth images and require post-processing of real-world depth data, such as clipping, hole filling, and temporal filtering, to address sensor imperfections. Alternatively, Hua et al. [19] uses point clouds as visual input, but still adds noise in simulation and applies cropping and downsampling in the real world. Besides, Tao et al. [39] introduces a computation-cost method to simulate the depth with typical noise patterns rendered by a real-world stereo camera. However, simulating a real-world camera or adding noise in the simulation is a last resort, as it may deteriorate the rich geometry information and precise manipulation.

3 Camera Depth Models

Existing depth foundation models can estimate proper relative depth without a geometric prior. However, for real-world manipulation tasks, models must predict absolute metric depth. This requires two key capabilities: 1) identifying semantically meaningful local regions for objects and backgrounds in RGB images, and 2) assigning accurate metric depths to these regions using coarse depth prompts from camera depth images. Notably, this task extends beyond simple denoising or depth completion, as raw depth readings from various depth cameras exhibit diverse working ranges, failure modes, noise patterns, and biases. Fig. 4 provides a glimpse of the noisy depth images produced by consumer-grade depth cameras.

3.1 Noise from Depth Cameras

We categorize two general types of noise, *i.e.*, the value noise and the hole noise, as depicted in Fig. 2-left. Intuitively, hole noise manifests as missing data in depth readings, often caused by depth estimation algorithms (e.g., stereo matching) or environmental factors, such as lighting or material properties. Value noise encompasses all other inaccuracies, including biases specific to each camera, as well as blur, jitter, and other distortions. For instance, stereo matching-based cameras often produce holes around object boundaries, while LiDAR-based cameras struggle with black or highly reflective surfaces. Both types perform poorly on transparent or mirror-like objects, such as glass. These noise patterns depend on the camera's intrinsic parameters and physical installation.

Therefore, develop an effective metric depth model for a specific camera, the model must 1) refine coarse, low-quality depth prompts from the camera into precise metric depth estimates, while 2) correcting faulty

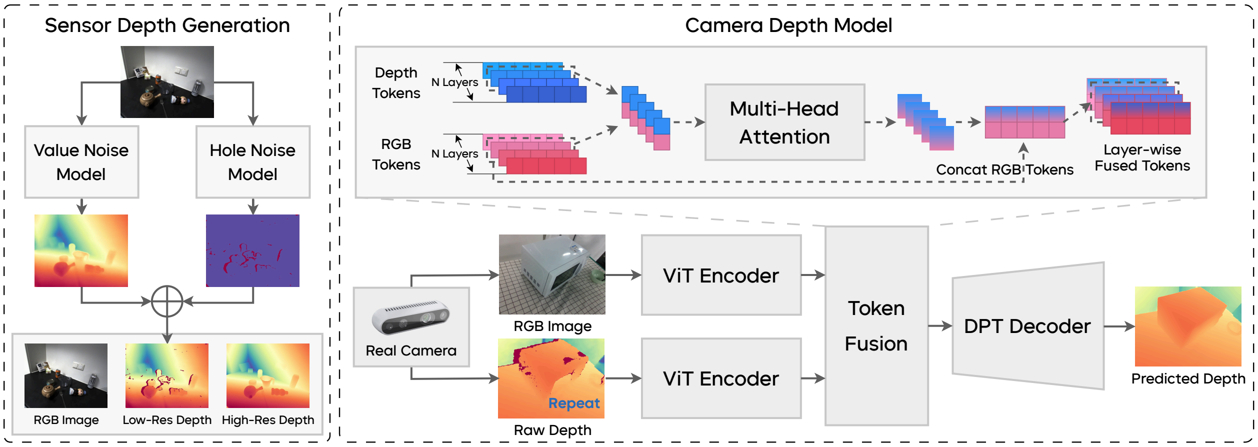


Figure 2 Overview of camera depth models. Left: camera depth generation to synthesize the datasets for training camera depth models, where the value/hole noise models are trained using the collected dataset. (Sec. 3.4) Right: the camera depth model, which is built on two ViT encoders [7] and fine-tuned from a depth foundation model [55]; the RGB and depth tokens are fused before being given to a DPT decoder [36]. Such a structure allows the model to receive sparse depth from sensors for prediction without any pre-processing like hole-filling. (Sec. 3.2)

depth readings by leveraging semantic information from RGB images. Balancing reliance on sensor data with skepticism of its inaccuracies poses a significant challenge, making a generalizable solution nontrivial. This motivates the development of camera-specific depth models (CDMs) tailored to individual depth cameras.

3.2 Model Design

We designed our CDMs M for specific depth cameras to take a pair of an RGB image $\mathbf{I} \in \mathbb{R}^{3\times H\times W}$ and a depth image $\mathbf{D} \in \mathbb{R}^{H\times W}$ from the depth camera, and predict a high-quality metric (absolute) depth image $\hat{\mathbf{D}} \in \mathbb{R}^{H\times W}$. The proposed model structure of CDMs is illustrated in Fig. 2-right. In particular, we design a dual-branch ViT [7] architecture to achieve the above-mentioned capabilities by separately capturing semantic information from the RGB and the depth images, as well as scale information that is cross-modal but aligned in feature tokens X:

$$X^{\mathbf{I}} = \mathrm{ViT}^{\mathbf{I}}(\mathbf{I}), X^{\mathbf{D}} = \mathrm{ViT}^{\mathbf{D}}(\mathbf{D}) \ , \tag{1}$$

where $X^{\mathbf{I}} = \{X^{\mathbf{I}}_1, \cdots, X^{\mathbf{I}}_N\}$ and $X^{\mathbf{D}} = \{X^{\mathbf{D}}_1, \cdots, X^{\mathbf{D}}_N\}$ are feature tokens encoded by the RGB branch and the depth branch, separately.

Subsequently, we fuse these two types of information through a feature token fusion module. Since the token fusion module primarily serves to augment semantic information with scale information, it is only necessary to fuse tokens corresponding to the same spatial locations. Based on this, the fusion module only performs self-attention on corresponding tokens to accomplish bidirectional feature fusion, and results in depth features $\tilde{X}$ imbued with scale information:

$$\tilde{X} = \sum_i \mathrm{MHA}(\{[X^{\mathbf{I}}_i; X^{\mathbf{D}}_i]\}^N_{i=1}) \ , \tag{2}$$

where MHA stands for multi-head attention, and [;] is the concatenation operation. The entire fusion process occurs across multiple levels of feature tokens, allowing for deeper integration and the ability to incorporate both global- and local-scale information, especially when the camera depth prompt has large missing regions, where global-scale information is particularly needed.

Additionally, we concatenate the original RGB feature tokens into the fused feature tokens. These fused feature tokens, along with the RGB feature tokens, are concatenated to prevent loss of semantic information, and then passed through a DPT head [36] to produce scale-aware depth estimation results D.

$$\hat{\mathbf{D}} = \mathrm{DPT}([X^{\mathbf{I}}; \tilde{X}]) \ . \tag{3}$$

Compared with previous works that fuse the prompted depth information simply in the shallow decoding phase [27, 47], our proposed CDM structure provides a much more informative representation of the depth feature and its alignment with the RGB feature, and thus is able to perceive the raw depth image directly, without preprocessing such as hole-filling [27, 47]. Through the simple but representative structure design, CDM simplifies the inference procedure, provides the metric depth, and works as a simple plugin after the camera input, thereby fulfilling the three desiderata mentioned in the beginning.

3.3 ByteCamDepth: A Multi-Camera Depth Dataset

To train our CDMs, we will need a dataset that contains triplets, *i.e.*, RGB image I, low-quality depth image D, and ground-truth depth image $\overline{\mathbf{D}}$. However, the low-quality depth images, although they have usually been handcrafted by adding typical noise patterns to ground-truth depth images that are simulated by basic depth measurement principles, other factors are hard to be accurately modeled in the simulation, for instance, the camera parameters, the implementation and optimization details in each depth camera hardware and software are case by case, resulting distinct noise behaviors. On the contrary, we can easily collect low-quality depth data from real sensors, but it is hard to get perfect depth data. Therefore, naturally, we propose to learn the noise pattern with neural networks automatically from real-world data, and then synthesize the noisy low-quality depth image with the learned noise models.

To this end, we collect typical depth patterns and construct a dataset for various depth cameras that are commonly used in daily robot experiments. Specifically, our dataset spans 10 depth modes from 7 different depth cameras, including different stereo and lidar cameras. To achieve highly efficient data collection, we design a multi-camera mount device to capture data simultaneously, as illustrated in Fig. 3. Our datasets contain more than 17,000 images for each camera, sampled from videos at 5Hz, covering 7 different scenes including kitchens, living rooms, markets, bedrooms, bathrooms, offices, and breakrooms, as shown in Fig. 4.

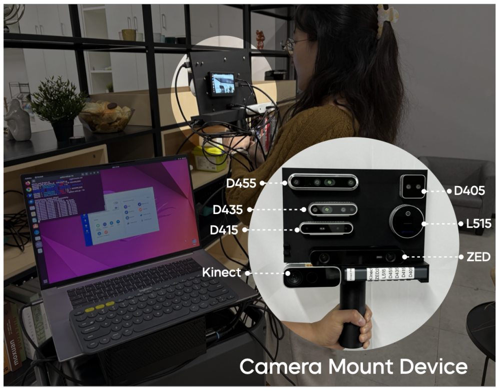


Figure 3 Multi-camera mount device for capturing the color-depth image pairs from multiple depth cameras all at once. We mount seven cameras, including five RealSense cameras (D405, D415, D435, D455, L515), a ZED camera, and an Azure Kinect camera. For the Zed camera, we record the raw data and replay it with its 4 modes (performance, ultra, quality, neural) offline. In practice, we use two computers to capture all the data due to the USB bus bandwidth limits.

3.4 Data Synthesis with Noise Models

We train two noise models on our collected depth dataset for each camera, which are then used for generating stylized low-quality depth images on open datasets to train CDMs.

Hole noise model. We treat the hole noise prediction as a binary-class prediction given the RGB image I, thereby training the hole noise model N_{hole} as a pretrained DINOv2 backbone [32] with a DPT head [36] to predict the dense mask (*i.e.*, hole/non-hole) for each pixel on the camera depth image $\mathbf{D}$. Formally, this corresponds to optimizing the following objective:

$$\ell(N_{\text{hole}}(\mathbf{I})) = \sum_{i=0,j=0}^{i=H,j=W} \left[y_{i,j} \log \sigma(x_{i,j}) + (1 - y_{i,j}) \log(1 - \sigma(x_{i,j}))\right] \ , \tag{4}$$

where $x_{i,j} = N_{\text{hole}}(\mathbf{I})_{i,j}$ denotes the i,j-th pixel on the mask image predicted by the hole noise model N_{hole}, $y_{i,j} = \mathbb{I}(\mathbf{D}_{i,j} = 0)$ denotes if the i,j-th pixel corresponds to a hole, and σ is the sigmoid function.

Value noise model. Motivated by the fact the depth foundation models are predicting a clean style depth image, we regard the value noise prediction as a stylized relative depth prediction problem, thereby turning to the help of Depth Anything V2 [55] (DAV2) by taking the low-quality depth image as the labels for prediction. The training objective of the value noise model N_{value} is simply an L_1 loss of the predicted depth $\hat{\mathbf{D}}_{\text{hole}} = N_{\text{hole}}(\mathbf{I})$ and a normalized ground truth depth $\hat{\mathbf{D}}$:

$$\ell(N_{\text{hole}}) = L_1(f(\overline{\mathbf{D}}), \hat{\mathbf{D}}_{\text{hole}}) \ , \tag{5}$$

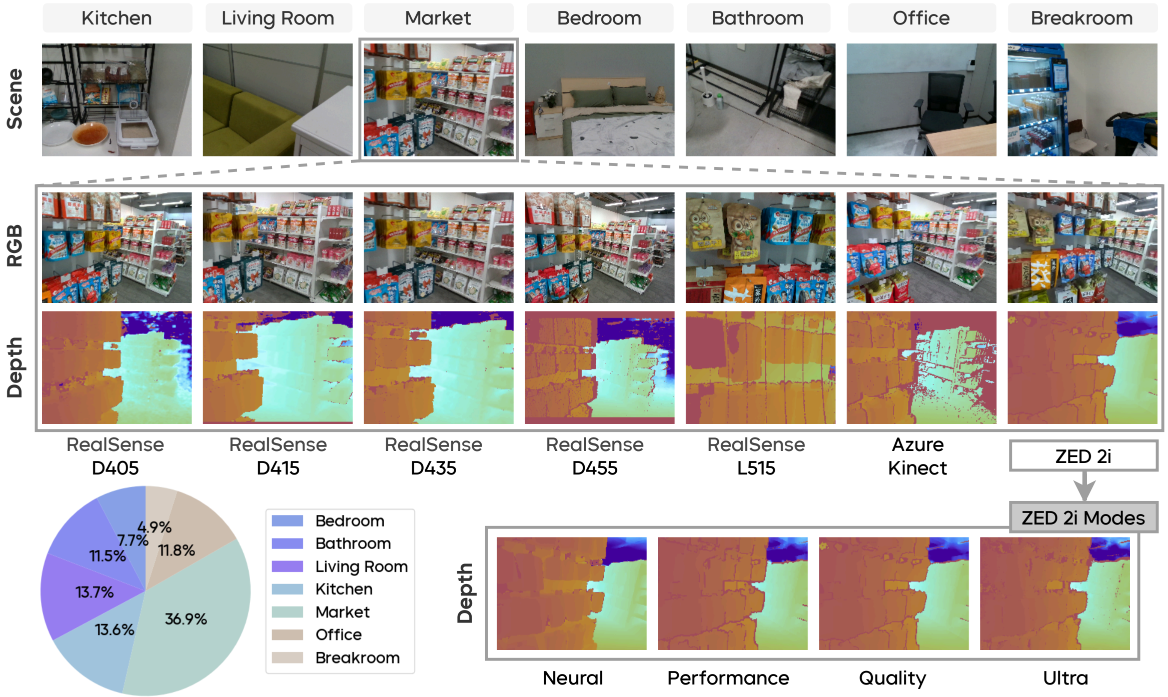


Figure 4 Illustration of the collected ByteCamDepth datasets, which contains the raw depth data from 7 cameras, 10 modes (including 4 modes of the ZED 2i camera) in 7 different scenes.

where f is the normalization function, in our project we use the affine-invariant transformation proposed in Wang et al. [45]. To make sure the value noise can learn proper relative scales during the data synthesis stage, we also fine-tune the DAV2 model on the synthesized dataset before turning it into a value noise model.

Synthesizing camera depth. Having the noise models, we can synthesize the noisy low-quality data on open synthesized datasets with clean ground truth depth. Denote the hole noise model as H and the value noise model as V, we cured the synthesized noisy data $\tilde{D}$ given the RGB image I via:

$$\tilde{\mathbf{D}} = \mu(V(\mathbf{I})) * (H(\mathbf{I}) < 0.5) \ , \tag{6}$$

where μ is the affine-invariant unscaling function [45] recovering the metric of the predicted relative value noise, referring to the metric of the ground truth depth.

3.5 CDM Training

Although we can synthesize training data for specific cameras, and these two types of noise models can learn similar noise patterns compared to the raw depth obtained from the real depth camera, we observed several problems, especially with the value noise models.

Guided filter for value noise. The most significant one is that the value noise model is hard to maintain the correct metric scale on the synthesized datasets, after being fine-tuned on the ByteCamDepth dataset. This results in large metric gaps between the synthesized camera depth and the ground truth depth, and further leads the trained CDM to overlook the metric information contained in the camera depth prompt. We propose to employ the guided filter [15]. The guided filter assumes the output image B is a local linear transformation of the guided image G:

$$b_i = x_k g_i + y_k \ , \tag{7}$$

where b_i, g_i are the i-th pixel of B and G, respectively; x_k, y_k are the scale and shift parameters in the kernel window, which is obtained through optimizing the minimize the error of the transformation between the input image A and the output image B:

$$\sum_{i\in\omega_i}\left((x_k g_i + y_k - a_i)^2 + \epsilon x_k^2\right) \ , \tag{8}$$

where ϵ is a regularization term. Note that in our case, the guided filter takes the value noise as the guided image G and the ground-truth depth as the input image to be filtered A, so that the resulting image B can keep the geometry/structure of the value noise but maintain the roughly correct metric scale in the whole image. As the kernel size k grows, the output image keeps much more structure of the noise and less metric related to the ground truth depth. Therefore, we use a randomized kernel size k (from small to large) and take the guided filter as an augmentation strategy of the value noise before adding hole noise, which helps to get the best results. Besides, both noise models barely learn the pattern of high-frequency noises, due to the nature of the neural network and our DAV2-like training strategy, so we randomly add such noises by handcrafted rules as an augmentation strategy.

Training loss. Referring to Lin et al. [27], we use the L_1 loss combined with the gradient loss for better edge depth to train our CDMs M, given image $\mathbf{I}$ and its raw depth $\mathbf{D}$:

$$\ell(M) = L_1(\overline{\mathbf{D}}, \hat{\mathbf{D}}) + \ell_{\text{grad}}(\overline{\mathbf{D}}, \hat{\mathbf{D}}) \ , \tag{9}$$

$$\ell_{\text{grad}}(\overline{\mathbf{D}}, \hat{\mathbf{D}}) = \left(\left|\tfrac{\partial(\hat{\mathbf{D}}-\overline{\mathbf{D}})}{\partial x}\right| + \left|\tfrac{\partial(\hat{\mathbf{D}}-\overline{\mathbf{D}})}{\partial y}\right|\right) \ . \tag{10}$$

During our training, we use disparity as the training target. The weights of the ViT encoder in the RGB and the depth branch are both initialized from DINO-v2 [32], and the decoder is trained from scratch. For the single-channel depth images, by default, they are copied three times before being fed into the network. We synthesized our training data on four simulated datasets: HyperSim [37], DREDS [6], HISS [48], and IRS [43], in a total of 280,000+ images.

4 Sim-to-Real Manipulation through CDMs

Our camera depth models (CDMs) allow us to obtain a 'simulation-like' depth image in the real world, which provides accurate geometry information. Therefore, simulation data can be fully utilized to learn a manipulation policy, which can be seamlessly transferred to the real world. To evaluate the power of CDM and its benefits to robot manipulation, we develop a geometry-based sim-to-real pipeline that contains four main stages: scene construction, camera alignment, simulated data collection, and imitation learning. After, we directly deploy the trained policy onto real-world robots, with the corresponding CDM as an inference plug-in. In particular, we choose depth instead of pointclouds as the observation for the following reasons: 1) human relies on single-view stereo visual observations and can do many things; 2) pointclouds fused from multi-view cameras require careful camera calibrations and are more sensitive to irrelevant backgrounds.

Scene construction. Since we rely on depth-only visual transfer, without any color information, the geometry-based sim-to-real pipeline does not require rigorous alignment of the exact appearance between objects and backgrounds in the simulation and the real world. Instead, we introduce geometrically similar objects and construct simple geometry as the background in the simulation. After building the environment, we set up the camera to obtain visual observations. We note that manipulation also requires a reasonable approximation of interactions between robots and objects, yet this is beyond the focus of this project. Thereafter, we manually assign the physical attributes of objects and modify the open-source robot description files to ensure plausible interactions, without aiming for absolute physical accuracy.

Camera alignment. To align the camera pose between the simulation and the real scene, we adopt a differentiable-rendering-based camera calibration method [2] to estimate the camera extrinsics in the real scene with minimal human effort, which only requires a few corresponding masks of the robot arm between real and virtual scenes. However, the calibrated poses are hard to perfectly align due to the differences in camera models between simulation and real-world sensors. To mitigate the gap from such misalignment, we slightly randomize camera poses during data collection in simulation, which helps the policy be robust to small discrepancies in viewpoint and better work in real-world deployments.

Data generation. To generate demonstrations efficiently in simulation, we develop an extension of MimicGen [30], with whole-body control (WBC) [14] to generate smoother, high-quality demonstrations. Our method introduces several data generation features like random reset to generate retrying demonstrations, controllable velocity for fine-adjustment, and so on. With WBC, the algorithm can be further extended to wheeled robots for mobile manipulation tasks. Details of the algorithm can be further referred to Appendix G.

Imitation learning with depth. We adopt a policy structure similar to the one used in Hua et al. [19] and Lin et al. [27]. Although some previous works have shown some robustness using a point cloud based policy [19, 58, 59], their point clouds are also transformed from the depth image captured by the depth camera and requires cropping and downsampling to alliviate noises. Since now we can obtain an accurate depth where the geometry information is complete, we choose to use depth image directly as the input to our policy. The policy encodes the depth image with a pre-trained ResNet, where the first layer of the network is replaced with a 1-channel convolutional layer; the proprioceptive states are encoded by a from-scratch MLP; a diffusion head [4, 17] is adopted to predict the action sequence. We directly use the one-step single-view depth image rendered in simulation for training the policy, without adding any noise, but with only the `RandomShiftScaleRotate` augmentation to alleviate the camera calibration biases. Besides the depth image, the observation space also includes the joint position and the gripper status. In real-world deployment, we make our CDM a plugin between the camera and the policy, which predicts a clean depth image based on the raw depth image and the RGB image from the depth camera. The predicted depth image is then used for real-time policy inference.

5 Experiments

The experiments involved are mainly threefold.

- Does the camera depth model achieve better performance given specific types of low-quality depth?
- How does the accurate geometry information benefit real-world robot manipulation?
- How the "sim-like" geometry contribute to zero-shot sim-to-real robotics manipulation?

5.1 Depth Performance

We mainly evaluate the trained camera depth models (CDMs) on the Hammer dataset [20], a real-world dataset that contains warped depth data paired with RGB images collected by three depth sensors: the RealSense D435 (stereo depth based on active structure light and IR images), L515 (a D-Tof camera), and a Lucid Helios (an I-Tof camera). Note that the dataset is not used for training, showing a zero-shot performance. We compared our CDMs against two baseline methods, PromptDA [27] and PriorDA [47], both of which are metric depth prediction methods using prompt depth images and require hole-filling preprocessing during inference time. Since our CDM directly takes a prompt depth image as it is, we test two cases, *i.e.*, Filled and Holed, denoting whether the low-quality depth is filled or directly given to the model for prediction. We also compared PromptDA fine-tuned on our synthesized dataset to show the advantage of the structure design.

The results are shown in Tab. 1, where we can observe and conclude several things. 1) Both PromptDA and PriorDA failed to obtain good depth prediction without hole-filling preprocessing, which requires additional computation time in real-time robot experiments. 2) Even with hole-filling, our CDMs achieve the state-of-the-art performance on corresponding data splits. 3) With the same training data and augmentation strategy, our CDMs still perform better than PromptDA, showing the advantage of our designed structure. 4) The model trained on specific synthesized camera noise data should work better on the same camera data split, like PromptDA; however, to our surprise, the CDM-L515 generalized well to the 435 datasets and can even achieve slightly better results than the CDM-D435 model, indicating the test cases of the D435 camera in the Hammer dataset can be generalized by the CDM-L515. 5) Both CDMs have better zero-shot generalization ability on the data split with a different depth sensor (the I-Tof Lucid Helios camera), potentially because CDMs solve some common noise problems among depth cameras.

Split	Filled / Holed /	L1 ↓	RMSE ↓	AbsRel ↓	$\delta_{0.5}$ ↑	δ_1 ↑
D435 (IR Stereo)	**Ours (CDM-D435)**	**0.0258**	**0.0404**	**0.0312**	**0.9842**	**0.9951**
	Ours (CDM-L515)	**0.0182**	**0.0338**	**0.0217**	**0.9877**	**0.9956**
	PromptDA*(435)	0.0434	0.0666	0.0599	0.9459	0.9770
	PromptDA*(515)	0.1830	0.2387	0.2750	0.8802	0.9186
	PromptDA	0.1703	0.2971	0.2437	0.6704	0.7229
	PriorDA	1.2031	0.6856	1.2030	0.0837	0.1717
	PromptDA	0.0396	0.0691	0.0484	0.9503	0.9772
	PriorDA	0.0388	0.0754	0.0461	0.9632	0.9880
	Raw Depth	0.0550	0.1458	0.0708	0.9179	0.95429
L515 (D-Tof)	**Ours (CDM-L515)**	**0.0156**	**0.0297**	**0.0229**	**0.9754**	**0.9919**
	Ours (CDM-D435)	**0.0165**	**0.0349**	**0.0246**	**0.9613**	**0.9855**
	PromptDA*(515)	0.0235	0.0666	0.0349	0.9291	0.9730
	PromptDA*(435)	0.0254	0.0438	0.0379	0.9234	0.9640
	PromptDA	0.0483	0.0400	0.0612	0.8867	0.9259
	PriorDA	0.5412	0.6134	0.9211	0.0850	0.1794
	PromptDA	0.0207	0.0515	0.0304	0.9480	0.9699
	PriorDA	0.0177	0.0385	0.0274	0.9502	0.9763
	Raw Depth	0.0312	0.0813	0.0475	0.9098	0.9429
Helios (I-Tof)	**Ours (CDM-L515)**	**0.0248**	**0.0403**	**0.0334**	**0.9468**	**0.9871**
	Ours (CDM-D435)	**0.0272**	**0.0457**	**0.0372**	**0.9297**	**0.9806**
	PromptDA	0.0207	0.0515	0.0304	0.9480	0.9699
	PriorDA	0.0324	0.0597	0.0461	0.8984	0.9638
	Raw Depth	0.0312	0.0813	0.0475	0.9098	0.9429

Table 1 Quantitative comparisons of metric depths prediction on Hammer [20] dataset (zero-shot evaluation). The terms Filled. and Holed. refer to whether the low-quality depth is filled or directly given to the model for prediction. *(split) denotes fine-tuning on our synthesized datasets with the same augmentation strategy. Raw depth refers to the metric of directly using a low-quality depth image without a model. CDMs are named as the camera type, which are trained on the corresponding synthesized noise of that camera. All results are computed directly from the output of these models, without any alignment postprocessing.

Depth Model	Toothpaste-and-Cup		Stack-Bowls	
	Pick Toothpaste	Put Toothpaste into Cup	Pick Bowl	Stack Bowl
None	0/15	0/15	6/15	3/15
CDM-D435	**10/15**	**6/15**	**11/15**	**9/15**

Table 2 Depth-only imitation results w/w.o CDMs, each task with 50 demonstrations.

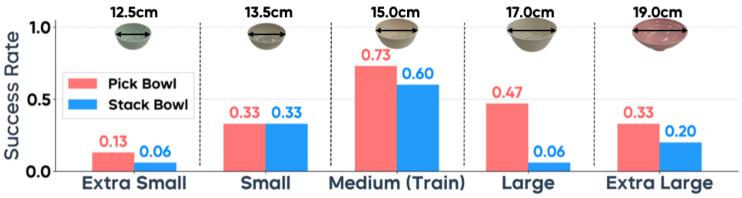
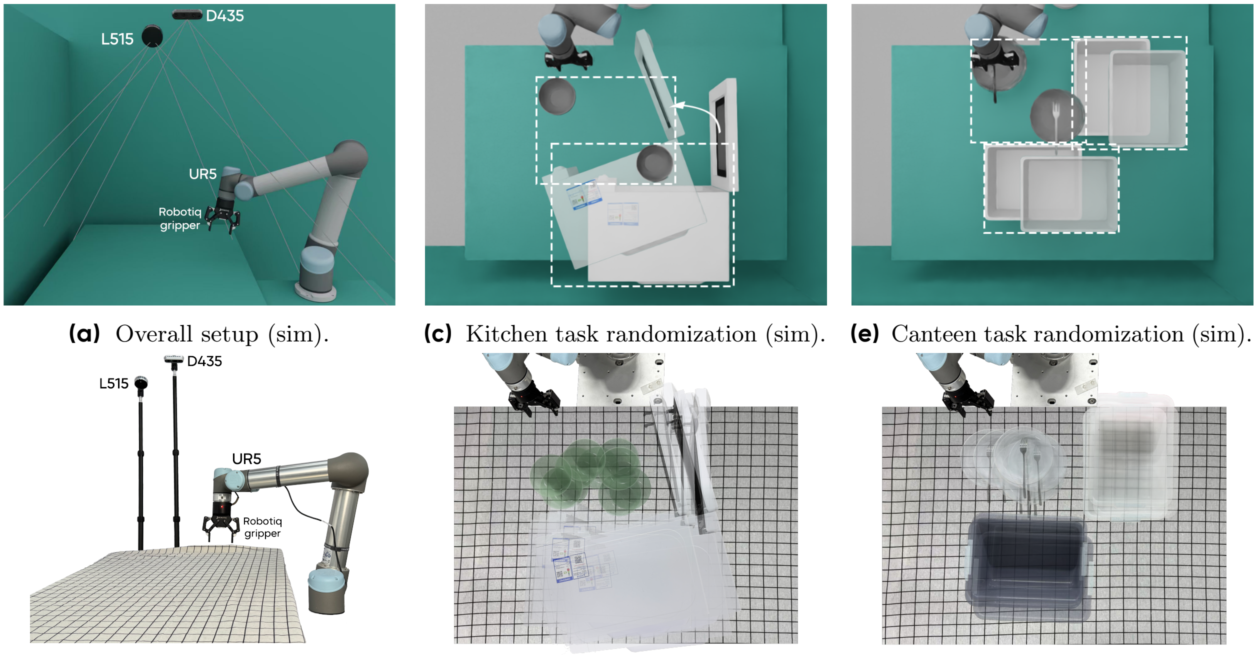


Figure 5 Generalization over different sizes of objects. The policy trained without CDM cannot generalize to unseen sizes.

5.2 Depth-Only Imitation Learning

We aim to investigate how the accurate geometry information produced by the camera depth models benefits robot manipulation tasks. To this end, we design a pilot study that contains two pick-and-place tasks on a daily-use depth camera, RealSense D435. The *Toothpaste-and-Cup* task requires the robot to pick the toothpaste into the cup, and the *Stack-Bowls* task requires the robot to stack two bowls. We manually collect 50 trajectories by teleoperation for each task, and conduct the test on five different positions, each with three trials. In particular, for the *Stack-Bowls* task, we trained our policy on a normal-sized bowl and tested on bowls of five different sizes, including four unseen sizes. Since we do not involve color information in the policy, the policy naturally generalizes to various colors, so we ignore the difference in texture. The results are shown in Tab. 2 and Fig. 5. We can easily observe that, learning and inference with the high-quality depth data produced by the CDM highly improve the policy's ability to achieve tasks. And it is worth noting that with the accurate geometry information, the policy can generalize to bowls of different sizes, which is nontrivial for the policy trained without CDM.



(a) Overall setup (sim). **(c)** Kitchen task randomization (sim). **(e)** Canteen task randomization (sim).

(b) Overall setup (real). **(d)** Kitchen task distribution (real). **(f)** Canteen task distribution (real).

Figure 6 UR-5 manipulation experiment setup. (a)(d) The overall setup for simulation and the real-world experiments, where we test two different cameras (D435 and L515) and train single-view policies for both cameras. In simulation, we add small randomization to the camera pose when generating demonstrations. (c)(e)The training environments of the kitchen and the canteen tasks in the simulation, where the dotted frame and the arrow denote the randomization range of objects. (d)(f) The test distributions of the two tasks, where we compose the position of the bowls and microwaves, follow the randomization boundaries in the simulation.

5.3 Zero-Shot Sim-to-Real Manipulation

Robot and task setup. We construct our sim-to-real pipeline using a tabletop UR5 robot equipped with a Robotiq gripper, as illustrated in Fig. 6b. As mentioned before, the visual observation of the policy is the depth image from a single third-view camera. We design two long-horizon manipulation tasks: the kitchen task and the canteen task. 1) *The kitchen task* tests the ability to utilize articulation objects: the robot is required to pick up a bowl on the table, put it into the microwave, and then close the door of the microwave. Note that the microwave door is glass, which is seen as a hole from the original camera depth, and poses an additional challenge for depth capturing. 2) *The canteen task* requires recognizing and accurately grasping a slim fork and a thin plate, which are rather noisy, and the fork is even unseen from the original camera depth: the robot should pick up the fork and put it into the box in front of it, then pick up the plate and dump the trash into the left box, at last place the plate into the front box. We collect $\sim$ 680 demo trajectories for the kitchen task and $\sim$ 800 for the canteen task in simulation, train a policy by imitation learning, and directly deploy the policy in the real world by plugging in a depth model. The test setup is illustrated in Fig. 6d and Fig. 6f, where we test 10 positions and each for 3 times, resulting in 30 tests in total for both tasks. In the real world, we test two cameras: one is the RealSense D435 (IR Stereo) camera, and the RealSense L515 (D-Tof lidar) camera. For both cameras, we gather their intrinsics and calibrate the extrinsics using the method as mentioned in Section 4.

Results. The zero-shot sim-to-real results are collected in Tab. 3, where we compare the policy performances using our CDMs against the same policy using two state-of-the-art prompt-based depth models, and directly using the raw depth image. We have several interesting observations: 1) The CDM works better under their specific camera type, although in the previous section, the CDMs showed generalization under the depth metric on a static dataset. 2) CDMs work better than previous baselines, even on different camera types, showing the advantage of the training dataset and the structure design. 3) The real-world policy performance

Table 3 Zero-shot sim-real results using CDMs as the plugin in a real-world robot pipeline.

Camera	Depth Model	Kitchen Task				Canteen Task					
		Pick Bowl	Put Bowl into Microwave	Close Microwave	Total	Pick Fork	Place Fork	Pick Plate	Dump Plate	Place Plate	Total
Sim (D435-View)	None	43/50	33/50	32/50	30/50	40/50	28/50	47/50	45/50	33/50	21/50
D435	None	0/30	0/30	0/30	0/30	0/30	0/30	0/30	0/30	0/30	0/30
	PromptDA	11/30	5/30	0/30	0/30	17/30	16/30	7/30	2/30	6/30	1/30
	PriorDA	16/30	8/30	7/30	7/30	**30/30**	**30/30**	1/30	0/30	0/30	0/30
	CDM-D435	**29/30**	**26/30**	**26/30**	**26/30**	**30/30**	**30/30**	**15/30**	**14/30**	**14/30**	**14/30**
	CDM-L515	**29/30**	22/30	16/30	14/30	**30/30**	29/30	0/30	0/30	0/30	0/30
Sim (L515-View)	None	43/50	34/50	37/50	32/50	40/50	26/50	46/50	43/50	31/50	20/50
L515	None	0/30	0/30	0/30	0/30	0/30	0/30	0/30	0/30	0/30	0/30
	PromptDA	3/30	0/30	0/30	0/30	3/30	0/30	3/30	0/30	0/30	0/30
	PriorDA	17/30	3/30	2/30	2/30	10/30	8/30	3/30	3/30	3/30	3/30
	CDM-D435	22/30	11/30	9/30	9/30	13/30	11/30	11/30	10/30	9/30	9/30
	CDM-L515	**25/30**	**18/30**	**18/30**	**18/30**	**24/30**	**24/30**	**22/30**	**22/30**	**22/30**	**22/30**

matches the simulation performance, and some is even higher. The reason may be that the randomized position in the simulation is bigger, and some of them are difficult to complete all tasks.

Total latency. We also compare the total latency of using different depth models as the policy plugins, including the pre-processing, model inference (in a precision of Float32), and post-processing time. The results collected from a single 4090 GPU server are in Tab. 4, showing that without any further engineering optimization and quantization, CDMs provide a slow latency, allowing the policy to be run at >6Hz. Additional optimizations, quantization, and other enhancements could further decrease inference times.

Table 4 Total latency of depth models on a single 4090 GPU with a RealSense D435 providing the prompt depth, including the pre-processing, model inference (Float32), and post-processing time.

Depth Model	Total Latency (s)
PriorDA	0.154±0.005
PromptDA	0.188±0.005
CDMs	**0.151**±**0.002**

6 Conclusion and Future Works

This work introduces camera depth models (CDMs), designed to provide high-quality geometric information by enhancing depth perception for specific depth cameras. By delivering accurate depth predictions, CDMs enable robust robotic manipulation in real-world settings with simulation-like precision, effectively bridging the sim-to-real geometry gap. Integrating CDMs with real-world depth cameras, we successfully transferred depth-only visuomotor policies, trained solely in simulation, to real robots on long-horizon manipulation tasks, achieving high success rates. These results underscore the critical role of accurate geometric information, provided by CDMs, in enabling generalizable and effective robotic manipulation. While this study demonstrates CDMs in a depth-only sim-to-real pipeline, their potential extends far beyond this application. Future work could consider leveraging CDMs to relabel RGB-D data, enhancing policies with robust 3D representations. Moreover, by achieving simulation-level 3D perception in the real world and aligning sim-real geometry gaps, CDMs enable seamless integration of simulation and real-world 3D data. This approach could lead to more efficient data utilization strategies, fostering the development of large-scale robotic foundation models with human-level generalization ability for complex manipulation tasks.

Acknowledgements

We would like to express our gratitude to all other members at ByteDance Seed Robotics team for their support throughout this project. We thank Hang Li for his leadership of this team.

References

[1] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. arXiv preprint arXiv:2410.24164, 2024.

[2] Linghao Chen, Yuzhe Qin, Xiaowei Zhou, and Hao Su. Easyhec: Accurate and automatic hand-eye calibration via differentiable rendering and space exploration. IEEE Robotics and Automation Letters, 8(11):7234–7241, 2023.

[3] Xuxin Cheng, Kexin Shi, Ananye Agarwal, and Deepak Pathak. Extreme parkour with legged robots. arXiv preprint arXiv:2309.14341, 2023.

[4] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. The International Journal of Robotics Research, page 02783649241273668, 2023.

[5] Cheng Chi, Zhenjia Xu, Chuer Pan, Eric Cousineau, Benjamin Burchfiel, Siyuan Feng, Russ Tedrake, and Shuran Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. In Proceedings of Robotics: Science and Systems (RSS), 2024.

[6] Qiyu Dai, Jiyao Zhang, Qiwei Li, Tianhao Wu, Hao Dong, Ziyuan Liu, Ping Tan, and He Wang. Domain randomization-enhanced depth simulation and restoration for perceiving and grasping specular and transparent objects. In European Conference on Computer Vision (ECCV), 2022.

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929, 2020.

[8] Hao-Shu Fang, Chenxi Wang, Hongjie Fang, Minghao Gou, Jirong Liu, Hengxu Yan, Wenhai Liu, Yichen Xie, and Cewu Lu. Anygrasp: Robust and efficient grasp perception in spatial and temporal domains. IEEE Transactions on Robotics, 39(5):3929–3945, 2023.

[9] Zipeng Fu, Tony Z. Zhao, and Chelsea Finn. Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation. In Conference on Robot Learning (CoRL), 2024.

[10] Alessandro Gasparetto and V Zanotto. A new method for smooth trajectory planning of robot manipulators. Mechanism and machine theory, 42(4):455–471, 2007.

[11] Alessandro Gasparetto and Vanni Zanotto. A technique for time-jerk optimal planning of robot trajectories. Robotics and Computer-Integrated Manufacturing, 24(3):415–426, 2008.

[12] Vitor Guizilini, Igor Vasiljevic, Dian Chen, Rares Ambrus, and Adrien Gaidon. Towards zero-shot scale-aware monocular depth estimation. arXiv, 2023. In ICCV, pages 9233–9243.

[13] Xiaoshen Han, Minghuan Liu, Yilun Chen, Junqiu Yu, Xiaoyang Lyu, Yang Tian, Bolun Wang, Weinan Zhang, and Jiangmiao Pang. Re^{3}sim: Generating high-fidelity simulation data via 3d-photorealistic real-to-sim for robotic manipulation. arXiv preprint arXiv:2502.08645, 2025.

[14] Jesse Haviland, Niko Sünderhauf, and Peter Corke. A holistic approach to reactive mobile manipulation. IEEE Robotics and Automation Letters, 7(2):3122–3129, 2022.

[15] Kaiming He, Jian Sun, and Xiaoou Tang. Guided image filtering. IEEE transactions on pattern analysis and machine intelligence, 35(6):1397–1409, 2012.

[16] Tairan He, Chong Zhang, Wenli Xiao, Guanqi He, Changliu Liu, and Guanya Shi. Agile but safe: Learning collision-free high-speed legged locomotion. arXiv preprint arXiv:2401.17583, 2024.

[17] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. Advances in neural information processing systems, 33:6840–6851, 2020.

[18] Mu Hu, Wei Yin, Chi Zhang, Zhipeng Cai, Kaixuan Wang, Xiaozhi Chen, and Chunhua Shen. Metric3d v2: A versatile monocular geometric foundation model for zero-shot metric depth and surface normal estimation. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2024.

[19] Pu Hua, Minghuan Liu, Annabella Macaluso, Yunfeng Lin, Weinan Zhang, Huazhe Xu, and Lirui Wang. Gensim2: Scaling robot data generation with multi-modal and reasoning llms. arXiv preprint arXiv:2410.03645, 2024.

[20] HyunJun Jung, Patrick Ruhkamp, Guangyao Zhai, Nikolas Brasch, Yitong Li, Yannick Verdie, Jifei Song, Yiren Zhou, Anil Armagan, Slobodan Ilic, et al. On the importance of accurate geometry data for dense 3d vision tasks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 780–791, 2023.

[21] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, et al. Openvla: An open-source vision-language-action model. In Conference on Robot Learning, pages 2679–2713. PMLR, 2025.

[22] Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. Rma: Rapid motor adaptation for legged robots. In Robotics: Science and Systems (RSS), 2021.

[23] Hang Lai, Jiahang Cao, Jiafeng Xu, Hongtao Wu, Yunfeng Lin, Tao Kong, Yong Yu, and Weinan Zhang. World model-based perception for visual legged locomotion. arXiv preprint arXiv:2409.16784, 2024.

[24] Xinghang Li, Minghuan Liu, Hanbo Zhang, Cunjun Yu, Jie Xu, Hongtao Wu, Chilam Cheang, Ya Jing, Weinan Zhang, Huaping Liu, et al. Vision-language foundation models as effective robot imitators. In ICLR, 2024.

[25] Xinhai Li, Jialin Li, Ziheng Zhang, Rui Zhang, Fan Jia, Tiancai Wang, Haoqiang Fan, Kuo-Kun Tseng, and Ruiping Wang. Robogsim: A real2sim2real robotic gaussian splatting simulator. arXiv preprint arXiv:2411.11839, 2024.

[26] Yunzhu Li, Shuang Li, Vincent Sitzmann, Pulkit Agrawal, and Antonio Torralba. 3d neural scene representations for visuomotor control. In Conference on Robot Learning, pages 112–123. PMLR, 2022.

[27] Haotong Lin, Sida Peng, Jingxiao Chen, Songyou Peng, Jiaming Sun, Minghuan Liu, Hujun Bao, Jiashi Feng, Xiaowei Zhou, and Bingyi Kang. Prompting depth anything for 4k resolution accurate metric depth estimation. In Proceedings of the Computer Vision and Pattern Recognition Conference, pages 17070–17080, 2025.

[28] Minghuan Liu, Zixuan Chen, Xuxin Cheng, Yandong Ji, Rizhao Qiu, Ruihan Yang, and Xiaolong Wang. Visual whole-body control for legged loco-manipulation. The 8th Conference on Robot Learning, 2024.

[29] Haozhe Lou, Yurong Liu, Yike Pan, Yiran Geng, Jianteng Chen, Wenlong Ma, Chenglong Li, Lin Wang, Hengzhen Feng, Lu Shi, et al. Robo-gs: A physics consistent spatial-temporal model for robotic arm with hybrid representation. arXiv preprint arXiv:2408.14873, 2024.

[30] Ajay Mandlekar, Soroush Nasiriany, Bowen Wen, Iretiayo Akinola, Yashraj Narang, Linxi Fan, Yuke Zhu, and Dieter Fox. Mimicgen: A data generation system for scalable robot learning using human demonstrations. In 7th Annual Conference on Robot Learning, 2023.

[31] NVIDIA. Nvidia isaac sim, 2021. URL https://developer.nvidia.com/isaac-sim.

[32] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. arXiv preprint arXiv:2304.07193, 2023.

[33] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In 2018 IEEE international conference on robotics and automation (ICRA), pages 3803–3810. IEEE, 2018.

[34] Luigi Piccinelli, Yung-Hsu Yang, Christos Sakaridis, Mattia Segu, Siyuan Li, Luc Van Gool, and Fisher Yu. Unidepth: Universal monocular metric depth estimation. arXiv, 2024. In CVPR.

[35] Mohammad Nomaan Qureshi, Sparsh Garg, Francisco Yandun, David Held, George Kantor, and Abhishesh Silwal. Splatsim: Zero-shot sim2real transfer of rgb manipulation policies using gaussian splatting. arXiv preprint arXiv:2409.10161, 2024.

[36] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In Proceedings of the IEEE/CVF international conference on computer vision, pages 12179–12188, 2021.

[37] Mike Roberts, Jason Ramapuram, Anurag Ranjan, Atulit Kumar, Miguel Angel Bautista, Nathan Paczan, Russ Webb, and Joshua M. Susskind. Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding. In International Conference on Computer Vision (ICCV) 2021, 2021.

[38] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In Conference on Robot Learning, pages 785–799. PMLR, 2023.

[39] Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse kai Chan, Yuan Gao, Xuanlin Li, Tongzhou Mu, Nan Xiao, Arnav Gurha, Viswesh Nagaswamy Rajesh, Yong Woo Choi, Yen-Ru Chen, Zhiao Huang, Roberto Calandra, Rui Chen, Shan Luo, and Hao Su. Maniskill3: Gpu parallelized robotics simulation and rendering for generalizable embodied ai. Robotics: Science and Systems, 2025.

[40] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. arXiv preprint arXiv:2405.12213, 2024.

[41] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In 2017 IEEE/RSJ international conference on intelligent robots and systems (IROS), pages 23–30. IEEE, 2017.

[42] Jun Wang, Yuzhe Qin, Kaiming Kuang, Yigit Korkmaz, Akhilan Gurumoorthy, Hao Su, and Xiaolong Wang. Cyberdemo: Augmenting simulated human demonstration for real-world dexterous manipulation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 17952–17963, 2024.

[43] Qiang Wang, Shizhen Zheng, Qingsong Yan, Fei Deng, Kaiyong Zhao, and Xiaowen Chu. Irs: A large naturalistic indoor robotics stereo dataset to train deep models for disparity and surface normal estimation. arXiv preprint arXiv:1912.09678, 2019.

[44] Ruicheng Wang, Sicheng Xu, Cassie Dai, Jianfeng Xiang, Yu Deng, Xin Tong, and Jiaolong Yang. Moge: Unlocking accurate monocular geometry estimation for open-domain images with optimal training supervision, 2024. URL https://arxiv.org/abs/2410.19115.

[45] Ruicheng Wang, Sicheng Xu, Cassie Dai, Jianfeng Xiang, Yu Deng, Xin Tong, and Jiaolong Yang. Moge: Unlocking accurate monocular geometry estimation for open-domain images with optimal training supervision. In Proceedings of the Computer Vision and Pattern Recognition Conference, pages 5261–5271, 2025.

[46] Ruicheng Wang, Sicheng Xu, Yue Dong, Yu Deng, Jianfeng Xiang, Zelong Lv, Guangzhong Sun, Xin Tong, and Jiaolong Yang. Moge-2: Accurate monocular geometry with metric scale and sharp details, 2025. URL https://arxiv.org/abs/2507.02546.

[47] Zehan Wang, Siyu Chen, Lihe Yang, Jialei Wang, Ziang Zhang, Hengshuang Zhao, and Zhou Zhao. Depth anything with any prior. arXiv preprint arXiv:2505.10565, 2025.

[48] Songlin Wei, Haoran Geng, Jiayi Chen, Congyue Deng, Cui Wenbo, Chengyang Zhao, Xiaomeng Fang, Leonidas Guibas, and He Wang. D^{3}roma: Disparity diffusion-based depth sensing for material-agnostic robotic manipulation. In ECCV 2024 Workshop on Wild 3D: 3D Modeling, Reconstruction, and Generation in the Wild, 2024.

[49] Bowen Wen, Matthew Trepte, Joseph Aribido, Jan Kautz, Orazio Gallo, and Stan Birchfield. Foundationstereo: Zero-shot stereo matching. arXiv, 2025.

[50] Hongtao Wu, Ya Jing, Chilam Cheang, Guangzeng Chen, Jiafeng Xu, Xinghang Li, Minghuan Liu, Hang Li, and Tao Kong. Unleashing large-scale video generative pre-training for visual robot manipulation. In ICLR, 2024.

[51] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, Li Yi, Angel X. Chang, Leonidas J. Guibas, and Hao Su. SAPIEN: A simulated part-based interactive environment. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2020.

[52] Yufei Xue, Wentao Dong, Minghuan Liu, Weinan Zhang, and Jiangmiao Pang. A unified and general humanoid whole-body controller for fine-grained locomotion. In Robotics: Science and Systems (RSS), 2025.

[53] Ge Yan, Yueh-Hua Wu, and Xiaolong Wang. Dnact: Diffusion guided multi-task 3d policy learning. arXiv preprint arXiv:2403.04115, 2024.

[54] Lihe Yang, Bingyi Kang, Zilong Huang, Xipark2024depthaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything: Unleashing the power of large-scale unlabeled data. In CVPR, 2024.

[55] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. Advances in Neural Information Processing Systems, 37:21875–21911, 2024.

[56] Wei Yin, Jianming Zhang, Oliver Wang, Simon Niklaus, Long Mai, Simon Chen, and Chunhua Shen. Metric3d: Towards zero-shot metric 3d prediction from a single image. arXiv, 2023. In CVPR, pages 9043–9053.

[57] Yanjie Ze, Ge Yan, Yueh-Hua Wu, Annabella Macaluso, Yuying Ge, Jianglong Ye, Nicklas Hansen, Li Erran Li, and Xiaolong Wang. Gnfactor: Multi-task real robot learning with generalizable neural feature fields. In Conference on robot learning, pages 284–301. PMLR, 2023.

[58] Yanjie Ze, Zixuan Chen, Wenhao Wang, Tianyi Chen, Xialin He, Ying Yuan, Xue Bin Peng, and Jiajun Wu. Generalizable humanoid manipulation with 3d diffusion policies. arXiv preprint arXiv:2410.10803, 2024.

[59] Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations. In Proceedings of Robotics: Science and Systems (RSS), 2024.

[60] Tony Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. Robotics: Science and Systems XIX, 2023.

[61] Haoyu Zhen, Xiaowen Qiu, Peihao Chen, Jincheng Yang, Xin Yan, Yilun Du, Yining Hong, and Chuang Gan. 3d-vla: A 3d vision-language-action generative world model. arXiv preprint arXiv:2403.09631, 2024.

[62] Haoyi Zhu, Yating Wang, Di Huang, Weicai Ye, Wanli Ouyang, and Tong He. Point cloud matters: Rethinking the impact of different observation spaces on robot learning. Advances in Neural Information Processing Systems, 37:77799–77830, 2024.

[63] Ziwen Zhuang, Zipeng Fu, Jianren Wang, Christopher Atkeson, Sören Schwertfeger, Chelsea Finn, and Hang Zhao. Robot parkour learning. In Conference on Robot Learning (CoRL), 2023.

[64] Ziwen Zhuang, Shenzhe Yao, and Hang Zhao. Humanoid parkour learning. In 8th Annual Conference on Robot Learning, 2024. URL https://openreview.net/forum?id=fs7ia3FqUM.

Appendix

A Visualization of Synthesized Noise

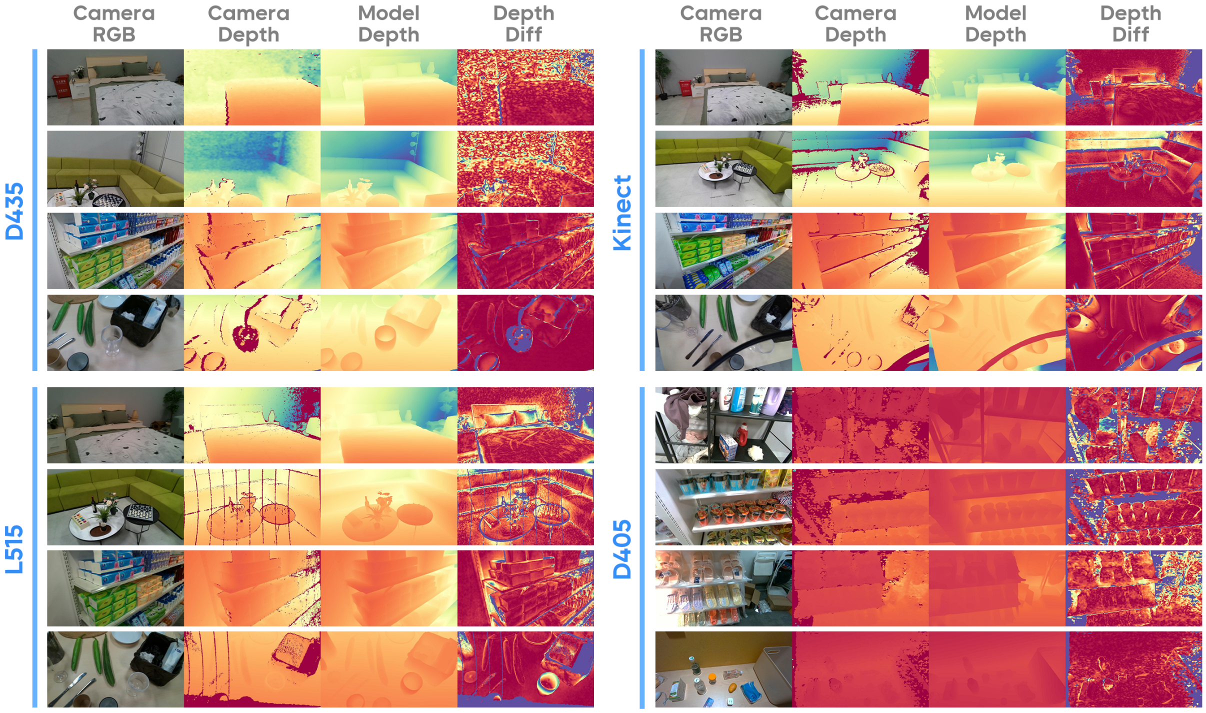


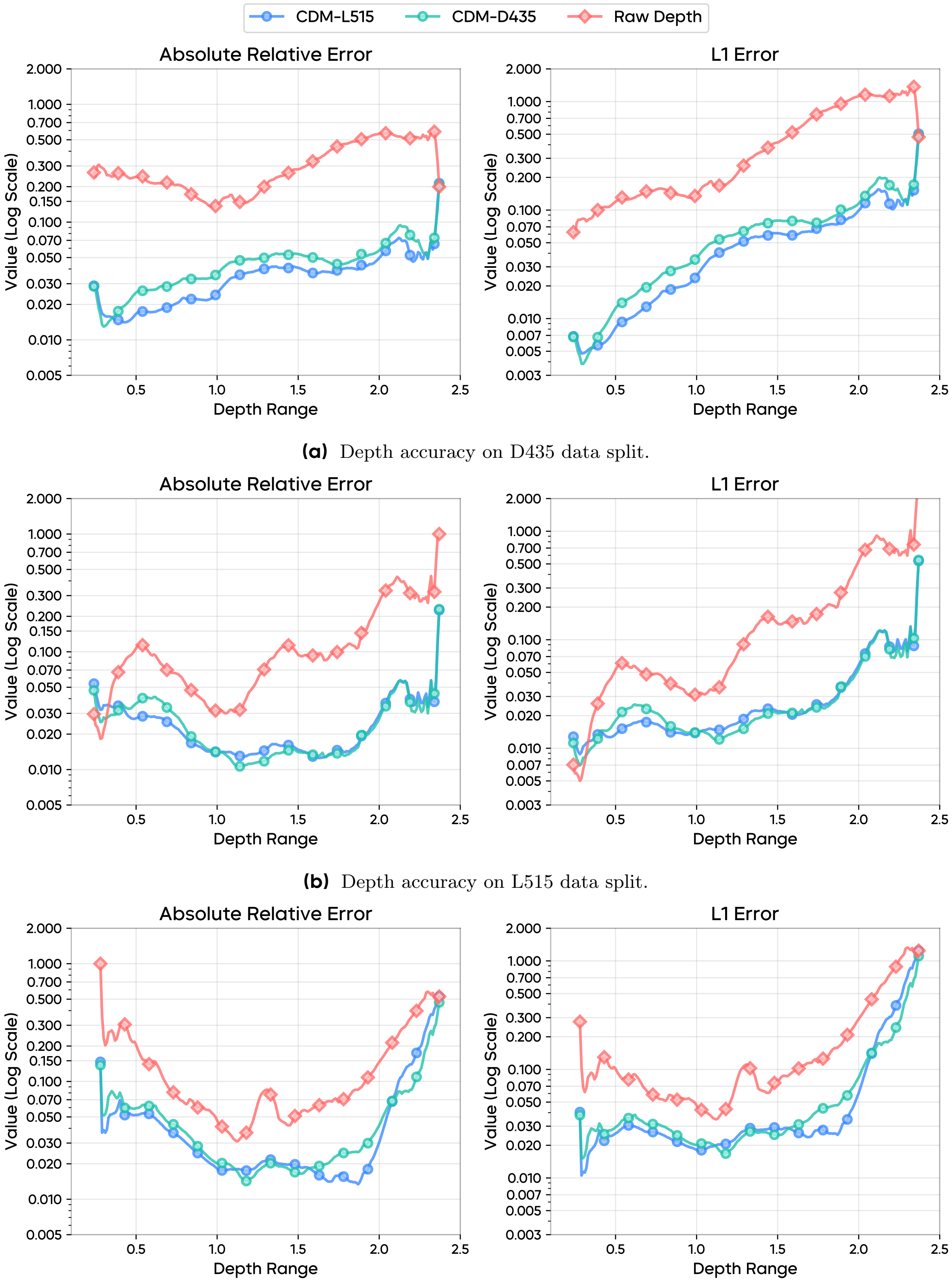
Figure 7 Depth predictions visualization of extended camera depth models, including CDM-D435 on RealSense D415, D435 and D455 captures, along with CDM-D405, CDM-L515, CDM-Kinect, and CDM-Zed2i (Neural) on their corresponding camera captures.

B Extended Camera Depth Models

Except for the two daily-used camera depth models (CDMs) used in the robot manipulation experiments, *i.e.*, RealSense D435 and RealSense L515, we further train three CDMs for RealSense D405, Azure Kinect and Zed2i (Neural Mode). Since we do not have the corresponding real-world dataset with the ground-truth depth label to quantitatively evaluate them, we simply visualize their predictions on representative scenes from the ByteDepth dataset, shown in Fig. 7. Since RealSense D435 uses a similar depth technology to RealSense D415 and RealSense D455, they share a similar noise mode, so we visualize CDM-D435 predictions upon these two cameras and find that CDM-D435 can also provide good predictions. We open all model weights to allow the community to further test and improve them.

C Depth Accuracy w.r.t Distance

Before leaving the factory and being sold to customers, a depth camera will be evaluated at various distances to fully examine its desired working range. To understand the work range of CDMs and as a reference to help people use them, we also provide the depth accuracy w.r.t the distance of CDMs (CDM-D435 and CDM-L515) on the Hammer dataset, in terms of absolute and relative error and the L1 error, shown in Fig. 8. From the accuracy curves, we observe that the raw depth has a larger error than the camera producer claims, for example, the RealSense D435 should have a less than 2% error rate when working under 1∼2 meters, which may be the bias of the dataset. Upon this dataset, CDMs can achieve a high accuracy, whose trend follows the accuracy of the original prompted depth.



(a) Depth accuracy on D435 data split.

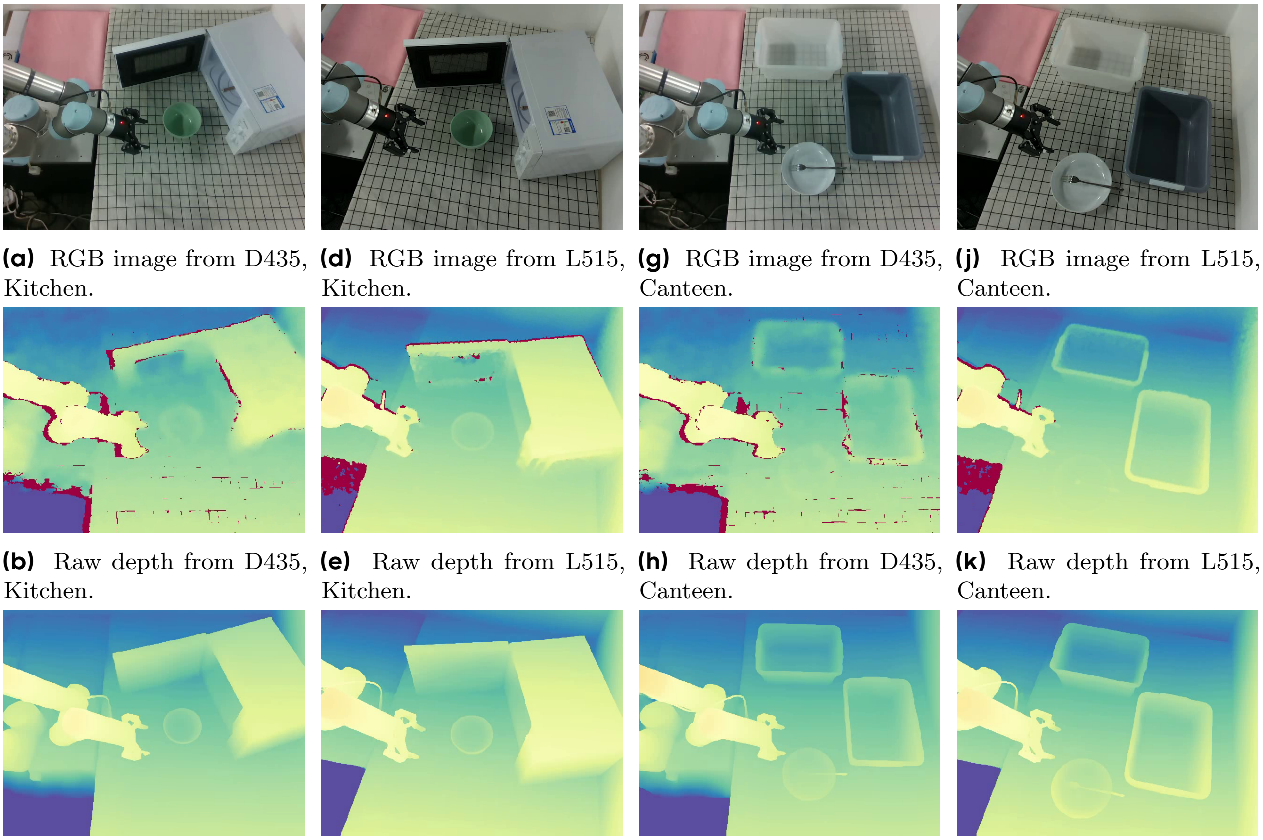
(b) Depth accuracy on L515 data split.

(c) Depth accuracy on Helios data split.

Figure 8 Depth accuracy evaluation of CDMs and other models on the Hammer dataset.

D Depth Comparison

We provide a detailed visual comparison between the raw camera depth and the predicted depth by the proposed CDMs, shown in Fig. 9. We can easily observe that both of these two representative depth cameras have their typical noise and failure modes. For example, both cameras fail to recognize the glass of the microwave and the metal fork; D435 has noisy depth on the plaid tablecloth; L515 has problems with the reflective outside part of the microwave, and the gripper fingers. In comparison, CDMs can provide accurate and complete geometry information.



(a) RGB image from D435, Kitchen.

(d) RGB image from L515, Kitchen.

(g) RGB image from D435, Canteen.

(j) RGB image from L515, Canteen.

(b) Raw depth from D435, Kitchen.

(e) Raw depth from L515, Kitchen.

(h) Raw depth from D435, Canteen.

(k) Raw depth from L515, Canteen.

(c) Depth by CDM-D435, Kitchen.

(f) Depth by CDM-L515, Kitchen.

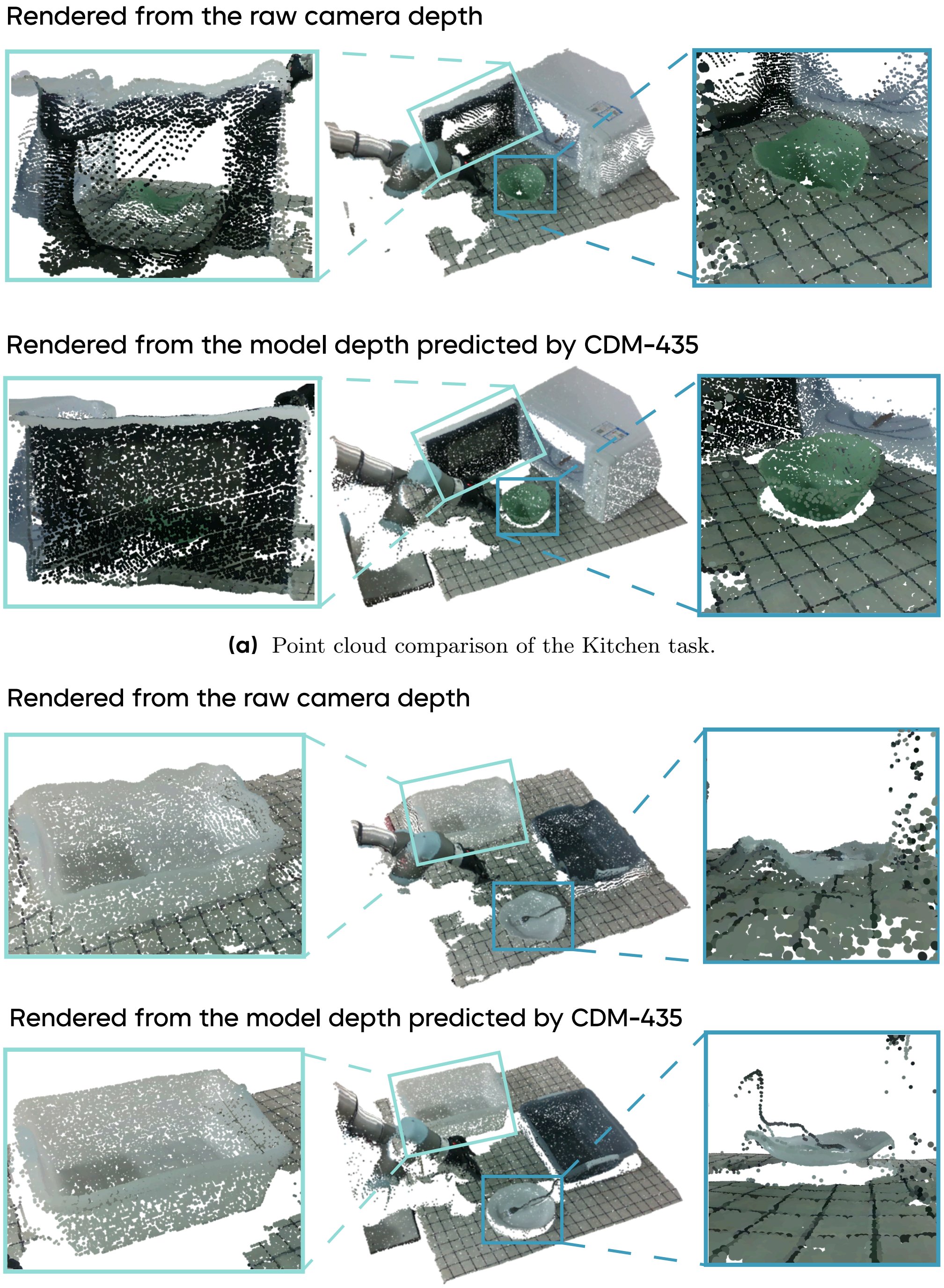
(i) Depth by CDM-D435, Canteen.

(l) Depth by CDM-L515, Canteen.

Figure 9 Detailed real-world cases of two representative depth cameras, RealSense D435 (active IR stereo camera) and RealSense L515 (lidar camera), including color images (first row), camera depth images (second row), and depth predicted by camera depth models proposed in this project (third row).

E Rendered Point Cloud Comparison

We further compared the rendered point clouds transformed from the raw camera depth and the ones predicted by the camera depth models (CDMs), shown in Fig. 10 and Fig. 11. From the rendering results, we can easily observe that the point clouds rendered by the raw depth camera are much noisier, where the objects are distorted and convey wrong geometry information. In comparison, the CDM provides a clean point cloud where the objects maintain most of their original geometry. It is worth noting that in the Canteen task, the geometry of the fork from the raw camera depth is integrated within the plate; the one predicted by the CDM is better, but still inaccurate. This is because the raw camera depth does not provide any useful information about the fork, and the model has to predict the whole from the semantic information of the color image, which may be confusing.



Rendered from the raw camera depth

Rendered from the model depth predicted by CDM-435

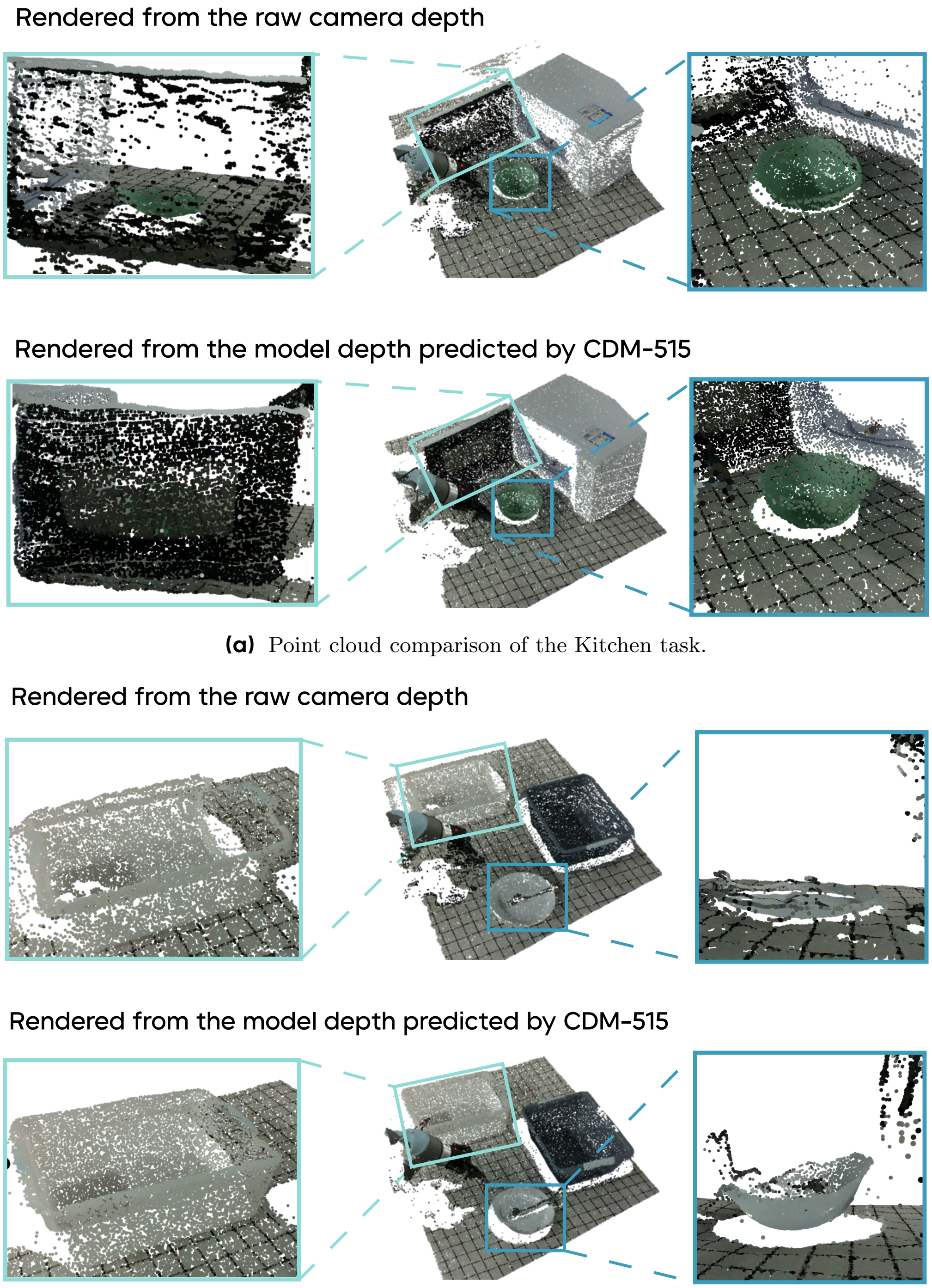
(a) Point cloud comparison of the Kitchen task.

Rendered from the raw camera depth

Rendered from the model depth predicted by CDM-435

(b) Point cloud comparison of the Canteen task.

Figure 10 **Rendered point cloud comparison** between the raw camera depth and the predicted depth of CDM-D435, upon the RealSense D435 camera.



Rendered from the raw camera depth

Rendered from the model depth predicted by CDM-515

(a) Point cloud comparison of the Kitchen task.

Rendered from the raw camera depth

Rendered from the model depth predicted by CDM-515

(b) Point cloud comparison of the Canteen task.

Figure 11 Rendered point cloud comparison between the raw camera depth and the predicted depth of CDM-515, upon the RealSense L515 camera.

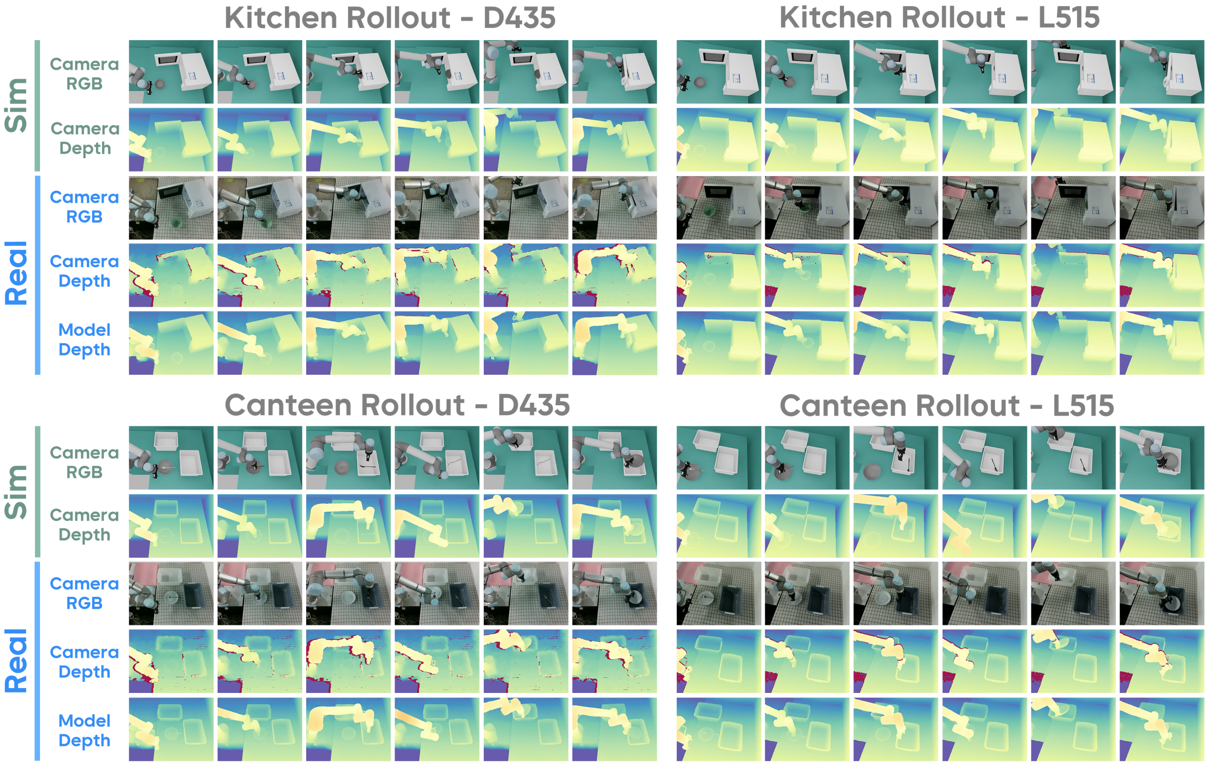


Figure 12 Sim-Real rollouts comparison on two tasks in the sim-to-real experiments, including two views (D435 view and L515 view). For the simulation, we show the rendered RGB and depth images; as for the real experiments, we visualize the RGB images and the depth images from the depth camera, with the predicted depth from the corresponding camera depth model.

F Comparison of Sim-Real Rollouts

We visualize the policy rollouts on two tasks in the sim-to-real experiments in Fig. 12, where we compare the key frames from the simulation and the real world separately. It is readily apparent that the camera depth model provides high-quality, simulation-like depth, offering accurate geometry information in the real world and thus bridging the geometry gap between simulation and reality.

G Data Generation with WBCMimicGen

G.1 Algorithm

To generate demonstrations efficiently in simulation, inspired by Haviland et al. [14], we propose WBCMimicGen, which extends with whole-body control (WBC). Compared to the original MimicGen algorithm, which utilizes inverse kinematics (IK) to solve given target end-effector poses, WBCMimicGen optimizes target joint velocities with WBC, thereby generating smoother, high-quality demonstrations. This approach can be further extended to wheeled robots for mobile manipulation tasks. Utilizing classical control methods, such as WBC, in the simulation preserves the advantage of precise perception, without considering any error, as we can easily acquire privileged information about objects, scenes, and robots.

Specifically, we regard the data generation problem as solving a trajectory of whole-body joint velocities that enables the end-effector to move with a specific velocity. Formally, denote joint velocities as $\boldsymbol{x}$, this problem can be modeled as a quadratic programming (QP) problem [14]:

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & f_o(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top Q\boldsymbol{x} + \mathcal{C}^\top \boldsymbol{x} \\ \text{subject to } & \mathcal{J}\boldsymbol{x} = {}^b\nu_e \ , \\ & \mathcal{A}\boldsymbol{x} \leq \mathcal{B} \ , \\ & \mathcal{X}^- \leq \boldsymbol{x} \leq \mathcal{X}^+ \ , \end{aligned} \tag{11}$$

where $x = (a_{\text{base}}, \dot{q}_{\text{active}}, \delta_1, \delta_2, \cdots, \delta_i)$ and $\mathcal{X}^{+,-}$ is the limits; a_{base} is the velocities of the robot base; $\dot{q}_{\text{active}}$ is the velocity of the joints related to the end-effectors in the QP (so called the active joints); δ_i are slack variables that can help construct a solvable QP. Without loss of generality, suppose there are k end-effectors and n joints, these variables can be expressed as:

$$\begin{aligned} \mathcal{Q} &= \operatorname{diag}(\lambda_q, \lambda_{\delta_1}, \cdots, \lambda_{\delta_k}) \in \mathbb{R}^{(n+6k)} \ , \\ \mathcal{C} &= \begin{pmatrix} \hat{\mathbf{J}}_m + \epsilon \\ 0_{6k\times 1} \end{pmatrix} \in \mathbb{R}^{(n+6k)} \ , \\ \mathcal{A} &= (\mathbf{1}_{n\times(n+6k)}) \in \mathbb{R}^{n\times(n+6k)} \ , \\ \mathcal{B} &= \begin{pmatrix} 0_b \\ \eta\frac{\rho_0-\rho_s}{\rho_i-\rho_s} \\ \vdots \\ \eta\frac{\rho_n-\rho_s}{\rho_i-\rho_s} \end{pmatrix} \in \mathbb{R}^n \ . \end{aligned} \tag{12}$$

Here $\hat{\mathbf{J}}_m$ is the manipulability Jacobian, ϵ is the base to end-effector angle, and ρ is the distance to the nearest joint limit, encouraging the joint not to stay too close to the limit.

G.2 Comparison Results

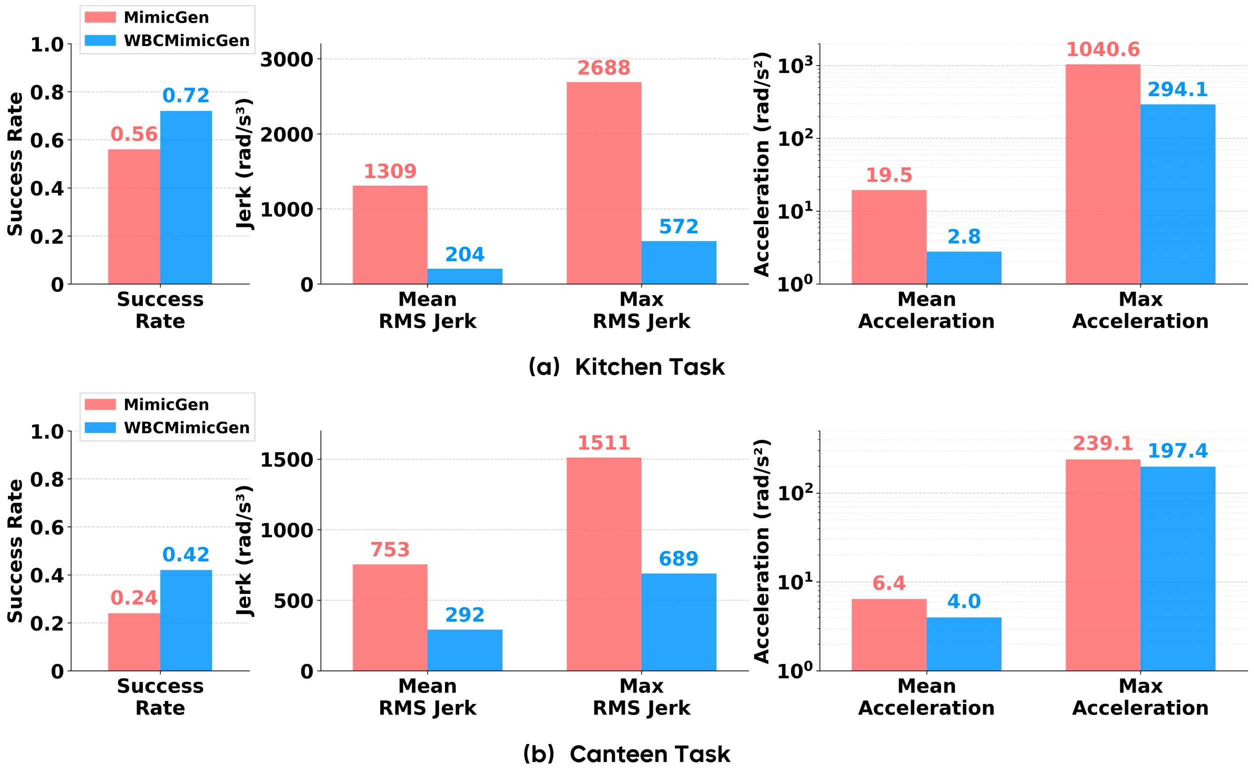
To evaluate the data quality generated by WBCMimicGen, we compare the trajectory smoothness, measured by the mean absolute acceleration and the root mean square (RMS) jerk (*i.e.*, the averaged rate of acceleration change) metrics against the data generated by the original MimicGen [30] algorithm. Previous works Gasparetto and Zanotto [10, 11] use metrics like these as objectives for better smoothness. As shown in Tab. 5, WBCMimicGen consistently generates smoother trajectories with significantly lower acceleration and jerk values across all joints. This improvement stems from the quadratic programming formulation of WBC, which incorporates velocity regularization and enforces joint velocity limits. We encourage readers to further visit the project page for a direct visual comparison of the generated trajectories.

Simulation experiments further validate these improvements. As detailed in Tab. **??**, models trained on WBCMimicGen data achieve higher success rates (72% vs 56% for Kitchen, 42% vs 24% for Canteen) while maintaining substantially lower RMS jerk and acceleration. The baseline approach exhibits much larger

Table 5 Comparison of the generated demonstrations over every robot joint (UR5).

Task	Method	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6
		Mean absolute acceleration (rad/s^2) ↓					
Kitchen	MimicGen	1.284	1.253	0.910	8.115	4.457	6.702
	WBCMimicGen	**0.183**	**0.495**	**0.311**	**4.925**	**2.810**	**6.539**
Canteen	MimicGen	1.180	1.792	1.310	6.487	2.961	1.177
	WBCMimicGen	**0.028**	**0.064**	**0.080**	**2.200**	**0.433**	**0.709**
		RMS jerk (rad/s^3) ↓					
Kitchen	MimicGen	461.374	313.714	277.185	1047.850	852.199	1252.070
	WBCMimicGen	**58.774**	**112.983**	**84.063**	**767.230**	**516.074**	**1020.545**
Canteen	MimicGen	435.487	384.092	425.695	1004.206	592.221	282.824
	WBCMimicGen	**8.610**	**19.879**	**25.812**	**547.281**	**111.007**	**207.548**

acceleration at action chunk boundaries, which increases the likelihood of dropping objects. In contrast, WBCMimicGen's smoother trajectories enhance both task reliability and safety, making them more suitable for real-world deployment.



(a) Kitchen Task

(b) Canteen Task

Figure 13 Performance comparison between policy models trained on demonstrations generated by MimicGen and our WBCMimicGen, on (a) Kitchen Task and (b) Canteen Task. The policy learned from WBCMimicGen reflects a higher success rate while keeping smooth on the rollout policy trajectory (lower RMS jerk and acceleration indicate).

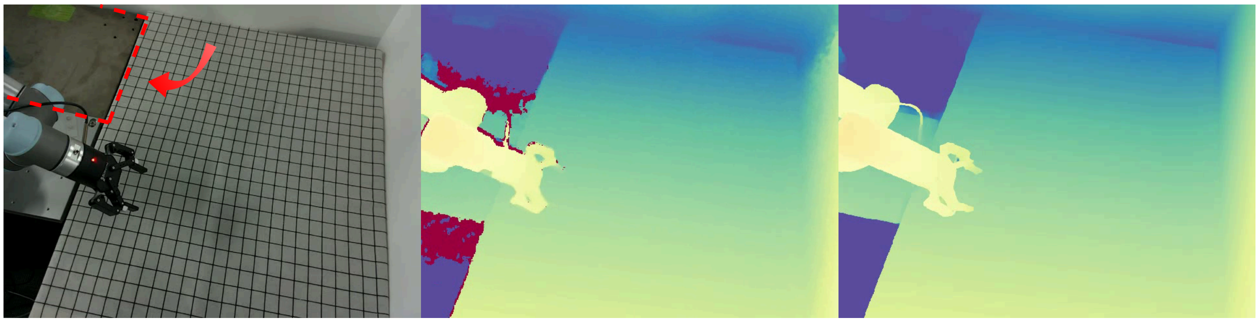


Figure 14 A typical failure case of CDM-515, caused by the wrong prompt and the less informative semantic information contained in the RGB image.

H Limitations and Failure Cases

Although CDMs can fix many errors of the source depth cameras due to the semantic information in the RGB image, they may still fall into some failure cases when the monocular semantic information is not enough to fix the error. In other words, when the prompted camera depth image has wrong metrics for a large region, the predicted depth can be misled. Here we provide a typical case on CDM-515 in Fig. 14, where the red dashed line highlights the area where the prompted camera depth hints that it is a hole. This is due to the metal plane causes the failure perception for the RealSense L515 is a LiDAR depth camera. Additionally, the RGB image does not bring informative semantic information for the CDM to fix that error.