Self-Supervised 3D Representations for 3D Diffusion Policy

Anderson Compalas
Electrical and Computer Engineering
UC San Diego
acompalas@ucsd.edu

Qiwen Xu
Electrical and Computer Engineering
UC San Diego
qix007@ucsd.edu

Rahat Bhatia
Electrical and Computer Engineering
UC San Diego
rabhatia@ucsd.edu

Abstract

3D Diffusion Policy (DP3) achieves strong visuomotor manipulation performance over 2D Diffusion Policy methods using a deliberately simple point cloud encoder, a per-point MLP with global max pooling trained end-to-end from expert demonstrations. While richer 3D encoders such as PointNet++ and PointNeXt have been shown to hurt DP3 performance in the low-data regime, the question of whether *self-supervised* 3D representations can provide a more transferable geometric prior remains unexplored. Motivated by the success of DINOv2-based diffusion policies in 2D, we present a proof-of-concept study integrating Point-JEPA, a joint-embedding predictive architecture pretrained on ShapeNet-55, as a drop-in encoder replacement for DP3. We evaluate on three Adroit dexterous manipulation tasks with 10 expert demonstrations each, comparing frozen Point-JEPA encoders at three projection dimensionalities (64, 128, 256) against the DP3 baseline. Results are mixed: JEPA-64 outperforms the baseline on door (0.70 vs. 0.65) and JEPA-128 nearly matches on hammer (0.80 vs. 0.95), but all JEPA variants substantially underperform on pen ($\leq$0.40 vs. 0.80), a task requiring fine-grained local contact reasoning. Training dynamics reveal that diffusion loss alone is an unreliable proxy for policy quality, and that the benefit of SSL pretraining is highly sensitive to task-level geometric complexity. These findings suggest that 3D SSL encoders can transfer meaningfully to visuomotor control in some settings, and that future work on domain-specific pretraining or local-aware SSL methods may unlock further gains.

1 Introduction

Robotic manipulation requires generating feasible and diverse visuomotor behaviors from partial observations of a scene. In learning-based manipulation, the challenge is to model a *multi-modal* distribution over action sequences that can successfully complete a task despite uncertainty from partial observability and sensor noise.

Diffusion-based policies have emerged as a powerful approach for capturing multi-modal action distributions through conditional denoising diffusion [5, 3]. Building on this, 3D Diffusion Policy (DP3) [15] demonstrates that conditioning on 3D point cloud representations leads to substantial improvements, achieving a 24.2% relative improvement over 2D policies across 72 simulation tasks. A key design choice of DP3 is its deliberately simple encoder: a 3-layer MLP applied point-wise with global max pooling, trained end-to-end with the policy. While effective, this architecture

cannot model inter-point relationships or long-range geometric structure. PointNet's global max pooling discards all inter-point relationship information by design (it is essentially a bag-of-points encoder) [8]. This raises the question of whether a richer geometric representation could improve policy performance. Notably, the DP3 paper's own ablations show that replacing this encoder with more complex pretrained alternatives, including PointNet++ and PointNeXt with pretrained weights, consistently *hurts* performance [9, 10, 15], suggesting that task-supervised end-to-end training is critical in the low-data regime.

In parallel, recent work has shown that large-scale self-supervised pretraining can improve 2D visuomotor policies. DINOv3-Diffusion Policy [4] demonstrates that replacing task-supervised image encoders with pretrained vision transformers can match or outperform supervised baselines. This motivates a natural question: if richer pretrained 2D representations benefit 2D diffusion policies, can richer pretrained 3D representations similarly benefit DP3?

This work is a proof-of-concept investigation of that question. We replace DP3's MLP encoder with Point-JEPA [11], a joint-embedding predictive architecture. Point-JEPA predicts abstract latent representations of masked spatial regions rather than reconstructing raw point positions, encouraging the encoder to capture global geometric structure. We use the DP3 codebase verbatim except for the encoder swap, ensuring any performance differences are attributable to the representation.

Our main contributions are:

1. A minimal integration of a pretrained 3D SSL encoder into DP3, requiring only a new extractor module and config, a clean testbed for evaluating 3D SSL transfer to visuomotor control.
2. An empirical evaluation across three tasks and three projection dimensionalities, providing a systematic study of 3D SSL pretraining for diffusion-based visuomotor control.
3. Findings that JEPA-64 outperforms the baseline on door (+5%) and JEPA-128 nearly matches on hammer (-15%), alongside task-dependent training dynamics that provide insight into when 3D SSL transfer helps or hurts.

2 Related Work

2.1 Diffusion Policy and 3D Diffusion Policy

Diffusion Policy [3] reformulates behavioral cloning as a conditional denoising diffusion process [5], enabling multi-modal action distributions. DP3 [15] extends this to 3D point cloud observations using a compact MLP encoder, achieving strong generalization across viewpoint, appearance, and object instance. Importantly, DP3's ablations show that pretrained point cloud encoders (PointNet++, PointNeXt) underperform the task-supervised DP3 encoder even with pretraining, a finding that directly motivates investigating whether a different class of 3D SSL (predictive rather than supervised) can close this gap.

2.2 Self-Supervised Visual Learning for Manipulation

DINOv3-Diffusion Policy [4] evaluates pretrained vision transformers as drop-in replacements in image-based diffusion policies, showing up to 10% absolute improvement. DINO-style methods [2, 6] rely on multi-crop augmentation and patch tokenization tied to pixel-space structure, without a direct analog in unordered 3D point sets, motivating purpose-built 3D SSL approaches.

2.3 Self-Supervised Learning for 3D Point Clouds

Contrastive methods such as PointContrast [13] pull together features from different views of the same 3D scene. Reconstruction-based methods such as Point-MAE [7] predict missing point positions, emphasizing low-level geometric fidelity. Point-JEPA [11] instead predicts abstract features of masked regions in representation space, encouraging semantic structure capture over surface detail, a distinction relevant to downstream control tasks where high-level geometry matters more than exact point positions. Ze et al. [14] show SSL-pretrained point cloud representations improve RL-based manipulation sample efficiency, motivating our investigation in the imitation learning and diffusion policy setting.

3 Background

3.1 3D Diffusion Policy Architecture

DP3 [15] consists of two jointly trained components.

DP3 Encoder. The point cloud $P \in \mathbb{R}^{N\times 3}$ (xyz only, no color) is processed by a 3-layer MLP applied independently to each point, followed by global max pooling and a linear projection head:

$$f_{\text{DP3}}(P) = \underbrace{\text{Linear}(256 \to 64) \circ \text{LayerNorm}}_{\text{projection}} \circ \underbrace{\text{MaxPool}}_{\text{aggregate}} \circ h(P) \in \mathbb{R}^{64} \tag{1}$$

$$h(P) = \underbrace{\text{MLP}_{3\to 64\to 128\to 256} \circ \text{LayerNorm/ReLU}}_{\text{per-point MLP}}(P) \tag{2}$$

The MLP is applied to each point independently before pooling, so the encoder cannot model inter-point relationships. LayerNorm layers are interleaved to stabilize training. The total encoder has approximately 0.06M parameters.

1D Convolutional Diffusion Backbone. The conditioning vector $c = [f_{\text{DP3}}(P);\ \text{MLP}(q)] \in \mathbb{R}^{128}$ concatenates the 64-dim point cloud feature with a 64-dim projection of the robot proprioceptive state q. With $N_{\text{obs}} = 2$ observation steps, c is used as a global conditioning signal of dimension 256 for a 1D U-Net. Following Ho et al. [5], the forward process corrupts the action sequence with Gaussian noise; the U-Net is trained to denoise using the reverse process:

$$a^{k-1} = \alpha_k \left(a^k - \gamma_k \epsilon_\theta(a^k, k, c)\right) + \sigma_k \mathcal{N}(0, I) \tag{3}$$

DP3 uses *sample prediction* rather than noise prediction for the denoising target, with training objective:

$$\mathcal{L} = \text{MSE}\big(\epsilon^k,\ \epsilon_\theta\big(\sqrt{\bar{\alpha}_k}\, a^0 + \bar{\beta}_k\, \epsilon^k,\ k,\ v,\ q\big)\big) \tag{4}$$

DDIM [12] is used as the noise scheduler with 100 training and 10 inference timesteps. The action prediction horizon is $H = 4$, with $N_{\text{act}} = 3$ actions executed per inference step.

3.2 Point-JEPA Architecture

Point-JEPA [11] adapts I-JEPA [1] to 3D point clouds. Three components are used at inference time: tokenizer, positional encoding, and transformer encoder.

Tokenizer. Given $P \in \mathbb{R}^{N\times 3}$, Farthest Point Sampling selects c group centers. For each center, the k nearest neighbors are grouped, normalized by subtracting center coordinates (separating local structure from position), and processed by a mini-PointNet sub-encoder to produce a token $t_i \in \mathbb{R}^{384}$. During pretraining, $c = 64$ centers and $k = 32$ neighbors are used on 1024-point ShapeNet clouds.

Greedy Sequencer. A key contribution of Point-JEPA is a greedy sequencer that orders tokens by spatial proximity before context/target selection [11]. Starting from the center point with minimum coordinate sum (near the object edge), each step selects the spatially nearest unvisited center. This produces a sequence where adjacent indices are approximately spatially adjacent, enabling efficient spatial masking of contiguous regions analogous to block masking in I-JEPA.

Positional Encoding. A 2-layer MLP maps each group center (x, y, z) to a positional embedding in $\mathbb{R}^{384}$, added at every transformer layer.

Transformer Encoder (ViT-Small). A 12-layer transformer with 6 attention heads and hidden dimension 384 processes the full token sequence, enabling explicit modeling of inter-group relationships via self-attention. Tokens are mean-pooled to a global descriptor $z \in \mathbb{R}^{384}$.

JEPA Pretraining Objective. During pretraining, $M = 4$ spatially contiguous target blocks are masked. A lightweight predictor g_ϕ with positional conditioning predicts the *target encoder's* representations of masked regions:

$$\hat{y}^{(i)} = g_\phi(x,\ \{m_j\}_{j \in B_i}), \quad \mathcal{L}_{\text{SSL}} = \frac{1}{M} \sum_{i=1}^{M} \sum_{j \in B_i} \text{SmoothL1}(\hat{y}_j,\ \text{sg}(y_j)) \tag{5}$$

The target encoder is an EMA of the context encoder. Predictions are made in representation space, not input space—the model learns abstract geometric structure rather than surface-level point positions. The checkpoint was pretrained on ShapeNet-55 (41,952 instances, 55 categories) for 500 epochs.

Global Feature Bias. The Point-JEPA paper explicitly notes an emphasis on global over local features as a limitation [11], evidenced by strong linear probing and few-shot classification but slightly weaker part segmentation. This has direct implications for task-dependent transfer, discussed in Section 6.

3.3 Our Integration

We replace f_{DP3} with the frozen Point-JEPA context encoder plus a learned projection head:

$$f_{\text{JEPA}}(P) = \text{LayerNorm}(W_p \cdot \text{MeanPool}(f_\theta(P))) \in \mathbb{R}^{d_{\text{out}}} \tag{6}$$

The conditioning vector becomes $c = [f_{\text{JEPA}}(P);\ \text{MLP}(q)]$. We use the DP3 codebase verbatim, with three additions: (1) a new `PointJEPAExtractor` module, (2) a new Hydra config `dp3_jepa.yaml`, and (3) Python 3.8 compatibility patches to Point-JEPA (`from __future__ import annotations` and a pure-PyTorch `cdist+topk` kNN fallback replacing unavailable PyTorch3D ops). We reduce the tokenizer to $c = 32$ centers for Adroit's 512-point clouds to maintain similar group density to pretraining.

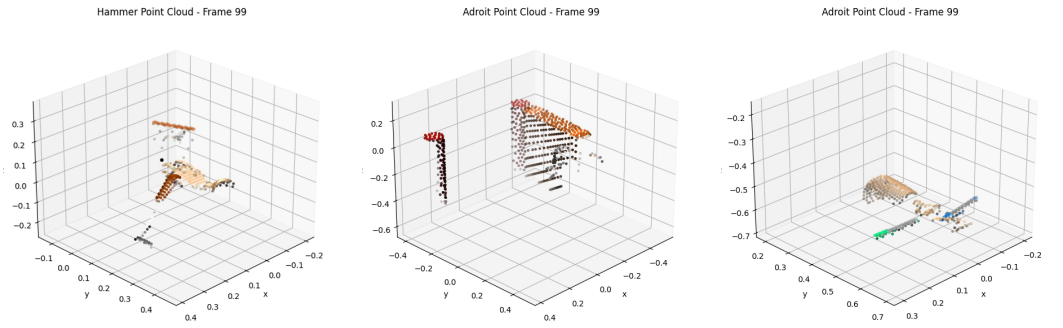
4 Experimental Setup

4.1 Tasks and Data

We evaluate on three Adroit dexterous manipulation tasks (Figure 1): **Hammer** (drive a nail into a board, action dim 26), **Door** (open a latched door, action dim 28), and **Pen** (reposition a pen in-hand, action dim 24). Each task uses 10 expert demonstrations generated by VRL3-trained RL agents [15], yielding approximately 1000 timesteps per task. Observations are 512-point xyz point clouds (no color) from a fixed single-view camera in MuJoCo, and 24-dimensional proprioceptive state. Point clouds are cropped to a workspace bounding box and downsampled via Farthest Point Sampling, following DP3.

4.2 Training Details

All models were set to train for 3000 epochs (although some runs were interrupted before then), batch size 128, AdamW (lr=10^{-4}, wd=10^{-6}), cosine schedule with 500 warmup steps. DDIM with 100 training and 10 inference timesteps. Horizon $H = 4$, $N_{\text{obs}} = 2$, $N_{\text{act}} = 3$. Evaluation runs 20 rollouts every 200 epochs; `test_mean_score` is the fraction of those 20 episodes in which the task is completed successfully (e.g., a score of 0.80 means 16 out of 20 rollouts succeeded). The reported value for each model is the peak `test_mean_score` achieved across all evaluation checkpoints over the full training run, where a checkpoint is saved whenever a new best test score is achieved. Note that the DP3 baseline results reported here are from our own independently trained runs using a single seed, rather than the numbers reported in the original DP3 paper, which averaged the top-5 checkpoints over 3 seeds. Differences in evaluation protocol may account for discrepancies between our baseline scores and those in Ze et al. [15].



(a) Hammer (point cloud) (b) Door (point cloud) (c) Pen (point cloud)

(d) Hammer (RGB) (e) Door (RGB) (f) Pen (RGB)

Figure 1: **Adroit task observations.** Top row: 512-point xyz point clouds as seen by the encoder. Bottom row: corresponding RGB frames. The policy observes only the point cloud and proprioceptive state. The point clouds capture hand and object geometry but lack the rich visual detail of RGB.

4.3 Models Compared

- **DP3 (baseline)**: 3-layer MLP encoder ($3 \to 64 \to 128 \to 256$) + max pool + projection to 64 dims. End-to-end trained. Encoder: 0.06M parameters.
- **JEPA-64**: Frozen Point-JEPA + linear projection to 64 dims. Identical total obs feature dim as baseline ($64 + 64 = 128$), enabling the most direct architectural comparison with all other variables held equal.
- **JEPA-128**: Frozen Point-JEPA + projection to 128 dims. Tests whether a larger bottleneck better preserves the information in Point-JEPA's 384-dim features, since compressing to 64 dims may discard useful geometric structure.
- **JEPA-256**: Frozen Point-JEPA + projection to 256 dims. Tests the upper range of projection size, at the cost of a larger global conditioning vector for the diffusion backbone. All JEPA encoder: 21.9M parameters (frozen).

5 Results

5.1 Main Results

5.2 Hammer Task

The hammer task (Figure 2) shows a non-monotonic relationship between projection head size and performance: JEPA-128 achieves 0.80, JEPA-256 peaks at 0.50 before degrading, and JEPA-64 peaks at 0.45 with the most severe degradation over training.

A consistent pattern across all tasks is that the baseline achieves meaningful task success much earlier in training. On hammer, the baseline shoots up to approximately 0.65 within the first 500 epochs while all JEPA variants remain near zero for hundreds of epochs before slowly climbing. The

Table 1: Peak `test_mean_score` across all evaluation checkpoints over the full 3000-epoch run. A checkpoint is saved whenever a new best test score is achieved; reported values reflect the best checkpoint per run. **Bold** indicates best per task.

Model	d_{out}	Hammer	Pen	Door
DP3 (baseline)	64	**0.950**	**0.800**	0.650
JEPA-64	64	0.450	0.250	**0.700**
JEPA-128	128	0.800	0.375	0.600
JEPA-256	256	0.500	0.400	0.650

Table 2: Architecture summary. JEPA encoder (21.9M) is frozen; only the projection head, state MLP, and diffusion backbone are trained.

Model	Encoder Params	d_{out}	Obs Feat Dim	Global Cond
DP3 (baseline)	0.06M	64	128	256
JEPA-64	21.9M (frozen)	64	128	256
JEPA-128	21.9M (frozen)	128	192	384
JEPA-256	21.9M (frozen)	256	320	640

diffusion loss curves tell a different story: all JEPA variants converge to near-zero denoising loss within 20–30 epochs, compared to hundreds of epochs for the baseline. This divergence between fast diffusion loss convergence and slow task success acquisition suggests that the frozen JEPA encoder makes the diffusion backbone's regression problem easy to fit but does not provide the task-relevant geometric signal needed to generate successful rollouts early in training. The baseline's end-to-end co-adaptation of encoder and backbone together produces a working policy much sooner.

5.3 Pen Task

The pen task (Figure 3) shows all JEPA variants substantially underperforming the baseline at 0.80. Peak scores are 0.25 (JEPA-64), 0.375 (JEPA-128), and 0.40 (JEPA-256), though given the high variance in the curves these differences should not be over-interpreted as a systematic trend.

As with hammer and door, the baseline achieves task success much earlier, jumping to approximately 0.60 within the first 500 epochs while JEPA variants lag behind. The diffusion loss (Figure 3b) shows the opposite: unlike hammer, all four runs converge to nearly identical loss curves throughout training. This contrast, identical diffusion loss but very different task success rates, confirms that diffusion loss is an unreliable indicator of policy quality when encoder alignment is the bottleneck. The JEPA encoder's frozen ShapeNet features are misaligned with the fine-grained local geometry required for in-hand pen reorientation, and while larger projection heads partially compensate, none come close to matching the task-supervised baseline.

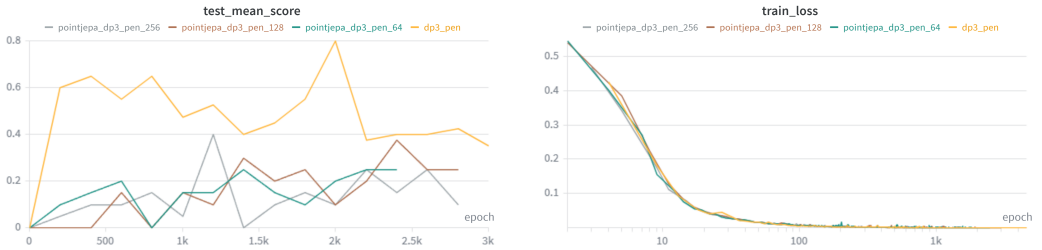
5.4 Door Task

The door task (Figure 4) presents the most interesting results. The baseline again achieves task success earliest, rising to $\approx$0.50 within the first few hundred epochs while all JEPA variants start near zero. However, JEPA-64 is the only model across all tasks and all variants to strictly exceed the baseline's peak performance, reaching 0.70 at epoch 2200 vs. the baseline's 0.65.

JEPA-128 and JEPA-256 reach peaks of approximately 0.60 and 0.65 respectively but with high variance across evaluations, and neither surpasses the baseline. The door diffusion loss (Figure 4b) mirrors the pen pattern rather than hammer, all four runs converge nearly identically, suggesting the frozen encoder provides no particular advantage or disadvantage in terms of denoising difficulty on this task.

(a) Test success rate (b) Diffusion loss (log scale)

Figure 2: **Hammer task.** (a) PointNet baseline converges to 0.95. JEPA-128 achieves the best SSL result (0.80) but with high variance. JEPA-256 peaks at 0.50 and degrades. JEPA-64 peaks at 0.45 with pronounced later degradation. (b) All JEPA variants converge to near-zero diffusion loss faster than the baseline due to the frozen encoder providing structured conditioning from the start, yet this faster convergence does not translate to better task performance.



(a) Test success rate (b) Diffusion loss (log scale)

Figure 3: **Pen task.** (a) Baseline peaks at 0.80 while all JEPA variants substantially underperform. Peak scores are 0.25 (JEPA-64), 0.375 (JEPA-128), and 0.40 (JEPA-256), though all remain well below the baseline. (b) Unlike hammer, all four runs converge to near-identical diffusion loss curves, indicating the denoising loss is an unreliable indicator of policy quality when encoder alignment is the bottleneck.

6 Discussion

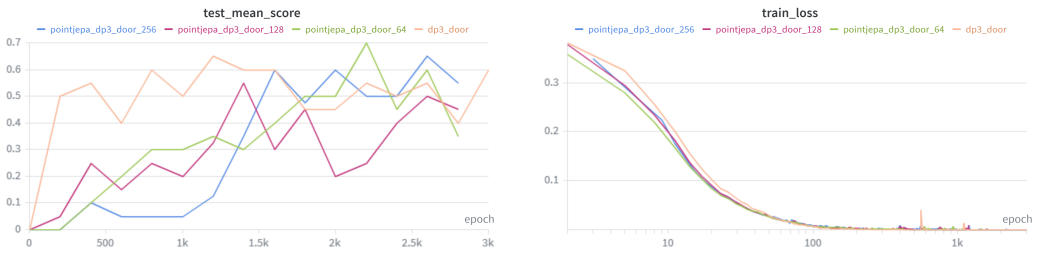
6.1 Proof of Concept: 3D SSL Can Function in DP3

The primary takeaway is that frozen 3D SSL encoders *can* function as drop-in replacements in the DP3 pipeline. The DP3 paper's own Table V shows that pretrained supervised encoders (PointNet++, PointNeXt) achieve near-zero success rates on the same Adroit tasks [15]. Our JEPA-64 achieving 0.45 on hammer and 0.70 on door substantially exceeds those prior negative results, suggesting the JEPA pretraining objective produces more transferable features than supervised 3D encoders. This motivates further investigation into predictive SSL for visuomotor control.

6.2 Convergence Dynamics as a Task-Specific Diagnostic

Across all three tasks, the baseline achieves meaningful task success substantially earlier in training than any JEPA variant, the test success rate curve for the baseline rises steeply within the first few hundred epochs while JEPA variants remain near zero before slowly climbing. This is despite JEPA variants reaching near-zero diffusion loss much faster on hammer, illustrating a consistent decoupling between how quickly the diffusion backbone fits its training objective and how quickly the policy learns to succeed at the task. The frozen JEPA encoder makes the denoising regression easy to minimize but does not provide the task-relevant geometric signal needed to generate successful rollouts early in training.

The diffusion loss pattern itself varies by task. On hammer, JEPA variants converge faster than the baseline, suggesting the frozen features reduce the backbone's learning difficulty on that task



(a) Test success rate

(b) Diffusion loss (log scale)

Figure 4: **Door task.** (a) The baseline achieves task success earliest, climbing to ≈0.50 within the first few hundred epochs. JEPA-64 is the only variant to exceed the baseline's peak (0.70 vs. 0.65), albeit much later in training. JEPA-128 and JEPA-256 peak around 0.60 and 0.65 respectively but with high variance. (b) Unlike hammer, all four runs converge to nearly identical diffusion loss curves, consistent with the pen task pattern.

specifically. On pen and door, all runs converge nearly identically regardless of encoder type, indicating the frozen encoder provides neither advantage nor disadvantage in denoising difficulty. In all cases, diffusion loss convergence speed is a poor predictor of final task performance.

6.3 Global Feature Bias Explains Task-Dependent Transfer

Point-JEPA's own authors note its emphasis on global over local features as a limitation [11]. Our results map directly onto this characteristic. The performance ordering—door (+5% for JEPA-64) > hammer (−15% for JEPA-128) > pen (−50% for JEPA-256)—corresponds precisely to a ranking by global vs. local geometric reasoning:

- **Door**: Locate a handle (global), grasp, rotate. JEPA's global descriptor helps; JEPA-64 exceeds the baseline.
- **Hammer**: Align tool with nail (semi-global), strike. JEPA-128 nearly matches the baseline.
- **Pen**: Reorient in-hand (local). Fine-grained finger-object contact geometry is critical. All JEPA variants substantially underperform (peak scores of 0.25, 0.375, and 0.40 for JEPA-64, 128, and 256 respectively), all well below the 0.80 baseline.

6.4 Caveats and Limitations

Simulated data with RL expert ceiling. All experiments use simulated Adroit environments with expert demonstrations from VRL3-trained RL agents. The quality of demonstrations is bounded by RL policy performance in simulation, and point clouds are idealized raycasts rather than real sensor data.

Adroit only. We evaluate exclusively on three Adroit tasks. DP3 supports DexArt, MetaWorld, Bi-DexHands, and real-world datasets. Our conclusions may not generalize to gripper-based, bimanual, or articulated object tasks where geometric complexity differs substantially.

Frozen encoder. We did not experiment with finetuning the Point-JEPA encoder. Even light finetuning might substantially close the performance gap, particularly for pen where the encoder appears to provide actively misleading features.

Single 3D SSL method. We evaluate only Point-JEPA. Methods with stronger local feature learning (Point-MAE, EdgeConv, Point-M2AE) may perform differently, particularly on fine-grained tasks.

ShapeNet domain gap. Point-JEPA was pretrained on ShapeNet-55 (clean, normalized CAD objects). Adroit point clouds are in robot workspace coordinates, contain the robot hand alongside objects, and are not normalized. Domain-specific pretraining would likely yield substantially better transfer.

6.5 Future Directions

1. **Domain-specific pretraining.** Train Point-JEPA on manipulation-relevant 3D data to reduce the distribution shift.
2. **Light encoder finetuning.** Allow partial adaptation through joint finetuning, particularly for tasks where the frozen encoder is actively misleading (pen).
3. **Broader 3D encoder comparison.** Evaluate Point-MAE, EdgeConv, and Point-M2AE to understand which inductive biases matter for which task types.
4. **Diverse benchmarks.** Evaluate on DexArt, MetaWorld, and real-world data.
5. **OOD generalization.** SSL pretraining may provide larger benefits in out-of-distribution test scenarios where DP3's real-world experiments suggest 3D representations are particularly advantageous [15].

7 Conclusion

We presented a proof-of-concept evaluation of frozen Point-JEPA representations in the DP3 diffusion policy framework, motivated by the success of DINOv2-based policies in 2D. DP3's MLP encoder is deliberately simple, a per-point MLP with global max pooling that cannot model inter-point relationships, and our hypothesis was that richer transformer-based geometric features might improve performance. Compared to prior results showing pretrained supervised 3D encoders failing entirely on Adroit tasks, our JEPA-64 achieving 0.45–0.70 across tasks represents meaningful progress.

Results are mixed but constructive. JEPA-64 outperforms the DP3 baseline on door (+5%) and JEPA-128 nearly matches on hammer (-15%), demonstrating that frozen 3D SSL encoders can function effectively in the DP3 visuomotor pipeline. On door, JEPA-64 is the only model across all tasks and variants to strictly exceed the baseline's peak, while JEPA-128 and JEPA-256 reach comparable but not superior performance. The severe underperformance on pen is explained by Point-JEPA's known global feature bias, which is ill-suited to the fine-grained local contact geometry required for in-hand reorientation. Across all tasks, the baseline achieves task success substantially earlier in training, even when JEPA variants show faster diffusion loss convergence on hammer, highlighting that diffusion loss is an unreliable proxy for policy quality when encoder alignment is the bottleneck.

These findings validate the core premise that richer 3D representations can benefit visuomotor diffusion policies in some settings, and motivate future work on domain-aligned pretraining, light finetuning, and 3D SSL methods with stronger local feature learning. As 3D SSL methods continue to improve and manipulation-relevant pretraining data becomes available, the gap between SSL and task-supervised encoders may narrow further.

References

[1] Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture, 2023. URL https://arxiv.org/abs/2301.08243.

[2] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers, 2021. URL https://arxiv.org/abs/2104.14294.

[3] 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, 2024. URL https://arxiv.org/abs/2303.04137.

[4] ThankGod Egbe, Peng Wang, Zhihao Guo, and Zidong Chen. Dinov3-diffusion policy: Self-supervised large visual model for visuomotor diffusion policy learning, 2025. URL https://arxiv.org/abs/2509.17684.

[5] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020. URL https://arxiv.org/abs/2006.11239.

[6] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2024. URL https://arxiv.org/abs/2304.07193.

[7] Yatian Pang, Wenxiao Wang, Francis E. H. Tay, Wei Liu, Yonghong Tian, and Li Yuan. Masked autoencoders for point cloud self-supervised learning, 2022. URL https://arxiv.org/abs/2203.06604.

[8] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation, 2017. URL https://arxiv.org/abs/1612.00593.

[9] Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space, 2017. URL https://arxiv.org/abs/1706.02413.

[10] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Abed Al Kader Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. Pointnext: Revisiting pointnet++ with improved training and scaling strategies, 2022. URL https://arxiv.org/abs/2206.04670.

[11] Ayumu Saito, Prachi Kudeshia, and Jiju Poovvancheri. Point-jepa: A joint embedding predictive architecture for self-supervised learning on point cloud, 2025. URL https://arxiv.org/abs/2404.16432.

[12] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models, 2022. URL https://arxiv.org/abs/2010.02502.

[13] Saining Xie, Jiatao Gu, Demi Guo, Charles R. Qi, Leonidas J. Guibas, and Or Litany. Pointcontrast: Unsupervised pre-training for 3d point cloud understanding, 2020. URL https://arxiv.org/abs/2007.10985.

[14] Yanjie Ze, Nicklas Hansen, Yinbo Chen, Mohit Jain, and Xiaolong Wang. Visual reinforcement learning with self-supervised 3d representations, 2023. URL https://arxiv.org/abs/2210.07241.

[15] Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations, 2024. URL https://arxiv.org/abs/2403.03954.

Contribution Statement

Anderson Compalas led the majority of the project, including the full Point-JEPA integration into the DP3 codebase, all Python 3.8 compatibility patches, the majority of training runs (all hammer, pen, and door JEPA-64/128 experiments, and all baseline runs), experimental analysis, and the drafting of the final report.

Qiwen Xu ran the JEPA-256 door training experiment and contributed to report writing and editing.

Rahat Bhatia did not contribute to the code implementation or report writing.