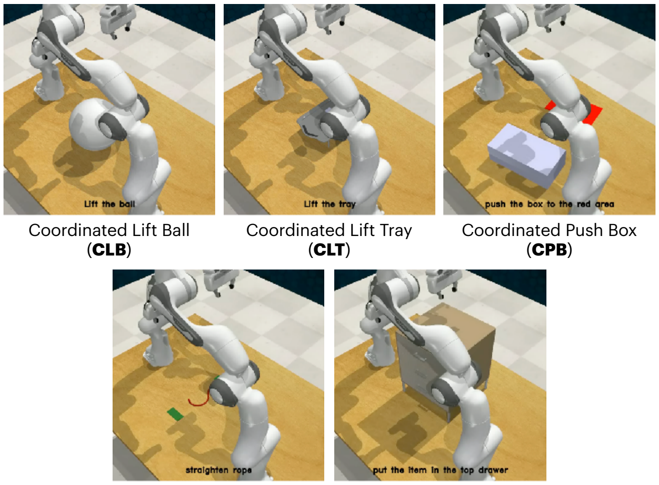
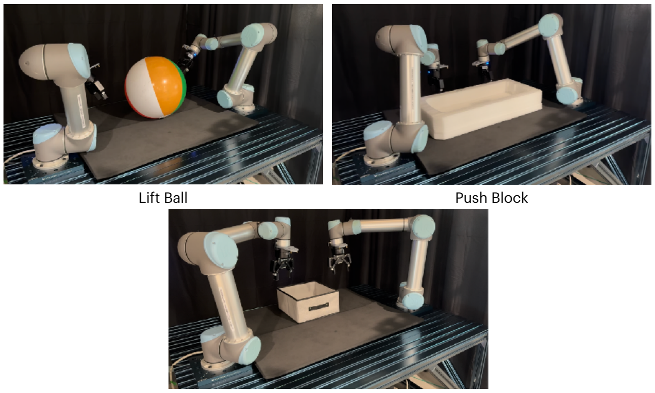
Appendix

A. Task Details



Coordinated Lift Ball (**CLB**)
Coordinated Lift Tray (**CLT**)
Coordinated Push Box (**CPB**)
Bimanual Straighten Rope (**BSR**)
Coordinated Put Item In Drawer (**CPID**)

Fig. 8: **Simulation environments.** Simulation environment for our bimanual manipulation tasks, adapted from PerAct2. Each simulation image is shown above its corresponding language goal. Text overlays within images indicate the language goal of the task. Abbreviations in parentheses correspond to task names used throughout the paper.



Lift Ball
Push Block
Lift Drawer

Fig. 9: **Real-world environments.** Real-world environment for our bimanual manipulation tasks. Each simulation image is shown above.

We show the simulation environment in Figure 8 and the real-world environment in Figure 9. For real-world experiments, we use an Intel RealSense D415 camera to capture RGB and RGB-D images at a resolution of 640 × 480 pixels. These images are first zero-padded and then rescaled to 128 × 128. We use the python-urx library to control the robot arms and I/O programming to operate the Robotiq 2F-85 grippers.

B. Hyperparameters

Table VI summarizes the ACT hyperparameters. We use the default PerAct2 chunk size of 10 for all simulation tasks and a chunk size of 2 for all real-world tasks. In both simulation and real-world, the RGB and RGB-D images are 128 × 128. An NVIDIA 4090 GPU is used to train the ACT policy.

Hyperparameter	Value
Learning Rate	1e-5
Batch Size	16
# Encoder Layers	4
# Decoder Layers	7
Feedforward Dimension	3200
Hidden Dimension	512
# Heads	8
Beta	100
Dropout	0.1

TABLE VI: Hyperparameters of ACT.

Hyperparameter	Value
Base Model	Stable Diffusion 2.1
Learning Rate	1e-5
Weight Decay	1e-2
Epochs	150
Batch Size	24
Image Size	512×512

TABLE VII: Hyperparameters of ControlNet.

Table VII summarizes the ControlNet hyperparameters. ControlNet was trained on a single NVIDIA A100 GPU with 80GB of VRAM. Image synthesis was done on a single NVIDIA P100 GPU.

C. Examples of Synthesized Images

Figure 10 shows the simulation synthesized image results for `Coordinated Put Item In Drawer (CPID)`, `Bimanual Straighten Rope (BSR)`, `Coordinated Lift Tray (CLT)`, and the `Coordinated Push Box (CPB)` tasks. Figure 11 shows the real-world synthesized image results for `Push Box` and `Lift Ball` tasks.

D. Additional Baseline Implementation Details

For the fine-tuned VISTA approach, we randomly sample 10 overhead camera viewpoints from a quarter-circle arc distribution to train ZeroNVS using VISTA's default fine-tuning hyperparameters. The ZeroNVS model is fine-tuned for 5,000 iterations on four NVIDIA A40 GPUs. The resulting fine-tuned model is used to synthesize overhead camera views for all timesteps across each demonstration episode. These synthetically generated overhead images serve as replacements for the original overhead camera data and are utilized to train the ACT policy.

E. Camera Perturbation Sampling

For contact-based states, we utilize the constraint optimization framework from D-CODA [15] to ensure consistent perturbations across both robotic arms. The approach leverages Dual Annealing [64], a global optimization method that handles constrained problems with early termination capabilities. The optimization variable consists of translation coordinates c_{trans}, which define the transformation applied to camera perturbations (normalized within $[-1,1]$).

The objective function incorporates penalties for several undesirable conditions: perturbations that are too small, end-effector configurations positioned too near the table surface, and end-effector poses too close with the other. To validate the kinematic feasibility of perturbed end-effector positions, we integrate a Levenberg-Marquardt (LM) inverse kinematics solver. Configurations that fail to produce valid joint solutions receive appropriate penalty weights in the optimization process. We define the overall optimization problem as:

$$\begin{aligned} \underset{c_{\text{trans}}}{\text{minimize}} \quad & \text{Cost}(c_{\text{trans}}) \\ \text{subject to} \quad & c_{\text{trans}} \in [-1,1]^3,\ c_{\text{trans}} \geq m_{\text{lb}}, \\ & \text{ProximityToTable}(c_{\text{trans}}) \geq d_{\text{table}}, \\ & \text{ProximityToOtherEEF}(c_{\text{trans}}) \geq d_{\text{eff}}, \\ & \text{IKSolver}(c_{\text{trans}}) = \text{valid}. \end{aligned} \tag{3}$$

We configure the perturbation parameters as follows: translation magnitudes are bounded by $[m_{\text{lb}}, m_{\text{ub}}] = [0.05, 0.1]$ meters for both contactless and contact-rich scenarios. Rotational perturbations for contactless states are constrained within $[r_{\text{lb}}, r_{\text{ub}}] = [-28.7°, 28.7°]$. The replacement interval parameter k, which determines the frequency at which original states are replaced with the synthesized states is set to $k = 8$ across all simulated and real-world experimental tasks.

F. Multi-Conditioning for ControlNet

To generate depth images consistent with both the RGB image and target pose image (Figure 4), we modified ControlNet to support multi-conditioning modalities. While the native ControlNet code only supports a single conditioning modality, recent works [65]–[69] have explored incorporating multiple conditioning inputs. However, at the time of publication, these approaches either lack publicly available code, have not been evaluated on skeleton pose conditioning (focusing instead on other modalities such as segmentation masks), or require significantly greater computational resources than the standard ControlNet implementation. Usually, these works require 8 GPU's with each at least 48 GB to train. We anticipate that future work can leverage these novel multi-modal conditioning works to further reduce image artifacts and improve generation quality.

G. Saftey Considerations

Internet pre-trained diffusion models such as Stable Diffusion [54] exhibit harmful biases [70] which could inadvertently influence robot behavior when fine-tuned for manipulation tasks. Robotic systems trained on synthetic data generated by these models should have extensive and thorough safety evaluations before deployment. To mitigate these risks, we recommend implementing safety guidance mechanisms during inference. These include classifiers to detect inappropriate generations and human oversight of synthesized training data.

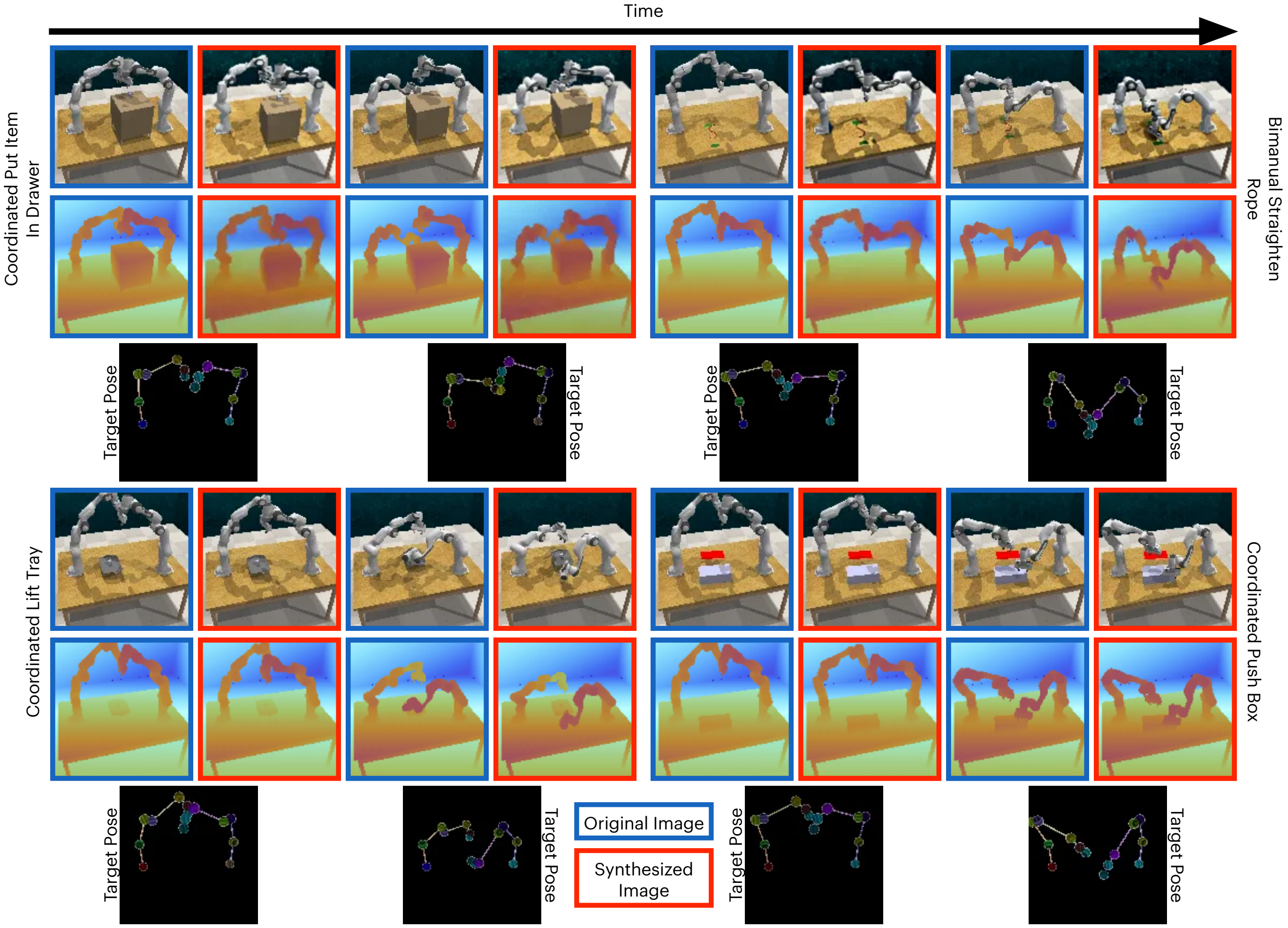


Fig. 10: **Synthesized images in simulation.** We present synthesized images from the `Coordinated Put Item In Drawer (CPID)`, `Bimanual Straighten Rope (BSR)`, `Coordinated Lift Tray (CLT)`, and the `Coordinated Push Box (CPB)` task across two timesteps. The blue bordered images show the original RGB and RGB-D images, while the red bordered images represent the generated target RGB and RGB-D images conditioned on the corresponding skeleton pose shown below.

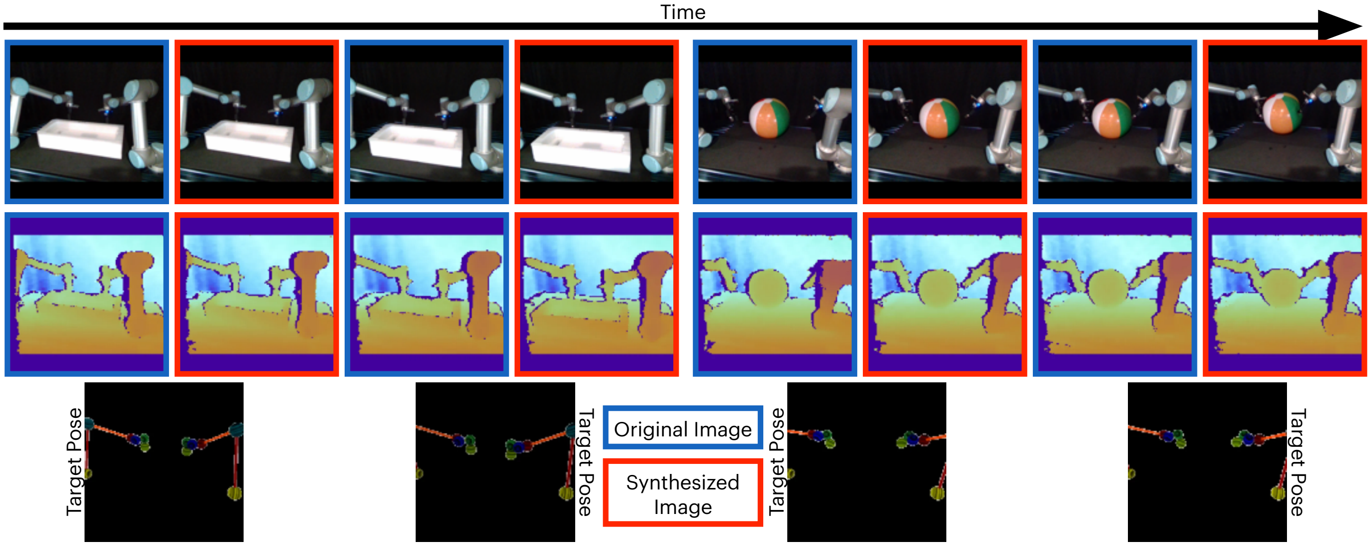


Fig. 11: **Synthesized images in the real-world.** We present synthesized images from the `Push Box` and `Lift Ball` task across two timesteps. The blue bordered images show the original RGB and RGB-D images, while the red bordered images represent the generated target RGB and RGB-D images conditioned on the corresponding skeleton pose shown below.