POINTDEX: DEXTEROUS GRASP GENERATION VIA POINT-BASED HAND-OBJECT-CONTACT MODELING

Anonymous authors
Paper under double-blind review

ABSTRACT

Dexterous grasp generation requires precise reasoning over the spatial and functional relationships among hands, objects, and their contacts. Yet existing approaches often struggle to model these interdependent entities effectively, and provide only limited means for steering the generated grasps. We introduce PointDex, an effective, steerable framework for dexterous grasp generation that captures hand–object interactions by representing grasp pose, object contact, and hand contact as a unified set of 3D points in a structured space. This explicit, disentangled yet jointly shared representation allows a single transformer to simultaneously denoise all point types, and naturally supports flexible steering by conditioning on any subset of points while denoising the rest. Given a 3D object and a language instruction, PointDex jointly generates posed hand keypoints and paired object–hand contact points, and subsequently recovers wrist pose and joint angles via inverse kinematics. On the DexGYS and Dexonomy benchmarks, PointDex improves language-conditioned grasp success by 16.6 points over the strongest prior method and reduces steering contact distance by up to 80%, while unifying diverse steering modes within a single model.

1 INTRODUCTION

Dexterous grasp generation with multi-fingered robot hands requires reasoning over the spatial and functional relationships among the hand, the object, and the contacts between them. Recent generative approaches model multimodal grasp distributions to produce diverse feasible grasps for a given object (Huang et al., 2023; Weng et al., 2024; Xu et al., 2023; Zhong et al., 2025), and language-conditioned methods further require the grasp to serve an intended use, such as holding a mug by its handle (Wei et al., 2024). Whether such a grasp succeeds depends largely on where and how the hand touches the object, and modeling contact explicitly has improved grasp quality and generalization (Jiang et al., 2021; Li et al., 2023; Liu et al., 2023; Lu et al., 2024).

Existing methods, however, relate these entities only indirectly. Most generative models parameterize a grasp as a wrist pose and joint angles (Weng et al., 2024; Xu et al., 2023; Zhong et al., 2025), while contacts are maps over the object surface or labels over hand parts (Jiang et al., 2021; Liu et al., 2023). The model must therefore bridge quantities that live in different spaces. Pose components have different units, common rotation parameterizations are discontinuous (Zhou et al., 2019), and each fingertip depends nonlinearly on every joint along its chain, so the model must implicitly learn forward kinematics. The same split limits *steering*, where users guide generation toward a desired grasp by specifying contacts. Recent methods condition on object regions, hand parts, or hand–object contact assignments (Lee et al., 2026; Mérand et al., 2026; Wang et al., 2024; Zuo et al., 2026), but each is built around one form of contact with its own conditioning mechanism, so it cannot easily accept another within the same generative model.

Our key observation is that all of these quantities can be expressed as 3D points. A grasp pose is determined by a fixed set of hand keypoints, an object contact is a point on the object surface, and a hand contact is a point on the hand surface. We therefore represent the grasp pose and both contacts as a shared representation of 3D points. This addresses both issues above. Keypoints make the pose explicit, placing the fingers directly, so hand–object proximity and penetration become distances between points, with no rotation parameters or implicit forward kinematics. Every contact

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

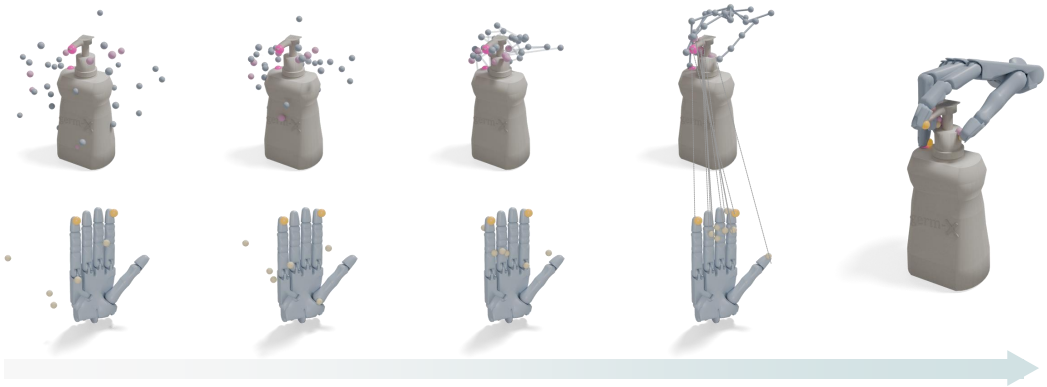


Figure 1: **PointDex generates a grasp and its contacts jointly as 3D points.** From left to right, one model denoises the hand keypoints (gray) and object contacts (pink) around the object (top), together with the hand contacts (yellow) on a canonical open hand (bottom). Saturated contacts are steering inputs held fixed throughout, and the model generates the rest. Dashed lines link each object contact to its paired hand contact. Inverse kinematics then recovers the posed hand from the keypoints (right).

constraint, whatever its type, also becomes a point that the model can either condition on or generate, so no constraint needs its own conditioning mechanism.

We introduce **PointDex**, an effective, steerable framework for dexterous grasp generation built on this observation. Given a 3D object and a language instruction, PointDex jointly generates posed hand keypoints and paired object–hand contact points, denoising all point types with a single transformer trained by flow matching. Inverse kinematics then recovers the wrist pose and joint angles from the generated keypoints. Hand and object contacts are disentangled into separate points with independent noise levels, so the model can condition on any subset of contact points and denoise the rest. At inference, object contacts specify *where to grasp* (e.g., a mug by its handle), hand contacts specify *which hand parts to use* (e.g., a thumb–index pinch), and both together specify *what touches where* (e.g., the index fingertip on a trigger). One model supports all three steering modes without separate modules or retraining.

Our contributions are as follows:

- **Shared point representation of grasps and contacts.** We represent grasp poses, object contacts, and hand contacts all as 3D points and generate them jointly with a single transformer. Replacing keypoints with joint angles, or removing contact points, lowers grasp success.
- **Unified multi-mode steering.** We represent hand and object contacts as separate points with independent noise levels, enabling object-contact, hand-contact, and correspondence-level steering within a single model without retraining.
- **Systematic evaluation.** We evaluate steering fidelity, grasp quality, and partial contact specifications across all three modes. PointDex raises language-conditioned grasp success on DexGYS by 16.6 points over DextER, the strongest prior method, and reduces the steering contact distance by up to 80% across DexGYS and Dexonomy.

2 Related Work

Dexterous Grasp Generation. Dexterous grasp generation must produce diverse feasible poses for multi-fingered hands. UniDexGrasp (Xu et al., 2023) learns diverse grasp proposals conditioned on object point clouds, while DGTR (Xu et al., 2024) approaches diversity through direct set prediction. More recent diffusion-based methods, including DexDiffuser (Weng et al., 2024) and DexGrasp Anything (Zhong et al., 2025), improve grasp diversity, feasibility, and generalization through iterative generation and physics-aware constraints. Despite these advances, most methods primarily generate the final hand configuration, leaving the underlying hand–object interaction implicit and motivating contact-aware formulations.

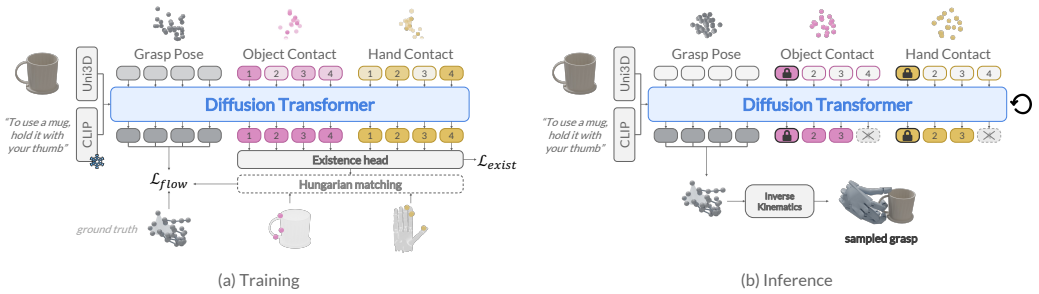


Figure 2: **Overview of PointDex.** The grasp pose (gray), object contacts (pink), and hand contacts (yellow) are all represented as 3D point tokens, and a single Diffusion Transformer denoises them jointly. (a) In training, the model receives noisy tokens and predicts the clean ones, conditioned on the object geometry and language instruction. (b) At inference, the model generates a grasp and its contacts from the object and instruction. If the user specifies any contacts, the model uses them as steering contacts and generates the rest accordingly.

Steerability in Robotic Manipulation. Recent manipulation policies increasingly expose intermediate representations that permit control beyond task-level commands, including dense language annotations (Smith et al., 2025), spatial trajectory sketches (Gu et al., 2023), object motion specifications (Su et al., 2025), and multi-level command interfaces within a single policy (Chen et al., 2026; Hu et al., 2026). These approaches share a common goal of letting users shape policy behavior at a finer granularity than end-goal commands, whether through language, spatial sketches, or hierarchical abstractions. Our work brings this steerability perspective to dexterous grasp generation by exposing hand–object contact as a structured, user-controllable conditioning variable.

Contact-Aware Dexterous Grasp Generation. Contact provides an explicit representation of the physical interaction underlying a grasp. Early methods use contact to improve generation quality without exposing it to the user. GraspTTA (Jiang et al., 2021) and Contact2Grasp (Li et al., 2022) use contact maps as intermediate representations that guide pose generation, while ContactGen (Liu et al., 2023) and UGG (Lu et al., 2024) incorporate richer contact structures, including hand-part labels and joint hand–object–contact modeling. These approaches treat contact as an internal variable predicted during generation rather than a user-controllable input.

More recent methods expose contact as a steering interface, but each supports only one type of specification. RegionGrasp (Wang et al., 2024) conditions on object-surface contacts, MaskDexGrasp (Zuo et al., 2026) conditions on hand-part specifications, and ClickDiff (Li et al., 2024) and DextER (Lee et al., 2026) condition on paired hand–object correspondences. Most closely related, DextER shares our use of contact correspondences as a conditioning variable, but represents each correspondence as a single coupled token, requiring both the hand and object components to be specified or generated together. We instead represent the hand and object components of each correspondence as separate tokens with independent noise levels. This enables three steering modes within a single model, namely object-contact (*where to grasp*), hand-contact (*which hand parts to use*), and correspondence-level (*what touches where*), without task-specific architectures or retraining.

3 POINTDEX

3.1 PROBLEM FORMULATION

Given an object point cloud $\mathbf{P} \in \mathbb{R}^{N\times 3}$ and a language instruction ℓ, we sample a grasp $\mathbf{q} \sim p(\mathbf{q} \mid \mathbf{P}, \ell)$, where $\mathbf{q} = (\mathbf{R}_w, \mathbf{t}_w, \boldsymbol{\theta})$ has wrist pose $(\mathbf{R}_w, \mathbf{t}_w) \in SE(3)$ and finger joint angles $\boldsymbol{\theta} \in \mathbb{R}^J$. The grasp should agree with ℓ and be stable, i.e., hold the object under perturbation in simulation without penetration. We also model the contacts $\mathcal{C} = \{(\mathbf{x}_i^o, \mathbf{x}_i^h)\}_{i=1}^n$, each pairing an object surface point $\mathbf{x}^o$ with the touching point $\mathbf{x}^h$ on the canonical open hand, and sample $(\mathbf{q}, \mathcal{C}) \sim p(\mathbf{q}, \mathcal{C} \mid \mathbf{P}, \ell)$.

3.2 Grasp Pose and Contact Representation

Our core idea is to represent the grasp pose by 3D hand keypoints instead of joint angles, so that the pose and the contacts are all points in 3D Euclidean space.

Grasp pose representation. We represent the grasp pose using 3D hand keypoints in the world frame, each tied to a fixed location on the hand. We use S keypoints $\mathbf{X}^g \in \mathbb{R}^{S\times 3}$, whose j-th row $\mathbf{x}_j^g \in \mathbb{R}^3$ is the position of the j-th keypoint. The keypoints are anatomically ordered, so each index always corresponds to the same hand location. Once the keypoints are generated, inverse kinematics recovers the grasp $\mathbf{q}$ from them (Section 3.4).

Hand-object contact representation. We represent each hand–object contact by a pair of points $(\mathbf{x}^o, \mathbf{x}^h)$, where $\mathbf{x}^o \in \mathbb{R}^3$ is the contact point on the object surface in the world frame and $\mathbf{x}^h \in \mathbb{R}^3$ is the corresponding contact point on the surface of a canonical open hand. The number of contacts varies across grasps, whereas the model operates on a fixed number of points. We therefore use K contact slots, where slot i holds the pair $(\mathbf{x}_i^o, \mathbf{x}_i^h)$, and stack them as the rows of $\mathbf{X}^o, \mathbf{X}^h \in \mathbb{R}^{K\times 3}$. To accommodate grasps with fewer than K contacts, we additionally learn to predict whether each slot holds a real contact (Section 3.3). Expressing $\mathbf{x}^h$ on the canonical hand decouples hand-contact semantics from the current hand pose and object placement. A hand contact then names the same hand surface location in every grasp, so similar contact patterns are represented consistently across grasps. Finally, we concatenate the keypoints and contact points into the grasp representation

$$\mathbf{X} = [\,\mathbf{X}^g;\, \mathbf{X}^o;\, \mathbf{X}^h\,] \in \mathbb{R}^{(S+2K)\times 3}. \tag{1}$$

3.3 Unified Flow Matching and Contact Set Prediction

Given the grasp and contact representation $\mathbf{X}$ of Equation 1, we jointly generate the grasp pose and hand–object contacts with a single flow-matching model. Keypoints have fixed anatomical correspondence across samples, whereas contacts form an unordered set, and only n of the K contact slots hold real contacts, with n varying across grasps. We therefore combine point-wise flow matching with query-based set prediction for contact generation.

Rectified flow formulation. Let $\mathbf{X}_1$ denote a clean sample of $\mathbf{X}$ in Equation 1, and let $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ be Gaussian noise of the same shape. We define the linear interpolation and its target velocity

$$\mathbf{X}_{\boldsymbol{\tau}} = \boldsymbol{\tau} \odot \mathbf{X}_1 + (1 - \boldsymbol{\tau}) \odot \boldsymbol{\epsilon}, \qquad \mathbf{V} = \mathbf{X}_1 - \boldsymbol{\epsilon}, \tag{2}$$

where $\boldsymbol{\tau} \in [0,1]^{S+2K}$ holds one noise level per point and $\odot$ broadcasts it over the three coordinates. We follow the convention that $\tau = 0$ is pure noise and $\tau = 1$ is the clean data. A single Diffusion Transformer (DiT) (Peebles & Xie, 2023) predicts the velocity of all points jointly, $\hat{\mathbf{V}} = v_{\boldsymbol{\psi}}(\mathbf{X}_{\boldsymbol{\tau}}, \boldsymbol{\tau}, \boldsymbol{\kappa})$, where $\boldsymbol{\kappa}$ is the global condition defined below. Throughout, the superscripts g, o, and h mark the pose, object-contact, and hand-contact parts of any matrix with one row per point, and a bold lowercase letter denotes a row, so $\hat{\mathbf{v}}_i^o$ is row i of $\hat{\mathbf{V}}^o$. A hat marks a model prediction.

Point-wise noise levels. At inference, we steer generation by fixing the contact points at their supplied values ($\tau = 1$) and denoising only the remaining points. A model trained with one noise level shared by all points never sees clean and noisy points together, so such partially clean inputs would be out of distribution at inference. We therefore follow Diffusion Forcing (Chen et al., 2024) and give each of the $2K$ object- and hand-contact points its own noise level,

$$\tau_i^o,\ \tau_i^h \sim \mathcal{U}(0,1). \tag{3}$$

The two points of a pair are sampled independently as well, so the model also sees pairs whose object and hand points are at different noise levels, as object-only and hand-only steering require. The S keypoints are always generated and never supplied as a condition, so they share a single noise level

$$\tau^g \sim \mathcal{U}(0,1). \tag{4}$$

To expose the model to the clean contact points it receives at inference, we augment the noise-level distribution of each real contact point. With probability $q = 0.08$, we set $\tau = 1$ instead of drawing

it from $\mathcal{U}(0,1)$, so the point is clean. Since this is decided independently for the object- and hand-contact points, a contact pair may have neither, one, or both of its points clean. The vector $\boldsymbol{\tau}$ collects τ^g for every keypoint and τ_i^o, τ_i^h for the contact points.

Token embedding. We embed each point as a token,

$$\mathbf{h}_j^g = \mathbf{W}\gamma(\mathbf{x}_j^g) + \mathbf{p}_j^g, \qquad \mathbf{h}_i^o = \mathbf{W}\gamma(\mathbf{x}_i^o) + \mathbf{p}_i + \mathbf{e}^o, \qquad \mathbf{h}_i^h = \mathbf{W}\gamma(\mathbf{x}_i^h) + \mathbf{p}_i + \mathbf{e}^h, \tag{5}$$

where γ is a multi-frequency Fourier feature mapping (Mildenhall et al., 2021; Sun et al., 2025) and $\mathbf{W}$ projects its output to the DiT hidden dimension. The learned embeddings $\mathbf{p}_j^g$ index keypoints, $\mathbf{p}_i$ indexes contact slots, and $\mathbf{e}^o, \mathbf{e}^h$ mark the object and hand side of a pair. Both tokens of a pair share $\mathbf{p}_i$, which tells the model they belong together. The local geometry embedding adds the object geometry around each token to $\mathbf{h}_j^g$ and $\mathbf{h}_i^o$. For each such token, we take its 4 nearest points in a 512-point subsample of $\mathbf{P}$, and add an MLP embedding of their offsets and distances from the token's current position (ablated in the appendix, Table 10).

Denoising transformer. The DiT is a stack of L blocks on all $S + 2K$ tokens. Each block applies full self-attention followed by an MLP, with a residual connection around both. In the self-attention of block b, we add a learnable scalar β_b, the *pair attention bias*, to the attention logits between the object and the hand contact tokens of each slot, in both directions, before the softmax. It allows the two tokens of a pair to attend to each other, and we initialize every β_b to 4. The global condition $\boldsymbol{\kappa} = [\,\mathbf{f}_{\text{obj}};\, \mathbf{f}_{\text{txt}}\,]$ concatenates the Uni3D (Zhou et al., 2024) object feature and a learned linear projection of the frozen CLIP (Radford et al., 2021) text feature. Standard DiT modulates all tokens with a single timestep embedding, which cannot express token-wise noise levels. We instead condition each token j on its own vector

$$\mathbf{c}_j = [\,\phi(\tau_j);\, \boldsymbol{\kappa}\,], \tag{6}$$

where τ_j is the noise level of token j and ϕ is a sinusoidal embedding followed by an MLP. Each block computes the AdaLN-Zero (Peebles & Xie, 2023) shift, scale, and gate of token j from $\mathbf{c}_j$. A linear head maps the final hidden state of each token to its velocity $\hat{\mathbf{v}}$. An additional small MLP head reads the concatenated final hidden states of the two contact tokens in slot i and predicts a logit $\hat{\alpha}_i$ that the slot holds a real contact.

Contact set matching. Contact slots have no fixed order, and the number of real contacts varies across grasps. We therefore assign slots to ground-truth contacts on the fly at every training step by bipartite matching, as in DETR (Carion et al., 2020). A training grasp with $n \le K$ contacts places them in n random slots $\mathcal{G} \subseteq \{1, \dots, K\}$. Dummy contacts at a fixed sentinel coordinate would be trivial to detect, so we fill the other $K - n$ slots with dummy pairs sampled from the same object and canonical-hand surfaces as real contacts, following DiffusionDet (Chen et al., 2023). We match slots to ground-truth contacts using the clean estimate $\hat{\mathbf{X}}_1 = \mathbf{X}_{\boldsymbol{\tau}} + (1 - \boldsymbol{\tau}) \odot \hat{\mathbf{V}}$ and the cost

$$C_{im} = \|\hat{\mathbf{x}}_i^o - \mathbf{x}_m^o\|_2 + \|\hat{\mathbf{x}}_i^h - \mathbf{x}_m^h\|_2 - \lambda_{\text{match}} \log \sigma(\hat{\alpha}_i), \qquad m \in \mathcal{G}, \tag{7}$$

where σ is the logistic sigmoid and $\lambda_{\text{match}} = 0.1$. A slot with a clean token already holds its ground-truth contact, so we fix its assignment to itself and match only the remaining slots. This keeps clean contacts in place and trains the other slots to complete the set around them. Matched slots receive the existence target $y_i = 1$ and unmatched slots $y_i = 0$.

Training objective. Every pose token receives flow supervision. A contact token receives it only if its slot is matched and the token itself is noisy, which gives the masks $w_i^o = y_i\, \mathbb{1}[\tau_i^o < 1]$ and $w_i^h = y_i\, \mathbb{1}[\tau_i^h < 1]$. In a pair with one clean token, the model thus learns to predict the missing side. The flow loss is the masked mean squared error

$$\mathcal{L}_{\text{flow}} = \frac{\sum_{j=1}^{S} \|\hat{\mathbf{v}}_j^g - \mathbf{v}_j^g\|_2^2 + \sum_{i=1}^{K} \big[w_i^o\, \|\hat{\mathbf{v}}_i^o - \mathbf{v}_i^o\|_2^2 + w_i^h\, \|\hat{\mathbf{v}}_i^h - \mathbf{v}_i^h\|_2^2\big]}{S + \sum_i (w_i^o + w_i^h)}. \tag{8}$$

The existence of a slot with a clean token is given by that token, so the existence loss covers only slots whose two tokens are both noisy, $r_i = \mathbb{1}[\tau_i^o < 1 \wedge \tau_i^h < 1]$,

$$\mathcal{L}_{\text{exist}} = \frac{\sum_i r_i\, \text{BCE}(\hat{\alpha}_i, y_i)}{\sum_i r_i}. \tag{9}$$

We minimize $\mathcal{L} = \mathcal{L}_{\text{flow}} + \mathcal{L}_{\text{exist}}$, averaged over training grasps, noise levels $\boldsymbol{\tau}$, and noise $\boldsymbol{\epsilon}$.

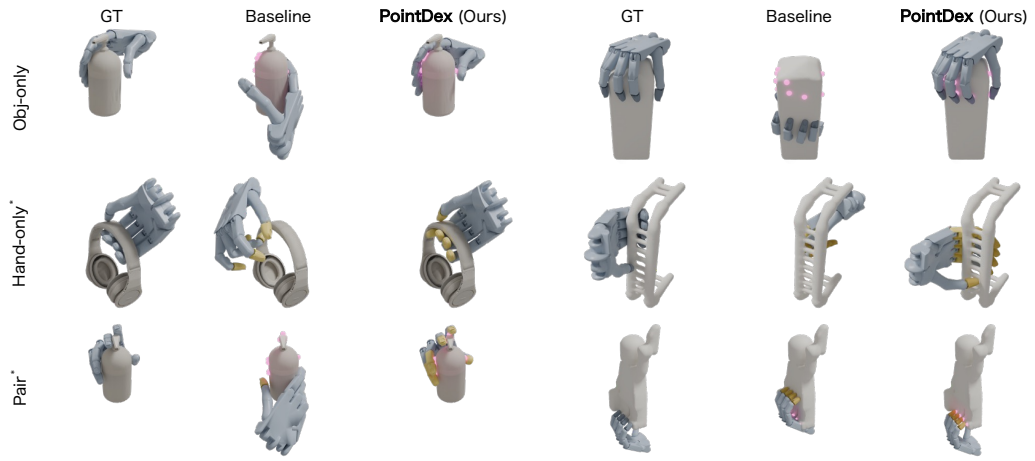


Figure 3: **Qualitative comparison with steering baselines.** Rows show obj-only, hand-only*, and pair* steering on DexGYS (Wei et al., 2024) (left) and Dexonomy (Chen et al., 2025a) (right). All methods receive the same steering contacts (● object points, ● hand links). The baselines are T(R,O) Grasp (Fei et al., 2025) (obj-only) and DextER (Lee et al., 2026) (hand-only*, pair*).

3.4 Steerable Grasp Sampling

At inference time, we jointly generate pose tokens, object-contact tokens, and hand-contact tokens in $\mathbf{X}_{\boldsymbol{\tau}}$ of Equation 1.

We call a contact supplied by the user as a condition a *steering contact*, its tokens *steering tokens*, and the remaining tokens *free tokens*, and write $\mathbf{m}$ for the binary mask that marks the steering tokens. Steering tokens remain clean, with a noise level 1 at every step, while the free tokens follow a shared uniform schedule $t_k = k/T$ with $T = 25$ Euler steps,

$$\boldsymbol{\tau}_k = (1 - \mathbf{m})\, t_k + \mathbf{m}, \tag{10}$$

so $\tau_{k,j} = 1$ exactly when token j is a steering token. Steering tokens then match the clean contact tokens seen in training (Section 3.3), and the model predicts the velocity of the free tokens conditioned on them. Starting from $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ at the free tokens and clean steering tokens, we integrate the learned velocity with an Euler step that updates only the free tokens,

$$\mathbf{X}_{\boldsymbol{\tau}_{k+1}} = \mathbf{X}_{\boldsymbol{\tau}_k} + \Delta t_k\, (1 - \mathbf{m}) \odot v_{\boldsymbol{\psi}}(\mathbf{X}_{\boldsymbol{\tau}_k}, \boldsymbol{\tau}_k, \boldsymbol{\kappa}), \qquad \Delta t_k = t_{k+1} - t_k. \tag{11}$$

Steering modes. The mask $\mathbf{m}$ determines the steering mode. Marking only object-contact tokens specifies where to grasp and leaves the contacting hand part free, for example grasping a mug by its handle without prescribing a finger arrangement. Marking only hand-contact tokens specifies which hand parts to use, for example a thumb–index pinch, and the model chooses the object contact locations. Marking both tokens of a contact pair specifies what touches where, for example the index fingertip on a trigger, and leaves the rest of the grasp unconstrained.

Langevin corrector. Following predictor–corrector sampling (Song et al., 2021), we run M Langevin steps (Song & Ermon, 2019) on the free tokens before each Euler step (ablated in Table 6). Since the free tokens share the noise level t_k, the linear path of Equation 2 turns the predicted velocity $\hat{\mathbf{V}} = v_{\boldsymbol{\psi}}(\mathbf{X}_{\boldsymbol{\tau}_k}, \boldsymbol{\tau}_k, \boldsymbol{\kappa})$ into the score

$$s_{\boldsymbol{\psi}}(\mathbf{X}_{\boldsymbol{\tau}_k}, \boldsymbol{\tau}_k, \boldsymbol{\kappa}) = -\frac{\mathbf{X}_{\boldsymbol{\tau}_k} - t_k\, \hat{\mathbf{X}}_1}{(1 - t_k)^2}, \qquad \hat{\mathbf{X}}_1 = \mathbf{X}_{\boldsymbol{\tau}_k} + (1 - t_k)\, \hat{\mathbf{V}}. \tag{12}$$

Starting from $\mathbf{X}^{(0)}_{\boldsymbol{\tau}_k} = \mathbf{X}_{\boldsymbol{\tau}_k}$, each step updates

$$\mathbf{X}^{(u+1)}_{\boldsymbol{\tau}_k} = \mathbf{X}^{(u)}_{\boldsymbol{\tau}_k} + (1 - \mathbf{m}) \odot \left[\eta_k\, s_{\boldsymbol{\psi}}(\mathbf{X}^{(u)}_{\boldsymbol{\tau}_k}, \boldsymbol{\tau}_k, \boldsymbol{\kappa}) + \sqrt{2\eta_k}\, (1 - t_k)^p\, \boldsymbol{\xi}^{(u)}\right], \tag{13}$$

with $\eta_k = c\,(1 - t_k)^2$ and $\boldsymbol{\xi}^{(u)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and the Euler step of Equation 11 continues from $\mathbf{X}^{(M)}_{\boldsymbol{\tau}_k}$. The factor $(1 - t_k)^p$ shrinks the injected noise toward the end of sampling. We use $M = 4$, $c = 0.2$, and $p = 0.5$, and apply the corrector only for $t_k \in [0.05, 0.95]$.

Table 1: **Language-conditioned grasp generation on the DexGYS validation set.** Baseline numbers are taken from DextER (Lee et al., 2026).

Method	Intention			Quality			Diversity		
	P-FID↓	CD↓	Con.↓ ($\times 10^{-1}$)	SR↑ (%)	Q_1↑ ($\times 10^{-1}$)	Pen.↓	δ_t ↑	δ_r ↑	δ_q ↑
GraspCVAE (Sohn et al., 2015)	29.02	3.14	0.96	29.12	0.54	0.55	0.18	1.76	0.18
GraspTTA (Jiang et al., 2021)	33.15	12.19	1.11	43.46	0.71	0.19	2.11	6.15	3.87
SceneDiffuser (Huang et al., 2023)	7.93	1.68	0.45	62.24	0.83	0.25	0.35	3.46	0.39
DGTR (Xu et al., 2024)	15.77	2.90	0.78	51.91	0.78	0.16	2.05	14.01	4.30
DexGYSNet (Wei et al., 2024)	5.60	1.20	0.36	63.31	0.83	0.22	6.12	55.68	6.12
DextER (Lee et al., 2026)	**0.20**	1.46	0.34	67.14	0.89	0.37	**8.84**	**77.98**	**13.63**
PointDex (Ours)	0.33	**1.03**	**0.23**	**83.76**	**1.56**	**0.13**	8.82	77.90	12.20

Table 2: **Ablation of pose representation.** Keypoint and joint-angle poses, both trained without the local geometry embedding.

	w/o contact steering	Pair			
Pose	SR ↑	SR ↑	d_{obj} ↓	d_{hand} ↓	d_{pair} ↓
Joint angle	63.30	63.93	10.04	9.31	20.57
Keypoint	82.44	81.21	5.39	4.56	7.41

Table 3: **Ablation of contact tokens.**

Variant	SR ↑
Full	83.76
w/o obj. contacts	81.77
w/o hand contacts	81.04
Pose only	79.81

Post-processing. We convert the S generated hand keypoints into the grasp $\mathbf{q}$. We first estimate the wrist pose $(\mathbf{R}_w, \mathbf{t}_w)$ by Kabsch alignment (Kabsch, 1976) between the generated palm keypoints and their counterparts in the local palm frame. With the wrist fixed, we recover the J finger joint angles $\boldsymbol{\theta}$ using iterative position-based inverse kinematics over the link-origin and fingertip keypoints. At each iteration, we linearize forward kinematics with the point Jacobians and solve a constrained quadratic program under joint limits and a ± 0.5 rad trust region. We run up to 8 iterations and use the resulting $\mathbf{q} = [\mathbf{t}_w, \text{axisangle}(\mathbf{R}_w), \boldsymbol{\theta}]$ for simulation and contact evaluation.

4 Experiments

4.1 Experimental Setup

We evaluate grasp generation on the DexGYS (Wei et al., 2024) and Dexonomy (Chen et al., 2025a) benchmarks. Both use a hand with $J = 22$ finger joints, on which we place $S = 29$ keypoints (Appendix A). For each hand link in contact, we annotate one contact as the object surface point closest to the link, paired with the hand point it touches, expressed in the canonical hand frame. For the steering experiments, we additionally measure steering fidelity, i.e., how closely a generated grasp realizes the specified steering contacts, using contact distance d, defined as the mean Euclidean distance in mm (↓). We distinguish three distances based on the supplied steering specification. The object-contact distance d_{obj} measures the distance from each specified object contact to the nearest point on the posed hand, while the hand-contact distance d_{hand} measures the distance from each specified hand contact to the nearest point on the object. When both sides of a contact pair are specified, the pair distance d_{pair} measures the distance between the specified hand contact and its paired object contact. We report each metric under its corresponding steering mode. Details of the benchmarks and training setup are provided in Appendix A.

4.2 Grasps and Contacts as a Shared Representation of 3D Points

We show that the shared 3D point representation drives grasp quality. PointDex outperforms prior methods on language-conditioned generation. Following the protocol of DextER (Lee et al., 2026) on the DexGYS validation set (Table 1), it reaches a success rate of 83.76% without steering input, 16.62 points above DextER, and obtains the highest grasp quality ($Q_1 = 1.56$) and the lowest penetration (0.13), Chamfer distance (1.03), and contact-map distance (0.23), with diversity comparable to DextER.

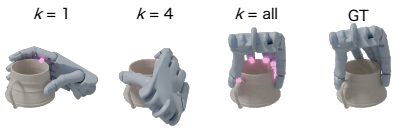
Table 4: **Steerable grasp generation on DexGYS.** * marks settings in which hand contacts are given as links. d^*_{hand} and d^*_{pair} then take the nearest point on the specified link in place of a specified hand point.

	Obj-only		Hand-only*		Pair*			
Method	SR ↑	d_{obj} ↓	SR ↑	d^*_{hand} ↓	SR ↑	d_{obj} ↓	d^*_{hand} ↓	d^*_{pair} ↓
T(R,O) Grasp (Fei et al., 2025)	48.12	18.19	–	–	–	–	–	–
DextER (Lee et al., 2026)	–	–	61.39	7.38	61.15	17.23	7.00	31.10
PointDex (Ours)	**83.22**	**6.26**	**88.87**	**4.10**	**82.89**	**5.62**	**3.72**	**6.14**
GT Floor	–	4.70	–	3.67	–	4.70	3.67	5.02

Table 5: **Effect of the number of steering contacts k.**

	Obj-only		Hand-only		Pair			
k	SR ↑	d_{obj} ↓	SR ↑	d_{hand} ↓	SR ↑	d_{obj} ↓	d_{hand} ↓	d_{pair} ↓
1	83.10	7.53	83.10	6.40	82.39	6.14	5.24	9.98
2	83.36	7.06	83.39	6.33	82.05	5.81	4.97	8.99
4	83.05	6.67	82.76	6.13	81.37	5.60	4.78	8.21
all	83.22	6.26	83.23	5.85	81.58	5.46	4.66	7.47
GT Floor	–	4.70	–	3.96	–	4.70	3.96	4.98

Figure 4: **Effect of k.** The grasp nears the ground truth (GT) as k grows.



Pose representation. Replacing the keypoint pose with a joint-angle pose, with the local geometry embedding disabled in both since it queries object anchors at token positions, drops the SR without contact steering from 82.44% to 63.30% and the pair-steering SR from 81.21% to 63.93%, and raises d_{obj}, d_{hand}, and d_{pair} from 5.39, 4.56, and 7.41 mm to 10.04, 9.31, and 20.57 mm (Table 2). A joint-angle pose must be related to the contacts through forward kinematics, whereas keypoints share their frame.

Contact tokens. Replacing the object or hand contact tokens with learnable tokens, which keeps the token count fixed, drops the SR from 83.76% to 81.77% and 81.04%, and replacing both drops it to 79.81% (Table 3). The loss thus comes from the missing contact information rather than from model capacity.

The gain in Table 1 therefore requires both parts of the shared representation, namely the pose as 3D keypoints and the contacts as 3D points in the same space.

4.3 Steerable Grasp Generation

Because hand and object contacts are separate points with independent noise levels, a single PointDex model supports object-contact, hand-contact, and correspondence-level steering, and follows each more closely than baselines built for one interface. We give all methods the same steering contacts, taken from our contact annotation (Appendix A.1.3) and translated into the conditioning interface of each method (Appendix A.1.6), in three settings, namely *obj-only* (object-surface contact points), *hand-only* (contacting hand links), and *pair* (paired object points and hand links). Each baseline is evaluated only on the settings its interface supports.

PointDex achieves lower contact distances than the corresponding baselines in every setting (Table 4). Under *obj-only* steering, d_{obj} decreases from 18.19 mm for T(R,O) Grasp to 6.26 mm. Under *hand-only* steering, d^*_{hand} decreases from 7.38 mm for DextER to 4.10 mm. The gap is largest under *pair* steering, where PointDex reduces d_{obj}, d^*_{hand}, and d^*_{pair} from 17.23, 7.00, and 31.10 mm to 5.62, 3.72, and 6.14 mm, while achieving higher SR in all settings. Figure 3 (left) shows DexGYS examples, in which the baselines grasp the object away from the specified contacts.

Number of steering contacts. Since each contact token has its own noise level, the same model accepts any number of steering contacts without retraining. As k grows from 1 to all, every contact distance decreases (Table 5), with d_{pair} falling from 9.98 mm to 7.47 mm, while SR changes only modestly. Figure 4 shows one sample in which the grasp approaches the ground-truth configuration as k grows.

Table 6: **Langevin corrector and pair attention bias.** All variants use the same number of forward passes, and *w/o pair bias* is retrained without the bias.

	w/o contact steering	Pair			
Variant	SR ↑	SR ↑	d_{obj} ↓	d_{hand} ↓	d_{pair} ↓
Full	83.76	81.58	5.46	4.66	7.47
w/o Langevin	76.42	78.77	5.78	4.96	8.38
w/o pair bias	78.14	81.05	6.92	5.47	12.49

Figure 5: **Effect of the pair attention bias β.** With β, the specified hand contact (●) meets its paired object contact (●). Without β, the hand contact lands away from its paired object contact.



Table 7: **Steerable grasp generation on Dexonomy.** Settings follow Table 4.

	Obj-only		Hand-only*		Pair*			
Method	SR ↑	d_{obj} ↓	SR ↑	d^*_{hand} ↓	SR ↑	d_{obj} ↓	d^*_{hand} ↓	d^*_{pair} ↓
T(R,O) Grasp (Fei et al., 2025)	7.92	15.69	–	–	–	–	–	–
DextER (Lee et al., 2026)	–	–	19.85	7.16	32.66	5.59	4.12	6.66
PointDex (Ours)	**47.57**	**5.10**	**45.24**	**3.50**	**44.61**	**4.17**	**3.21**	**4.55**
GT Floor	–	3.22	–	3.06	–	3.22	3.06	3.30

4.4 Ablation Studies

Langevin corrector. Removing the corrector at the same number of forward passes lowers the SR without contact steering from 83.76% to 76.42% and the pair-steering SR from 81.58% to 78.77%, and raises every contact distance, with d_{pair} going from 7.47 to 8.38 mm (Table 6). Since the compute is matched, the gain is not an effect of extra network evaluations.

Pair attention bias. Removing the pair attention bias lowers the SR without contact steering from 83.76% to 78.14% and leaves the pair-steering SR nearly unchanged, from 81.58% to 81.05%, but raises d_{pair} from 7.47 to 12.49 mm (Table 6). Without the bias, the object and hand tokens that share a slot embedding $\mathbf{p}_i$ often no longer mark the same contact (Figure 5). Appendix A.2 further reports the effect of the initial value of the bias (Table 9) and of the local geometry embedding (Table 10).

4.5 Dexonomy Benchmark

We repeat the comparison of Section 4.3 on the Dexonomy benchmark with the same steering contacts, the same link-to-contact mapping, and the same contact distances. Table 7 reports the results, and Figure 3 (right) shows examples. PointDex again outperforms the baselines in every setting they support. Under *obj-only* steering, it raises SR from 7.92% for T(R,O) Grasp to 47.57% and lowers d_{obj} from 15.69 to 5.10 mm. Under *hand-only* steering, SR rises from 19.85% for DextER to 45.24% and d^*_{hand} falls from 7.16 to 3.50 mm. Under *pair* steering, SR rises from 32.66% to 44.61%, and d_{obj}, d^*_{hand}, and d^*_{pair} drop from 5.59, 4.12, and 6.66 mm to 4.17, 3.21, and 4.55 mm.

5 Conclusion

We have presented PointDex, which represents grasp poses, object contacts, and hand contacts as a shared representation of 3D points and denoises them with a single flow-matching transformer at per-token noise levels. Steering then amounts to conditioning on any subset of contacts, so one model supports object-contact, hand-contact, and correspondence-level steering without retraining. PointDex raises language-conditioned grasp success on DexGYS by 16.6 points over DextER and reduces the pair distance of the steering baselines by up to 80%, coming within 2 mm of the ground-truth floor on both benchmarks.

Limitations. All experiments use the Shadow Hand. The representation itself is not tied to this hand, since posed hand keypoints and paired contact points can be defined on any hand surface, but each new hand requires its own keypoint layout and inverse kinematics, and we have not tested transfer to other robot hands or to human grasp data.

AI USE STATEMENT

In this work, we used generative AI tools to implement methods and to assist with translation. We have not used generative AI tools to develop conceptual frameworks, propose or refine hypotheses, design or provide feedback on the methodology or experiments, generate synthetic datasets, clean or reformat datasets, or interpret results, and formulating or proving mathematical claims and qualitative data analysis are not applicable to this work. Additionally, we used generative AI tools to create and edit software code, create and modify figures, draft parts of the paper, edit the paper for readability, identify and summarize relevant literature, and format and check references. We have reviewed all AI-assisted work. The authors reviewed and tested all AI-generated code, checked every reported number against the experiment logs, verified every reference against its published version, and reviewed and revised all text. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work uses publicly available grasp datasets and simulation only, and involves no human subjects or personal data. We do not foresee direct negative societal impact beyond that of general-purpose robotic manipulation.

REPRODUCIBILITY STATEMENT

Section 3 describes the model, including the token layout (Section 3.2), the training objective and noise-level sampling (Section 3.3), and the sampler with the Langevin corrector and the inverse-kinematics post-processing (Section 3.4). Appendix A.1.5 gives the architecture and all training settings (Table 8), and Appendix A.1.3 describes how the contact pairs are annotated from the ground-truth grasps. Appendices A.1.1 and A.1.2 give the data splits and evaluation protocols of both benchmarks, and Appendix A.1.4 defines the steering metrics and the ground-truth floor. Appendix A.1.6 describes how each baseline is trained or loaded and how the steering contacts are passed to it. We will release the code, configurations, contact annotations, and checkpoints upon publication.

REFERENCES

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pp. 213–229. Springer, 2020.

Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. volume 37, pp. 24081–24125, 2024.

Jiayi Chen, Yubin Ke, Lin Peng, and He Wang. Dexonomy: Synthesizing all dexterous grasp types in a grasp taxonomy. *Robotics: Science and Systems*, 2025a.

Jiayi Chen, Yubin Ke, and He Wang. Bodex: Scalable and efficient robotic dexterous grasp synthesis using bilevel optimization. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 01–08. IEEE, 2025b.

Shoufa Chen, Peize Sun, Yibing Song, and Ping Luo. Diffusiondet: Diffusion model for object detection. In *2023 IEEE/CVF international conference on computer vision (ICCV)*, pp. 19773–19786. IEEE, 2023.

William Chen, Jagdeep Singh Bhatia, Catherine Glossop, Nikhil Mathihalli, Ria Doshi, Andy Tang, Danny Driess, Karl Pertsch, and Sergey Levine. Steerable vision-language-action policies for embodied reasoning and hierarchical control. 2026.

Xin Fei, Zhixuan Xu, Huaicong Fang, Tianrui Zhang, and Lin Shao. T (r, o) grasp: Efficient graph diffusion of robot-object spatial transformation for cross-embodiment dexterous grasping, 2025.

Jiayuan Gu, Sean Kirmani, Paul Wohlhart, Yao Lu, Montserrat Gonzalez Arenas, Kanishka Rao, Wenhao Yu, Chuyuan Fu, Keerthana Gopalakrishnan, Zhuo Xu, et al. Rt-trajectory: Robotic task generalization via hindsight trajectory sketches. 2023.

Xintong Hu, Xuhong Huang, Jinyu Zhang, Yutong Yao, Yuchong Sun, Qiuyue Wang, Mingsheng Li, Sicheng Xie, Yitao Liu, Junhao Chen, et al. Finevla: Fine-grained instruction alignment for steerable vision-language-action policies. *arXiv preprint arXiv:2605.27284*, 2026.

Siyuan Huang, Zan Wang, Puhao Li, Baoxiong Jia, Tengyu Liu, Yixin Zhu, Wei Liang, and Song-Chun Zhu. Diffusion-based generation, optimization, and planning in 3d scenes. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16750–16761. IEEE, 2023.

Hanwen Jiang, Shaowei Liu, Jiashun Wang, and Xiaolong Wang. Hand-object contact consistency reasoning for human grasps generation. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11087–11096. IEEE, 2021.

Wolfgang Kabsch. A solution for the best rotation to relate two sets of vectors. *Foundations of Crystallography*, 32(5):922–923, 1976.

Junha Lee, Eunha Park, and Minsu Cho. Dexter: Language-driven dexterous grasp generation with embodied reasoning. 2026.

Haoming Li, Xinzhuo Lin, Yang Zhou, Xiang Li, Yuchi Huo, Jiming Chen, and Qi Ye. Contact2grasp: 3d grasp synthesis via hand-object contact constraint. 2022.

Peiming Li, Ziyi Wang, Mengyuan Liu, Hong Liu, and Chen Chen. Clickdiff: click to induce semantic contact map for controllable grasp generation with diffusion models. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pp. 273–281, 2024.

Puhao Li, Tengyu Liu, Yuyang Li, Yiran Geng, Yixin Zhu, Yaodong Yang, and Siyuan Huang. Gendexgrasp: Generalizable dexterous grasping. In *2023 IEEE international conference on robotics and automation (ICRA)*, pp. 8068–8074. IEEE, 2023.

Shaowei Liu, Yang Zhou, Jimei Yang, Saurabh Gupta, and Shenlong Wang. Contactgen: Generative contact modeling for grasp generation. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 20552–20563. IEEE, 2023.

Jiaxin Lu, Hao Kang, Haoxiang Li, Bo Liu, Yiding Yang, Qixing Huang, and Gang Hua. Ugg: Unified generative grasping. In *European Conference on Computer Vision*, pp. 414–433. Springer, 2024.

Julien Mérand, Boris Meden, Liming Chen, and Mathieu Grossard. Cotograsp: Contact-topology-conditioned dexterous grasp synthesis via canonical workspace learning. In *European Conference on Computer Vision*, pp. 116–133. Springer, 2026.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. volume 65, pp. 99–106. ACM New York, NY, USA, 2021.

Alex Nichol, Heewoo Jun, Prafulla Dhariwal, Pamela Mishkin, and Mark Chen. Point-e: A system for generating 3d point clouds from complex prompts. *arXiv preprint arXiv:2212.08751*, 2022.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4172–4182. IEEE, 2023.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Laura Smith, Alex Irpan, Montserrat Gonzalez Arenas, Sean Kirmani, Dmitry Kalashnikov, Dhruv Shah, and Ted Xiao. Steer: Flexible robotic manipulation via dense language grounding. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 16517–16524. IEEE, 2025.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. *Advances in Neural Information Processing Systems*, 28, 2015.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations (ICLR)*, 2021.

Yue Su, Xinyu Zhan, Hongjie Fang, Yong-Lu Li, Cewu Lu, and Lixin Yang. Motion before action: Diffusing object motion as manipulation condition. *IEEE Robotics and Automation Letters*, 10 (7):7428–7435, 2025.

Tao Sun, Liyuan Zhu, Shengyu Huang, Shuran Song, and Iro Armeni. Rectified point flow: Generic point cloud pose estimation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

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

Ruicheng Wang, Jialiang Zhang, Jiayi Chen, Yinzhen Xu, Puhao Li, Tengyu Liu, and He Wang. DexGraspNet: A large-scale robotic dexterous grasp dataset for general objects based on simulation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 11359–11366, 2023.

Yilin Wang, Chuan Guo, Li Cheng, and Hai Jiang. Regiongrasp: A novel task for contact region controllable hand grasp generation. In *European Conference on Computer Vision*, pp. 33–48. Springer, 2024.

Yi-Lin Wei, Jian-Jian Jiang, Chengyi Xing, Xian-Tuo Tan, Xiao-Ming Wu, Hao Li, Mark Cutkosky, and Wei-Shi Zheng. Grasp as you say: Language-guided dexterous grasp generation. *Advances in Neural Information Processing Systems*, 37:46881–46907, 2024.

Zehang Weng, Haofei Lu, Danica Kragic, and Jens Lundell. DexDiffuser: Generating dexterous grasps with diffusion models. *IEEE Robotics and Automation Letters*, 9(12):11834–11840, 2024. doi: 10.1109/LRA.2024.3498776.

Guo-Hao Xu, Yi-Lin Wei, Dian Zheng, Xiao-Ming Wu, and Wei-Shi Zheng. Dexterous grasp transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17933–17942, 2024.

Yinzhen Xu, Weikang Wan, Jialiang Zhang, Haoran Liu, Zikang Shan, Hao Shen, Ruicheng Wang, Haoran Geng, Yijia Weng, Jiayi Chen, Tengyu Liu, Li Yi, and He Wang. UniDexGrasp: Universal robotic dexterous grasping via learning diverse proposal generation and goal-conditioned policy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4737–4746, 2023.

Yiming Zhong, Qi Jiang, Jingyi Yu, and Yuexin Ma. DexGrasp Anything: Towards universal robotic dexterous grasping with physics awareness. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22584–22594, 2025. doi: 10.1109/CVPR52734.2025.02103.

Junsheng Zhou, Jinsheng Wang, Baorui Ma, Yu-Shen Liu, Tiejun Huang, and Xinlong Wang. Uni3d: Exploring unified 3d representation at scale. In *International Conference on Learning Representations*, volume 2024, pp. 46766–46782, 2024.

Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5738–5746. IEEE, 2019.

Binghui Zuo, Lin Zhou, Haoxuan Xu, Jianan Yan, Zhipeng Yu, Zekai Liu, and Yangang Wang. Maskdexgrasp: Generative masked modeling for part-aware dexterous grasp synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 29598–29609, 2026.

A Appendix

A.1 Experimental Details

A.1.1 DexGYS Benchmark

Dataset. DexGYS (Wei et al., 2024) pairs 3D object point clouds with natural-language instructions and the corresponding dexterous grasps. We evaluate on the DexGYS validation set.

Metrics. We use the nine metrics of DextER (Lee et al., 2026), grouped by what they test. *Intention* metrics compare generated grasps with the reference grasps of the same instruction. P-FID computes the Fréchet distance between the distributions of generated and reference hands in the feature space of a point-cloud encoder (Nichol et al., 2022). CD is the Chamfer distance between each generated hand mesh and its ground-truth mesh. Con. compares where the two grasps touch the object, as the L2 distance between their contact maps over the object surface. *Quality* metrics test physical plausibility. Success is the fraction of grasps that pass the simulation test below, Q_1 is the force-closure quality, i.e., the smallest external wrench that breaks the grasp, and Pen. is the deepest penetration of the hand mesh into the object point cloud. *Diversity* metrics δ_t, δ_r, and δ_q are the standard deviations of the palm position, the palm orientation, and the joint angles over the samples generated for one input.

Success rate. We run the Isaac Gym test of DexGraspNet (Wang et al., 2023), as in DexGYS (Wei et al., 2024). The object is simulated under each of six gravity directions for 100 steps, and the grasp counts as successful if the hand keeps the object in stable contact for at least one direction and penetrates it by no more than 0.1 cm.

A.1.2 Dexonomy Benchmark

Dataset. We use the subset of Dexonomy (Chen et al., 2025a) released by DextER (Lee et al., 2026), which pairs 3D object point clouds with natural-language instructions. We evaluate on the seen validation split used by DextER.

Success rate. We use the MuJoCo (Todorov et al., 2012) test of DexGraspBench (Chen et al., 2025b). Six external forces are applied to the grasped object in turn, and the grasp counts as successful if the object ends within 5 cm and 15° of its initial pose under every force, with no penetration deeper than 1 cm.

A.1.3 Contact Annotation

For each grasp, we first collect candidate contact points, i.e., object surface points that lie close to the posed hand, each assigned to a hand link. We then build one contact pair $(\mathbf{x}^o, \mathbf{x}^h)$ for every link in contact. The object contact $\mathbf{x}^o$ is the candidate of that link closest to the hand surface, and the hand contact $\mathbf{x}^h$ is the nearest point to $\mathbf{x}^o$ on the same link of the posed hand, expressed on the canonical open hand. The pair therefore marks a near-contact rather than a coincident one: $\mathbf{x}^o$ lies on average 2.1 mm (DexGYS) and 1.9 mm (Dexonomy) from the hand surface.

A.1.4 Steering Metrics

This section gives the formal definitions of the contact distance metrics summarized in Section 4.1. Each measures how close a steering contact is to its realization in the generated grasp, as a Euclidean distance in mm.

Notation. We convert the generated pose tokens into the wrist pose and joint angles $\hat{\mathbf{q}}$ with inverse kinematics. Posing the canonical open-hand point cloud $\mathbf{P}^c_{\text{hand}} = \{\mathbf{h}^c_j\}_{j=1}^{N_{\text{hand}}}$ ($N_{\text{hand}} = 1024$) with $\hat{\mathbf{q}}$ gives the generated hand $\hat{\mathbf{P}}_{\text{hand}} = \{\hat{\mathbf{h}}_j\}_{j=1}^{N_{\text{hand}}}$, where $\hat{\mathbf{h}}_j$ is the location of canonical point $\mathbf{h}^c_j$ in the generated grasp. The object point cloud is $\mathbf{P}_{\text{obj}} = \{\mathbf{o}^{(l)}\}_{l=1}^{N_{\text{obj}}}$. We index the steering contacts supplied for a grasp by m. An object steering contact $\mathbf{s}^o_m$ is a point on the object surface. A hand steering contact is a canonical hand point $\mathbf{s}^h_m = \mathbf{h}^c_{j(m)}$, so its location in the generated grasp is simply $\hat{\mathbf{h}}_{j(m)}$.

Object-contact distance. For a set $\mathcal{S}_o$ of object steering contacts,

$$d_{\text{obj}} = \frac{1}{|\mathcal{S}_o|} \sum_{m \in \mathcal{S}_o} \min_j \| \mathbf{s}^o_m - \hat{\mathbf{h}}_j \|, \tag{14}$$

which measures whether some part of the hand reaches each specified object point.

Hand-contact distance. For a set $\mathcal{S}_h$ of hand steering contacts,

$$d_{\text{hand}} = \frac{1}{|\mathcal{S}_h|} \sum_{m \in \mathcal{S}_h} \min_l \| \hat{\mathbf{h}}_{j(m)} - \mathbf{o}^{(l)} \|, \tag{15}$$

which measures whether each specified hand point reaches the object.

Pair distance. For a set $\mathcal{S}_p$ of contact pairs $(\mathbf{s}^o_m, \mathbf{s}^h_m)$,

$$d_{\text{pair}} = \frac{1}{|\mathcal{S}_p|} \sum_{m \in \mathcal{S}_p} \| \hat{\mathbf{h}}_{j(m)} - \mathbf{s}^o_m \|, \tag{16}$$

which measures whether each specified hand point reaches its paired object point. Both endpoints are fixed by the specification, so this is the strictest of the three distances.

Link-level distances. In Table 4, each hand contact is given as a hand link a_m rather than a hand point. Let $L(a)$ index the points of $\hat{\mathbf{P}}_{\text{hand}}$ that belong to link a. We replace the hand point $\hat{\mathbf{h}}_{j(m)}$ with the point of link a_m that is closest to the target,

$$d^*_{\text{hand}} = \frac{1}{|\mathcal{S}_h|} \sum_{m \in \mathcal{S}_h} \min_{j \in L(a_m)} \min_l \| \hat{\mathbf{h}}_j - \mathbf{o}^{(l)} \|, \qquad d^*_{\text{pair}} = \frac{1}{|\mathcal{S}_p|} \sum_{m \in \mathcal{S}_p} \min_{j \in L(a_m)} \| \hat{\mathbf{h}}_j - \mathbf{s}^o_m \|. \tag{17}$$

They measure whether the specified link reaches the object and its paired object point, respectively.

GT floor. The GT floor rows apply the same distances to the ground-truth grasp of each test sample. Because the distances are measured between finite point clouds, they cannot reach zero even for the ground truth: the nearest point of the 1024-point hand cloud to a contact is generally a few millimeters away. The floor is thus the lowest value this evaluation can report.

A.1.5 Implementation Details

Model. The grasp representation $\mathbf{X}$ holds $S = 29$ pose tokens, namely eight palm points, sixteen link origins, and five fingertips, and $K = 16$ contact slots, for 61 tokens in total. Each coordinate is encoded by its raw value and Fourier features with 10 frequency bands, followed by a linear projection. The DiT has 6 blocks with hidden size 512, 8 attention heads, and dropout 0.1, and embeds each noise level into 256 dimensions. The object encoder is Uni3D-Ti (Zhou et al., 2024), initialized from its pretrained weights and fine-tuned with the rest of the model. The text encoder is a frozen CLIP ViT-B/16 (Radford et al., 2021) followed by a learned linear projection to 512 dimensions. The local geometry embedding reads the 4 nearest object anchors of each pose and object-contact token, the pair attention bias is initialized to 4 in every block, and the existence head is an MLP with hidden size 256. The model has 59.9M trainable parameters, and 123.0M including the frozen text encoder.

Training. Table 8 lists the training settings. Noise levels are drawn as described in Section 3.3.

Table 8: **Training settings.**

Setting	DexGYS	Dexonomy
Training steps	50k	75k
Batch size	128	128
GPUs (NVIDIA B200)	1	2
Optimizer	AdamW	
Learning rate	$2 \times 10^{-4} \rightarrow 2 \times 10^{-5}$	
LR schedule	cosine	
Weight decay	10^{-4}	
Gradient clipping	1.0	
Precision	mixed	
Observed-token prob. q	0.08	

A.1.6 Steering Comparison Setup

To compare steerable grasp generation between the baselines and PointDex in Tables 4 and 7, this section describes how each model is trained and how the steering contacts are given to it as conditioning. Since the baselines cannot take a point on the hand surface, these comparisons specify every hand contact only by its link.

Baselines. T(R,O) Grasp (Fei et al., 2025) does not take a language instruction. We add one by encoding the instruction with a frozen CLIP text encoder (Radford et al., 2021), projecting it with a learnable linear layer, and concatenating it to the diffusion timestep embedding. We train the model on the training set of each benchmark. For *obj-only* steering, we follow its contact guidance. The given object contacts are turned into a Gaussian heatmap over the object points. At each denoising step, we place the hand at the predicted clean pose and move it by one gradient step that reduces the heat-weighted distance from the object points to the hand surface. Since T(R,O) Grasp cannot take hand contacts, we evaluate it only without contact steering and under *obj-only*. For DextER (Lee et al., 2026), we use the checkpoints released for each benchmark and evaluate them without retraining. Since DextER does not take object contacts alone, we evaluate it without contact steering and under *hand-only* and *pair*.

PointDex. We map each link to a fixed hand steering contact, defined as the mean training-set contact on that link projected onto its surface. Since this mapping is fixed per link, it does not use any information from the test grasp.

A.2 Additional Results

A.2.1 Pair Attention Bias

Removing the pair attention bias degrades steering (Table 6). Table 9 shows how the results vary with the initial value of β_b; we use 4.

Table 9: **Effect of the initial value of the pair attention bias.** Each row initializes every β_b to the listed value.

Initial β_b	Pair SR ↑	d_{obj} ↓	d_{hand} ↓	d_{pair} ↓
0	79.98	5.68	4.95	8.03
2	80.93	5.46	4.68	7.45
4	81.58	5.46	4.66	7.47
8	80.88	5.48	4.69	7.22

A.2.2 Local Geometry Embedding

Removing the local geometry embedding lowers the SR without contact steering from 83.76% to 82.44% (Table 10).

Table 10: **Ablation of the local geometry embedding.** The variant is retrained without the embedding, with the rest of the recipe unchanged.

	w/o contact steering	Pair			
Variant	SR ↑	SR ↑	d_{obj} ↓	d_{hand} ↓	d_{pair} ↓
Full	83.76	81.58	5.46	4.66	7.47
w/o local geometry embedding	82.44	81.21	5.39	4.56	7.41

A.2.3 DEXONOMY BENCHMARK

We evaluate grasp generation without contact steering on the four Dexonomy evaluation splits of DextER (Lee et al., 2026) (Table 11). PointDex achieves the highest SR on every split, and on the unseen-grasp splits it keeps an SR above 33%, where DextER falls to about 11%.

Table 11: **Grasp generation without contact steering on the four Dexonomy evaluation splits.** The splits follow DextER and hold out the objects, the grasp types, or both.

	Seen Obj. & Grasp		Unseen Obj.		Unseen Grasp		Unseen Obj. & Grasp	
Method	P-FID ↓	SR ↑	P-FID ↓	SR ↑	P-FID ↓	SR ↑	P-FID ↓	SR ↑
T(R,O) Grasp (Fei et al., 2025)	1.75	32.46	1.77	31.43	2.58	28.41	2.43	27.39
DextER (Lee et al., 2026)	**0.21**	24.54	**0.29**	19.19	1.68	11.46	1.71	10.70
PointDex (Ours)	0.35	**39.37**	0.35	**37.63**	**1.31**	**33.15**	**1.21**	**33.32**

A.2.4 QUALITATIVE RESULTS

Figure 6 compares PointDex with the steering baselines on Dexonomy, and Figure 7 shows how the grasp changes with the number of steering contacts on DexGYS. Figure 8 shows randomly drawn Dexonomy test samples.

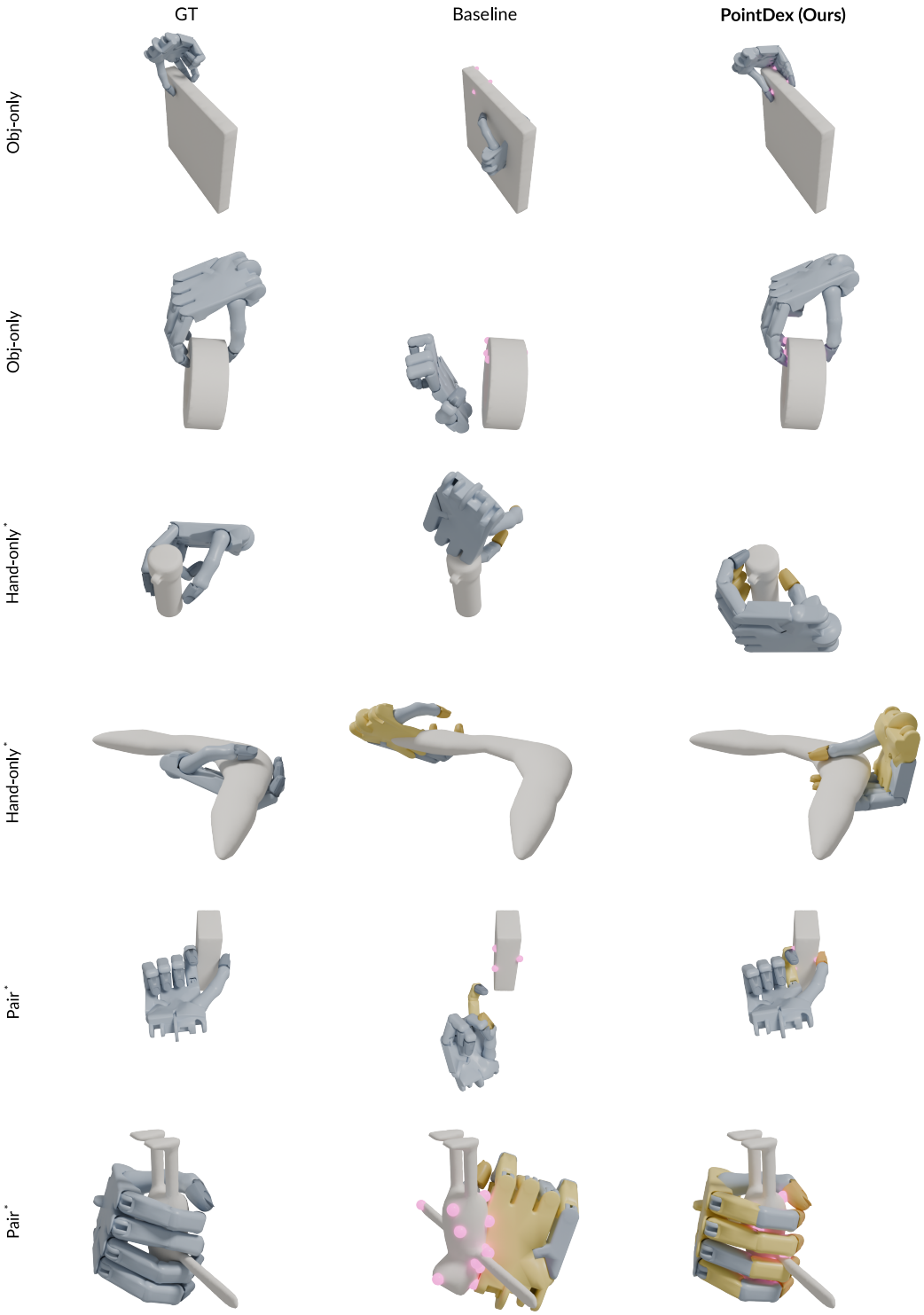


Figure 6: **Additional comparison with steering baselines on Dexonomy.** Rows show obj-only, hand-only*, and pair* steering, one test sample per row, with the settings of Table 7. All methods receive the same steering contacts (● object points, ● hand links). The baselines are T(R,O) Grasp (Fei et al., 2025) (obj-only) and DextER (Lee et al., 2026) (hand-only*, pair*).

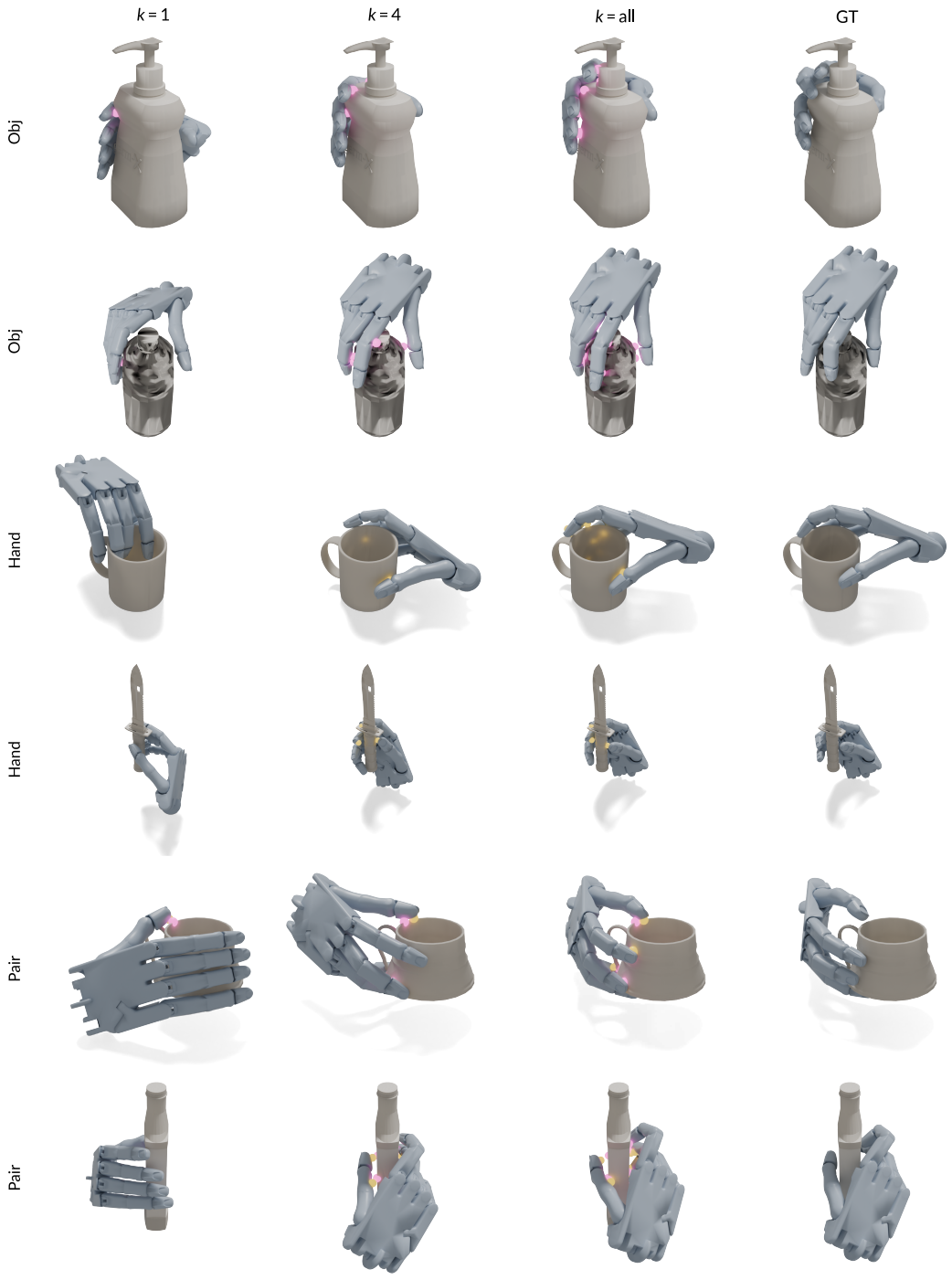


Figure 7: **Effect of the number of steering contacts k.** PointDex on DexGYS with k nested steering contacts (● object points, ● hand points) under obj-only, hand-only, and pair steering, as in Table 5. The grasp approaches the ground truth (GT) as k grows.



Figure 8: **Random samples on Dexonomy.** Six test samples drawn uniformly at random, with every steering setting of PointDex (steering contacts shown as ● object points and ● hand points).