PIXELDEC: GENERATIVE SCENE DECOMPOSITION WITH SUPERQUADRICS FROM A SINGLE IMAGE

Anonymous authors
Paper under double-blind review



Fig. 1: PIXELDEC takes a single RGB image (*left*) and generates a pixel-aligned 3D scene decomposition consisting of superquadric primitives (*middle*), a lightweight representation useful for downstream applications. For example, since superquadric parameters are directly interpretable, they form a token-efficient actionable scene representation for LLM-based robotic planning and reasoning (*right*).

ABSTRACT

We present PIXELDEC, an approach for generating pixel-aligned superquadric decompositions from a single RGB image. Prior work has shown the utility of compact superquadric representations for downstream applications. However, existing methods largely follow a deterministic reconstruction paradigm, requiring complete 3D point clouds as input and thus limiting their applicability to real-world settings, where complete 3D observations are rarely available. We propose a generative formulation of primitive-based decomposition that reconstructs superquadrics directly from a single image. It uses superquadrics not only as its output representation, but also as a compact representation space for modeling the distribution of plausible 3D shapes. When conditioned on single-view observations, PIXELDEC substantially outperforms prior superquadric decomposition methods operating on partial point clouds and achieves competitive performance with state-of-the-art single-view object-centric 3D scene reconstruction methods, with substantially lower runtime and memory requirements. Finally, we demonstrate that the generated superquadrics form an actionable representation that can be directly interpreted by an LLM and leveraged for downstream robotic tasks.

1 INTRODUCTION

Representing a scene with a small set of parametric primitives provides a compact and directly actionable description of its geometry. Superquadrics (Barr, 1981) are particularly well suited for this purpose: with only a few parameters, they span cuboids, cylinders, ellipsoids, and intermediate shapes, while sets of superquadrics can compactly approximate complex objects. Their explicit geometry also makes them useful for downstream interaction: grasps can be sampled in a superquadric's canonical frame (Tu & Desingh, 2025; Xu et al., 2025), collisions can be checked efficiently (Kim et al., 2023; Brunke et al., 2025; Zhao et al., 2026a), and vision-language models can manipulate scene primitives directly through language (Sella et al., 2026).

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.

Such decompositions were traditionally obtained through per-shape optimization, first by fitting superquadrics directly to data (Solina & Bajcsy, 1990; Jaklic et al., 2000) and later to point clouds (Liu et al., 2022). More recently, feed-forward networks have enabled accurate decomposition in a single pass (Paschalidou et al., 2019; Fedele et al., 2025). These methods, however, primarily decompose the observed geometry. Early work also explored predicting superquadrics from a single image (Paschalidou et al., 2019), but is limited to category-specific models trained on isolated synthetic objects and produces deterministic predictions in a canonical frame. In contrast, downstream applications require pixel-aligned, multi-object decompositions of complete 3D scenes: a grasp planned from only the visible side of a mug, for example, may collide with its unobserved geometry. This is particularly challenging from a single real-world image, where objects are often occluded or truncated by the image boundary. We address this by learning a 3D prior over complete superquadric decompositions, a task we term *generative scene decomposition*: predicting a complete superquadric decomposition of every object in a scene from a *single* image.

Two routes lead from a partial view to a complete decomposition. The first completes and then decomposes: 3D object generators (Chen et al., 2026; Siddiqui et al., 2026) synthesize a dense mesh before fitting superquadrics, yielding a costly two-stage pipeline with an unnecessary dense intermediate. The second completes while decomposing: SuperFlex (Tavernini et al., 2026) directly regresses complete decompositions from partial input. However, because it is deterministic, it cannot represent the ambiguity of the unobserved geometry and instead averages over plausible completions, often producing invalid shapes (see Fig. 3, 3rd row).

In PIXELDEC, we treat a partial observation as defining a distribution over complete decompositions rather than a single deterministic solution, and learn this distribution directly. Each object is represented by at most 16 superquadrics with 22 parameters each. Instead of generating a dense 3D intermediate, such as a sparse voxel grid (Xiang et al., 2025; Chen et al., 2026) or a set of latent vectors (Zhang et al., 2024; Siddiqui et al., 2026), we apply flow matching directly in superquadric parameter space. Sampling takes four Euler steps, with all scene objects processed in one batched pass in under a second. On ASE and TableVerse, PIXELDEC outperforms generate-then-fit pipelines based on SAM3D and ShapeR in both IoU and Chamfer distance, while requiring only 0.07,s per object compared to 21.6,s and avoiding a 2.2,MB intermediate mesh. Its primitives also outperform the dense ShapeR and SAM3D meshes, especially for objects truncated by the image boundary.

In summary, our contributions are:

- **Generative scene decomposition.** We formulate the task from a generative perspective learning a distribution over complete decompositions rather than regressing a single one. On ShapeNet, it lowers P-FID (Nichol et al., 2022) by an order of magnitude compared with the deterministic regressor which learns from the same data.
- **Primitives as the generative space.** We show that a few hundred primitive parameters are enough to serve as the state space of a flow model. A DiT that flows directly on superquadric parameters produces complete objects in only four Euler steps, two orders of magnitude faster than generating a mesh and fitting primitives to it, with an order of magnitude less model memory.
- **Actionable scene abstractions.** We show that our decompositions can be used as they are: in simulated table decluttering they raise the grasp success rate from 56.8% to 71.1% over SuperFlex primitives and allow a VLM to plan interactions with the physical world.

2 RELATED WORKS

Primitive-based Object Decomposition. *Optimization-based methods.* Decomposing 3D shapes into a small set of primitives is a long-standing problem, traditionally solved with classical fitting algorithms (Solina & Bajcsy, 1990; Jaklic et al., 2000). These methods are sensitive to noise and outliers and typically recover one primitive at a time. More recent optimization-based approaches improve robustness and reconstruction quality: EMS (Liu et al., 2022) introduces a probabilistic formulation for robust point-cloud fitting, while Marching-Primitives (Liu et al., 2023), Light-SQ (Wang et al., 2025b), and SuperFrusta (Ganeshan et al., 2026) improve multi-primitive reconstruction accuracy. However, they still assume access to the full volumetric extent of the shape, which is rarely available in real-world captures. Further, optimization-based methods fit each shape from scratch, which is slow and prevents them from learning shape priors that transfer across objects. Learning-based methods usually train a neural network with self-supervised reconstruction objectives and predict primitive parameters in a single forward pass. Early approaches (Tulsiani et al., 2017; Paschali-

dou et al., 2019; Deng et al., 2020; Paschalidou et al., 2021; Yang & Chen, 2021) mainly use primitives for shape *abstraction*, parsing objects of a given category, such as chairs, airplanes or human bodies (Paschalidou et al., 2019; 2020), into a few interpretable parts. SuperDec (Fedele et al., 2025) shifts the focus from *abstraction* to *reconstruction*: by refining primitive queries from local point features, it learns class-agnostic priors that let a model trained only on ShapeNet decompose real-world objects, making superquadrics a practical representation for downstream tasks (Fedele et al., 2026; Sella et al., 2026). Building on this architecture, SuperFlex (Tavernini et al., 2026) increases the expressiveness of each primitive with bending and tapering deformations and improves reconstruction accuracy with a volumetric loss, exceeding the accuracy of optimization-based methods at a fraction of their runtime. Since they are trained to explain their input, however, all these methods decompose only the geometry they observe. Closest to our work, the supervised variant of SuperFlex (Tavernini et al., 2026) also learns to output complete superquadrics from partial point clouds. However, it does so deterministically: by regressing a single decomposition, it averages over the ambiguities of the partial input instead of modeling them (Fig. 3). PIXELDEC shifts focus from reconstruction to generation, modeling this ambiguity as a distribution over complete decompositions learned directly in superquadric parameter space.

Single-Image 3D Object Generation. Recent image-to-3D models reconstruct complete, pixel-aligned objects from a single view by training large flow-matching transformers over 3D latent spaces, either sparse voxel grids (Xiang et al., 2025; Chen et al., 2026) or sets of latent vectors (Zhang et al., 2024; Zhao et al., 2025; Siddiqui et al., 2026), and generate dense, highly detailed meshes. Among them, SAM3D (Chen et al., 2026) reconstructs objects from a masked image and regresses their pose from the image, while ShapeR (Siddiqui et al., 2026) additionally conditions on the observed point cloud, producing reconstructions directly in the frame of the input. Generating such dense geometry, however, comes at a cost that limits their use in two ways. First, generation runs over thousands of latent tokens, whose output must be decoded into an implicit field and meshed by iso-surface extraction: a single object takes several seconds and gigabytes of memory, and a cluttered scene takes minutes, which rules out interactive and real-time applications. Second, the resulting dense meshes are hard to reason about and interact with. Primitives can still be fitted to them afterwards (Tavernini et al., 2026), but the resulting two-stage pipeline is not trained end-to-end and spends most of its compute on a dense intermediate that is then discarded. PIXELDEC instead generates the primitive parameters directly, decomposing all objects of a scene in a single batched pass in under a second, and outputs a lightweight representation that is directly actionable by downstream planning and editing tools.

Superquadrics as Robot Control Interface. Superquadrics have a long history in robotics, as their compact and analytically tractable description of object geometry makes them a convenient interface between perception and control. They have been used to synthesize grasp poses for isolated, partially observed, and large objects (Vezzani et al., 2017; Wu et al., 2023; Tu & Desingh, 2025), for fast collision checking (Ruan et al., 2022), and as differentiable constraint sets in safety filters (Brunke et al., 2025; Zhao et al., 2026b). Superquadric scene models are also cheap enough to run inside model predictive control for mechanical search in clutter (Kim et al., 2023), and enable manipulation of transparent objects that depth sensors cannot capture (Kim et al., 2024). In all of these systems, however, the bottleneck is perception rather than control: superquadrics are fitted to depth or multi-view point clouds and thus cover only observed geometry, leaving planners to reason about incomplete objects. PIXELDEC closes this gap by predicting complete scene-level superquadrics from a single RGB image, directly consumable by any of the above methods.

3 METHOD

We perform pixel-aligned 3D scene decomposition into superquadrics from a single image. At object level (Sec. 3.1–3.4), our model takes segmented images and pixel-aligned point maps as input. We then scale to full scenes (Sec. 3.5) via off-the-shelf instance segmentation (Carion et al., 2025) and point map prediction (Wang et al., 2025a).

3.1 SUPERQUADRICS GENERATION WITH FLOW MATCHING

Flow Matching. We adopt the flow matching formulation to model object-level superquadrics generation. Flow matching (Lipman et al., 2022) defines a probability path between an easy-to-sample distribution p_0 and a data distribution p_1. This path is described by an interpolant $\mathbf{x}_t$ which

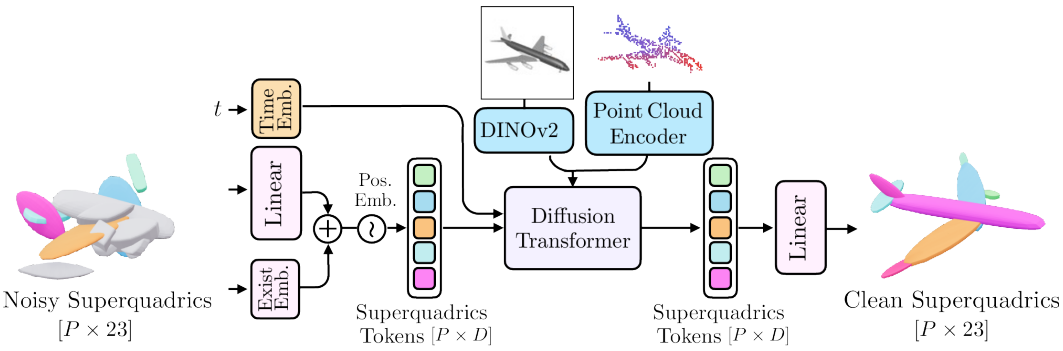

Fig. 2: **Model Illustration.** PIXELDEC performs flow matching directly over superquadric parameters. At timestep t, noisy superquadrics are processed by a diffusion transformer conditioned on DINOv2 image features and geometric features from a pixel-aligned point map, producing a clean superquadric decomposition.

couples a data sample $\mathbf{x}_1$ with its noise sample $\mathbf{x}_0$, given a time schedule $t \in [0, 1]$. As a result, the velocity of the probability path is given as: $\mathbf{v} = \mathbf{x}_1 - \mathbf{x}_0$. A flow model F_Θ is usually trained to approximate the flow velocity $\mathbf{v}$ given $\mathbf{x}_t$, time t, and conditioning $\mathbf{c}$. After training, generation is performed by solving the ODE: $\frac{d\mathbf{x}_t}{dt} = F_\Theta(\mathbf{x}_t, t)$, with $\mathbf{x}_0 \sim p_0$ from $t = 0$ to $t = 1$.

Flow Matching on Superquadric Parameters. Superquadrics are a compact yet expressive family of geometric primitives which can represent a wide variety of 3D shapes by only using a set of 22 continuous parameters. Following SuperFlex (Tavernini et al., 2026), we allow a maximum of $P = 16$ superquadrics slots to model each object and use the compact set of their geometric parameters $\mathbf{x}_c = [\mathbf{t}, R, \mathbf{s}, \boldsymbol{\varepsilon}, \boldsymbol{\tau}, \boldsymbol{\beta}_k, \boldsymbol{\beta}_\alpha]$ to describe each slot, together with a binary discrete value $\mathbf{x}_d \in \{0, 1\}$ which allows to model a variable number of active primitives per object. We refer to (Tavernini et al., 2026) for details on superquadric parameters. In our model, we exploit the compactness of this parameterization to perform flow matching directly on the space of superquadric parameters, modeling the continuous parameters with *continuous* flow matching and the binary existence value with *discrete* flow matching (Campbell et al., 2024; Gat et al., 2024). Given the different nature of those parameters, for each of them we define ad-hoc normalizations, prior p_0 distributions, and interpolations between prior and clean samples. We summarize our design choices in Sup. Mat.

3.2 ARCHITECTURE

We implement F_Θ as a Diffusion Transformer (DiT) which predicts an ordered set of $P = 16$ superquadric primitives per object and represents each primitive with a token. The network takes as input the noisy ordered set of superquadric parameters $\mathbf{x}_t$, the current time step t together with the conditioning $\mathbf{c}$ and predicts the clean superquadric parameters $\mathbf{x}_1$, which can be rendered based on their parameters and existence values. We show an overview of our architecture in Fig. 2 and provide more details in the next paragraphs.

Superquadric Parameters Encoding. Noisy superquadrics parameters $\mathbf{x}_t \in \mathbb{R}^{P\times 23}$ are fed into the DiT as superquadric tokens $\mathbf{T}_{\text{in}} \in \mathbb{R}^{P\times D}$ computed as:

$$\mathbf{T}_{in} = \text{Linear}(\mathbf{x}_{t,c}) + \text{Embedder}_e(\mathbf{x}_{t,d}) + \text{PE}\,, \tag{1}$$

where $\mathbf{x}_{t,c} \in \mathbb{R}^{P\times 22}$ are the continuous superquadric parameters, and $\mathbf{x}_{t,d}$ is one of the three categorical values (MASK, not-exist, exist) used to denote existence state of slots. Embedder_e is an embedding layer which maps each possible state to a D-dimensional vector and PE are fixed sinusoidal positional embeddings (Tancik et al., 2020) for each of the P superquadric slots.

Conditioned Flow Matching. The conditioning $\mathbf{c}$ is an image together with its pixel-aligned point map, where each point contains the 3D position of the pixel it came from. We encode points with a trainable PVCNN (Liu et al., 2019), pixels with a frozen DINOv2 (Oquab et al., 2024). Then, we concatenate them along the channel dimension and inject them into the DiT via cross-attention.

Clean Superquadric Prediction. A linear head projects each output token $\mathbf{T}_{\text{out}} \in \mathbb{R}^{P\times D}$ back to the superquadric parameters $\mathbf{x}_1 \in \mathbb{R}^{P\times 23}$.

3.3 TRAINING

Teacher supervision. In order to supervise our generative model, we adopt a distillation setup: our targets are produced by a frozen SuperFlex (Tavernini et al., 2026) teacher run on the *complete* object cloud, whereas PIXELDEC is a student which learns to recover the same output conditioned only on a *partial* observation. The teacher emits P primitives in a fixed slot order, and we supervise each of the student's P slots against the teacher primitive of the same index. Slot identity is thus inherited from the teacher and made explicit to the network through the fixed positional encoding.

Training Losses. Although F_Θ predicts the clean parameters $\hat{\mathbf{x}}_1 = F_\Theta(\mathbf{x}_t, t, \mathbf{c})$, we optimize the continuous parameters in velocity space (v-loss), following JiT (Li & He, 2026). We convert $\hat{\mathbf{x}}_1$ into an estimated velocity $\hat{\mathbf{v}}_t$ by rearranging the interpolation path. We then sum the weighted squared velocity errors over all parameter groups,

$$\mathcal{L}_{\text{geom}} = \mathbb{E}_{t,\mathbf{x}_0,\mathbf{x}_1}\left[\sum_{g\in\mathcal{G}} w_g \left\|\hat{\mathbf{v}}_t^g - \mathbf{v}_t^g\right\|^2\right], \tag{2}$$

where $\mathcal{G}$ ranges over all parameter groups and the w_g are per-group weights. We supervise the parameters of both active and non-active slots identically and, as shown in Tab. 3, this is fundamental. In addition to the continuous parameters, we supervise the posterior with a binary cross-entropy against $\mathbf{x}_d$ and define $\mathcal{L}_{\text{exist}} = \mathbb{E}_{t,\mathbf{x}_0,\mathbf{x}_1}\left[\frac{1}{P}\sum_{i=1}^{P} \text{BCE}\big(\sigma(\ell_i),\, \mathbf{x}_{d,i}\big)\right]$. The total training objective is defined as $\mathcal{L} = \mathcal{L}_{\text{geom}} + \lambda_e\, \mathcal{L}_{\text{exist}}$.

3.4 INFERENCE

At inference we integrate the geometry and existence flows jointly over a shared schedule of N evenly spaced steps. Geometry starts from a source draw $\mathbf{x}_{0,\mathbf{c}} \sim p_0$ and existence from an all-MASK state. At each step t_i the DiT reads the current geometry $\mathbf{x}_{t_i}$ and existence state and predicts the clean geometry $\hat{\mathbf{x}}_1$ and the existence posterior. At each step of denoising we update continuous parameters with an Euler step of the induced velocity and existence value with the mask-path sampler as in standard discrete flow matching.

3.5 GENERATIVE SCENE DECOMPOSITIONS FROM A SINGLE IMAGE

To generate superquadric decompositions of entire scenes from a single image, we first predict the pixel-aligned point maps with MoGe2 (Wang et al., 2025a) and 2D instance segmentation masks with SAM3 (Carion et al., 2025). The original image and point map are cropped around each predicted instance mask, the background of the crop is masked with black pixels, and 3D points of each instance are mean-centered and scaled into the unit cube $[-0.5, 0.5]^3$. Object conditionings are then batched, and efficiently processed by our object-level method, obtaining per instance predictions. Undoing the normalizations places the predictions in the common pixel-aligned input frame.

4 EXPERIMENTS

4.1 COMPARING WITH STATE-OF-THE-ART METHODS

Datasets and Metrics. We evaluate on three datasets. *ShapeNet* (Chang et al., 2015) is a large object-level dataset used for comparison with superquadric decomposition baselines. We follow the 13-class split of Choy et al. (2016), rendering depth maps from the provided meshes and back-projecting them into partial point clouds. *Aria Synthetic Environments (ASE)* is a synthetic scene-level dataset of furnished indoor scenes with photorealistic RGB-D renders and instance segmentation, populated with assets from ABO (Collins et al., 2022). Objects are partially observed due to occlusions and truncations; we evaluate on 300 views from 220 scenes. *TableVerse* (Wang et al., 2026) is a synthetic scene-level dataset of tabletops with everyday objects, whose 3D assets are generated via SAM3D (Chen et al., 2026) from real-world imagery; we evaluate on 350 views from 276 scenes across 187 object classes. We evaluate reconstruction accuracy (IoU, Chamfer distance, distance from observed points), generation quality (P-FID, P-KD), runtime and memory.

Methods in Comparison. We compare against deterministic and generative methods. Among *deterministic* approaches, EMS (Liu et al., 2022) fits superquadrics hierarchically via optimization; SuperDec (Fedele et al., 2025) and SuperFlex (Tavernini et al., 2026) are feed-forward methods

Method	IoU↑	L1↓	L2↓	P-FID↓	P-KID↓	# Prim.
EMS (Liu et al., 2022)	0.206	6.63	1.56	59.2	176.8	2.97
SuperDec (Fedele et al., 2025)	0.221	3.42	0.36	39.1	105.9	4.73
SuperFlex (Tavernini et al., 2026)	0.463	2.34	0.15	12.2	28.6	4.91
PIXELDEC (Ours, points)	0.603	1.70	0.08	0.88	0.13	5.58
PIXELDEC (Ours, points + image)	**0.622**	**1.62**	**0.06**	**0.86**	**0.06**	5.59

Tab. 1: **Quantitative Results on Partial ShapeNet**. Input point clouds are partial, single-view observations. Scores are averaged over all 13 categories. L1 and L2 are scaled by 10^2.

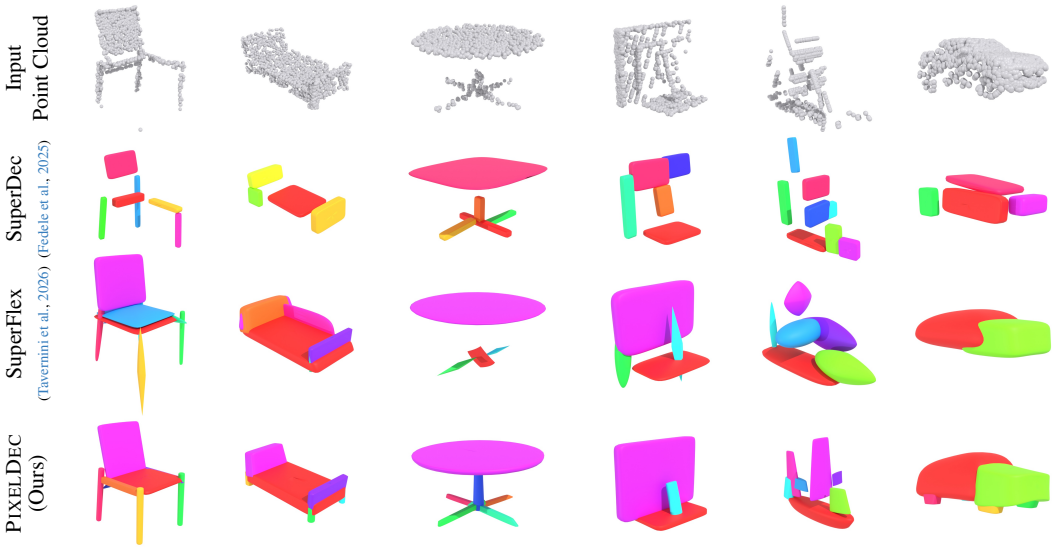

Fig. 3: **Qualitative Results on Partial ShapeNet.** SuperDec reconstructs only the observed geometry, while SuperFlex deterministically regresses a complete shape but averages modes of possible completions. PIXELDEC instead models a distribution over complete decompositions and generates plausible completions of the unobserved shape. Different colors indicate different primitives.

for undeformed and deformable decomposition respectively, with SuperFlex additionally handling point cloud completion. For *generative* baselines, we combine SAM3D (Chen et al., 2026) and ShapeR (Siddiqui et al., 2026) with SuperFlex: each reconstructs a complete 3D object from an image, which SuperFlex then decomposes into superquadrics.

Training Details. We train our model on ShapeNet for 1000 epochs with AdamW (batch 32, one-cycle schedule peaking at $2 \cdot 10^{-4}$, weight decay 0.01). We then fine-tune on ASE for 200 epochs (batch 64) and on TableVerse for 30 epochs (batch 32), both peaking at $5 \cdot 10^{-5}$.

Results on ShapeNet. We compare PIXELDEC with traditional baselines for superquadrics decomposition on ShapeNet, both quantitatively (Tab. 1) and qualitatively (Fig. 3). Note that, since all the baselines consider only a point cloud as input, for fairness of comparison we also report scores of our model when only conditioned on points. From Tab. 1 we observe that our model strongly outperforms previous decomposition methods in terms of reconstruction and quality of generations. The same result is proven qualitatively in Fig. 3 where we can also observe the failure modes of each method. *EMS* has no learning priors and is not able to generate the full shapes of partially observed objects. *SuperDec*, is trained to unsupervisely decompose full point clouds and thus only reconstruct the seen part of the object. *SuperFlex* for our same task, but does not use any generative prior - completing the missing parts with out of distribution pointy shapes (see yellow leg of the chair, legs of the table, vertical parts of the monitor), which are likely symptoms of mode averaging. Finally, our PIXELDEC is able to use the generative formulation to output clean superquadric shapes, despite the partial input.

Results on ASE and TableVerse. A possible way to obtain superquadrics from images would be running SoTA 3D object reconstruction models and then our teacher SuperFlex on top of their predictions. In these experiments, we show not only that our PIXELDEC is more efficient by several

Method	Rep.	Size↓ (KB)	IoU↑		L1↓		P→S↓		P-FID↓		P-KID↓		Mem_m (GB)	Mem_s (GB)	Time (s)
			ASE	TV	ASE	TV	ASE	TV	ASE	TV	ASE	TV			
ShapeR	Mesh	2,220	0.466	0.516	8.48	3.07	1.11	1.20	–	–	–	–	2.9	0.56	21.6
SAM3D†	Mesh	10,530	0.290	0.371	10.30	3.62	1.87	1.72	–	–	–	–	12.8	6.07	9.2
ShapeR + SuperFlex	SQ	0.36	0.406	0.457	8.84	3.48	**1.75**	**1.88**	8.52	5.36	3.5	4.2	3.1	0.58	21.6
SAM3D† + SuperFlex	SQ	0.38	0.325	0.384	10.69	4.11	2.67	2.40	8.83	5.97	3.9	6.3	13.0	6.09	9.2
PIXELDEC (Ours)	SQ	**0.35**	**0.500**	**0.571**	**5.80**	**2.50**	2.55	1.89	**5.72**	**3.06**	**1.7**	**−0.8**	**0.25**	**0.08**	**0.07**

Tab. 2: **Quantitative Results on ASE and TableVerse.**

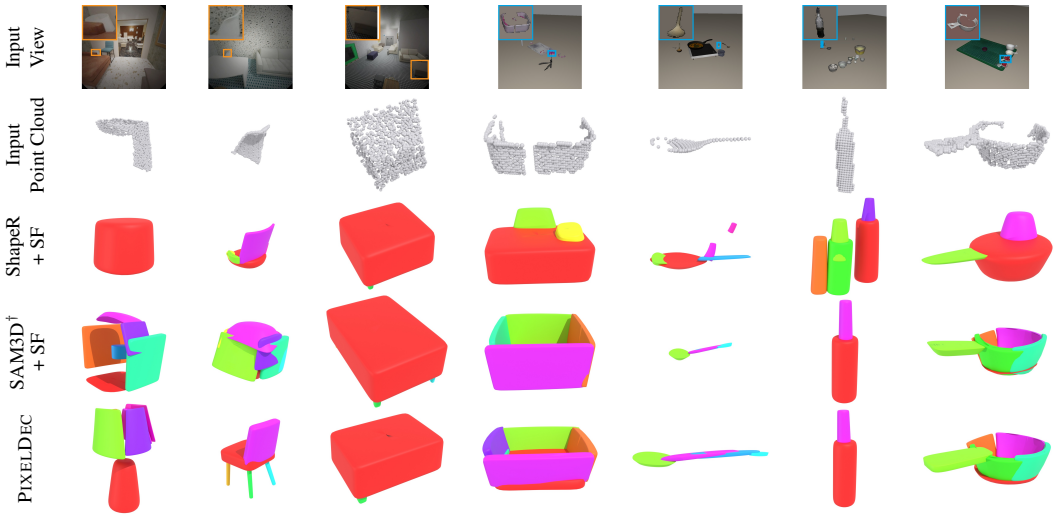

Fig. 4: **Qualitative Results on ASE and TableVerse.** *First row:* the input view, with the instance highlighted and the image crop, boxed in the colour of its dataset. *Second row:* observed depth points. *Below:* ShapeR and SAM3D reconstructed and then decomposed by SuperFlex (SF), and our PIXELDEC, which predicts the superquadrics directly. Different colors indicate different primitives.

orders of magnitude (Tab. 2), but also our final generations outperform these baselines, with particularly large gains when objects are severely truncated (see columns 1, 2, 3 of Fig. 4). We report quantitative (Tab. 2) and qualitative (Fig. 4) evaluation on objects from ASE and TableVerse.

Generalization to unseen domains. Although PIXELDEC is trained only on synthetic data, it generalizes to a wide range of unseen image domains. We show superquadric decompositions of cartoons (Fig. 1), of movie sets (Fig. 5), of robotic environments (Fig. 1, 9), and even of images generated by SD3 (Esser et al., 2024) (see Supp. Mat).

4.2 ANALYSIS EXPERIMENTS

Existence modeling. We tried three different approaches to model and supervise the existence token and evaluated them in Tab. 4. In *Continuous* we modeled the existence as a continuous parameter to generate and supervised it with velocity loss. In *Classifier* we modeled it as a parameter to deterministically, given the other geometric parameters. In *Discrete* we modeled it as a discrete parameter and supervised with discrete flow matching loss. We pick the *Discrete* approach as it outperforms both of the previous ones in terms of reconstruction quality and diversity of predictions.

Supervision of non-existing slots. As introduced in Sec. 3.3, both the teacher and the student outputs a fixed set of P superquadrics each of which can be either active or inactive. While we supervise all the *active* superquadrics, we need to define a strategy to treat the *inactive* ones. We experiment two main aspects: whether the inactive slots need to be supervised and whether teacher supervision is better than random. We report results in Tab. 3 and found that supervising all the slots is essential and using teacher supervision improves the results even further.

Number of inference steps. We experimented inference with different number of steps N. As shown in Tab. 5, performance saturates quickly with four Euler steps (62.2% mIoU) and degrades slightly beyond that. By default, we use 4 steps in the experiments.

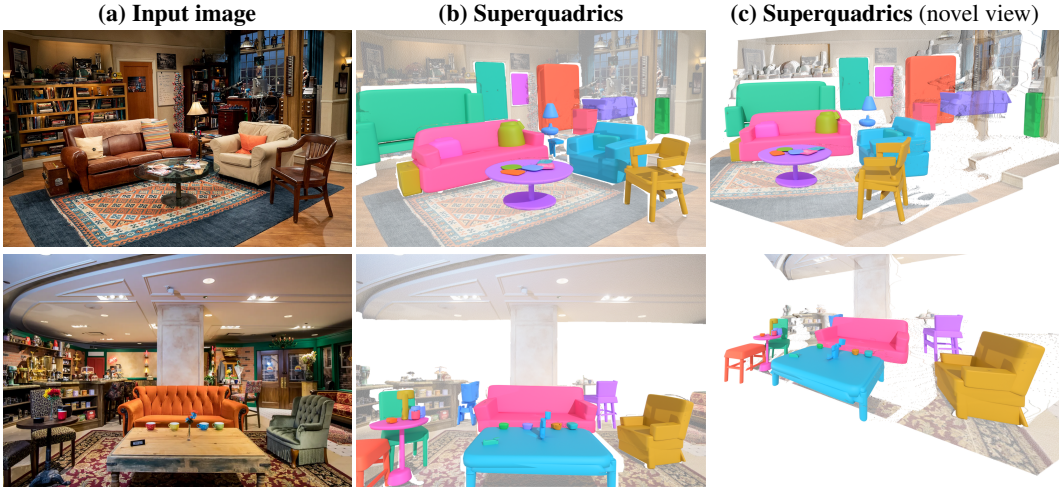

Fig. 5: **Decompositions of real indoor scenes.** We show (a) input image, generated superquadrics seen from (b) camera reference frame, (c) a novel viewpoint. We also visualize MoGe2 points from the background.

Padding	Sup.	IoU↑	L1↓	# Prim.
Teacher	✓	**61.9**	**1.62**	**5.60**
Random	✓	60.5	1.67	5.75
Teacher	✗	54.3	1.82	8.71
Random	✗	55.5	1.75	11.36

Tab. 3: **Padding of inactive slots.** Padding strategy and if non-active slots are supervised (Sup.).

Existence	IoU↑	Div.↑
Continuous	60.3	76%
Classifier	61.4	70%
Discrete (ours)	**61.5**	**80%**

Tab. 4: **Existence modeling** comparing reconstruction quality and diversity (Div.).

Steps	1	2	4	8	16
IoU↑	57.9	61.5	**62.2**	61.9	61.0
P-FID↓	2.01	1.01	0.86	0.80	**0.74**
P-KID↓	2.5	0.19	0.05	−0.12	−0.12

Tab. 5: **Number of inference steps.** IoU for reconstruction, P-FID and P-KID for generation quality.

Impact of Image Conditioning. While in Tab. 1 we have seen that DINOv2 features generally improve results, in Fig. 8 we inspect the impact of adding DINOv2 features on each ShapeNet category. We noticed that the image features help to disambiguate categories in which points are not enough to understand the semantic of the input object such as displays, telephones and rifles.

Generative completions Our PIXELDEC can sample distinct yet plausible superquadric generations given the same conditioning. In Fig. 6 we visualize different generations from our model.

What does our network learn? Superquadrics are not only a compact *output* space, but also a compact *representation* space in which our model learns to generate 3D objects, using one DiT slot for each superquadric. In Fig. 7 we visualize the self-attention activations of active superquadrics slots and observe that activation from parts that are semantically similar (e.g. legs of a table, wings of a plane) or dependent from each other (e.g. position and shape of a wings depends on the position of the body of the airplane) are generally high. This suggests the model is using this explicit representation space to learn symmetries and part correlations.

4.3 ACTIONABLE REPRESENTATION FOR ROBOTICS

A scene representation for robotics must be *actionable*: it should be explicit to support fast geometric queries that downstream modules rely on, and be easy to obtain and update to enable real-time closed-loop use. The superquadrics generated by PIXELDEC meet both the requirements and in this section we show how they can be employed to boost the performances of the SoTA grasping and trajectory algorithms and to provide an interpretable scene interface for VLMs and agents.

Grasp Generation and Trajectory Planning We show how the superquadrics generated from PIXELDEC can boost the performances of the SoTA grasping and trajectory algorithms evaluating them on the task of *table decluttering*, a well-known task in robotics which requires removing all the objects present in a table top and requires efficient grasping and path planning. We use scenes from TableVerse (Wang et al., 2026) to compare the performances when using observed point cloud (PC) or superquadrics from SuperFlex or PIXELDEC. We use GraspGen (Murali et al., 2026) (SoTA for grasping from partial pointclouds) or SuperQ-GRASP (Tu & Desingh, 2025) on top of each representation. We report results in Tab. 6. PIXELDEC achieves the highest declutter rate with both

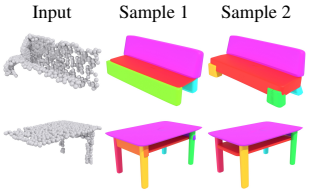

Fig. 6: **Generative completions.** For each input PIXELDEC generates different plausible completions.

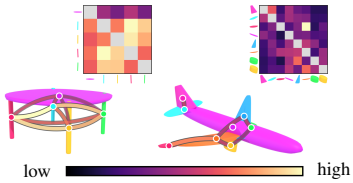

Fig. 7: **Self-attention between superquadric slots.** Patterns in SA matrices of superquadrics slots. Induced edges are colored by their strength.

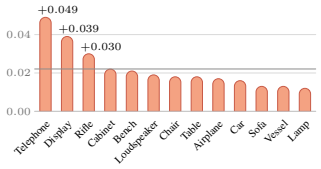

Fig. 8: **DINOv2 features.** IoU improvement given by additional DINOv2 features over each ShapeNet category.

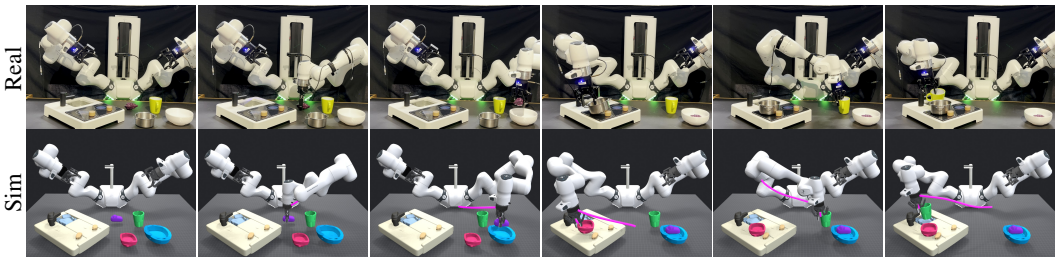

Fig. 9: **Language-guided kitchen cleanup.** Given a task instruction, a VLM agent uses the superquadrics from PIXELDEC and pick-and-place skills to place grapes in a bowl, a mug in the sink, and a pot on the stove.

samplers and raises GraspGen's grasp success rate by 14.7%, showing that generated superquadrics benefit learned grasping as well. SuperFlex, even if relying on the same underlying representation, brings no gain with the learned sampler (41.2%) and drops to 34.2% with the geometric one. We deploy the pipeline on a real dual-arm robot and report results in Supplementary Material and videos.

Long-Horizon Reasoning with Tool-Calling Agents While decluttering evaluates individual picks, long-horizon tasks also require reasoning about object semantics, interactions, and affordances. VLMs provide this high-level knowledge but require an interface to interact with the physical world. We build an agent based on Claude Opus 5.5, which perceives the scene through the compact superquadric representation produced by PIXELDEC and uses the same grasp and trajectory planners evaluated above to act in the real world. The agent decides in real time *what* to move and *where*, while grasps and trajectories are planned and validated in simulation; failures are reported back so the agent can revise its plan. After each action, the scene is re-observed and the representation is updated. From a single instruction, the agent completes a long-horizon kitchen clean-up task, placing food in a bowl, a mug in the sink, and a pot on the stove (Fig. 9); videos are in the appendix.

Representation	Grasping	DR↑	GSR↑	#Obst.↓
PC	Learned	41.8	56.4	2111
SQ-SuperFlex	Learned	41.2	56.8	48
SQ-PIXELDEC	Learned	<u>45.3</u>	**71.1**	**35**
SQ-SuperFlex	Geometric	34.2	47.5	48
SQ-PIXELDEC	Geometric	**45.7**	<u>62.3</u>	**35**

Tab. 6: **Table decluttering on TableVerse.** We compare using point cloud (PC) or superquadrics from SuperFlex or PIXELDEC, with a learned or geometric grasp sampler. DR: objects removed (%); GSR: grasp success rate (%); #Obst.: obstacles used by the planner. Best in bold, 2nd underlined.

5 CONCLUSION

We introduced PIXELDEC, a method for generating complete superquadric scene decompositions from a single image. By modeling partial observations as conditions on a distribution over complete shapes, PIXELDEC produces plausible completions of unobserved geometry with substantially better generation quality than a deterministic regressor. Operating directly in superquadric parameter space, it remains compact and efficient, decomposing full scenes in under a second while outperforming generate-then-fit pipelines. Finally, the resulting primitives are directly actionable for robotic tasks and provides VLMs with an interface to reason and act on the physical world.

AI USE STATEMENT

In this work, we used generative AI tools for different subtasks. For polishing writing by letting AI tools do a pass over grammar and suggest possible stylistic improvements. For retrieval and discovery to find related works which solved similar tasks in different domains as works in molecules generation which were doing flow matching over rotations. For research ideation and execution, specifically for writing code of visualizers and efficiently test novel ideas. We have not used generative AI tools for other tasks with required disclosure, and the rest of the required disclosure tasks are not applicable to this work. We have reviewed all AI-assisted work by reviewing and carefully testing the final code used for this work, reviewing the test, and all the suggested literature. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Alan H. Barr. Superquadrics and Angle-Preserving Transformations. *IEEE Computer Graphics and Applications*, 1981. 1

Lukas Brunke, Yanni Zhang, Ralf Römer, Jack Naimer, Nikola Staykov, Siqi Zhou, and Angela P. Schoellig. Semantically safe robot manipulation: From semantic scene understanding to motion safeguards. *IEEE Robotics and Automation Letters*, 2025. 1, 3

Andrew Campbell, Jason Yim, Regina Barzilay, Tom Rainforth, and Tommi Jaakkola. Generative Flows on Discrete State-Spaces: Enabling Multimodal Flows with Applications to Protein Co-Design. In *International Conference on Machine Learning (ICML)*, 2024. 4, 15

Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, Jie Lei, Tengyu Ma, Baishan Guo, Arpit Kalla, Markus Marks, Joseph Greer, Meng Wang, Peize Sun, Roman Rädle, Triantafyllos Afouras, Effrosyni Mavroudi, Katherine Xu, Tsung-Han Wu, Yu Zhou, Liliane Momeni, Rishi Hazra, Shuangrui Ding, Sagar Vaze, Francois Porcher, Feng Li, Siyuan Li, Aishwarya Kamath, Ho Kei Cheng, Piotr Dollár, Nikhila Ravi, Kate Saenko, Pengchuan Zhang, and Christoph Feichtenhofer. Sam 3: Segment anything with concepts. *arXiv preprint arXiv:2511.16719*, 2025. 3, 5

Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. ShapeNet: An Information-rich 3D Model Repository. *arXiv preprint arXiv:1512.03012*, 2015. 5

Xingyu Chen, Fu-Jen Chu, Pierre Gleize, Kevin J Liang, Alexander Sax, Hao Tang, Weiyao Wang, Michelle Guo, Thibaut Hardin, Xiang Li, et al. SAM 3D: 3Dfy anything in images. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026. 2, 3, 5, 6

Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European Conference on Computer Vision (ECCV)*, 2016. 5

Jasmine Collins, Shubham Goel, Kenan Deng, Achleshwar Luthra, Leon Xu, Erhan Gundogdu, Xi Zhang, Tomas F Yago Vicente, Thomas Dideriksen, Himanshu Arora, et al. ABO: Dataset and Benchmarks for Real-world 3D Object Understanding. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 5

Boyang Deng, Kyle Genova, Soroosh Yazdani, Sofien Bouaziz, Geoffrey Hinton, and Andrea Tagliasacchi. CvxNet: Learnable Convex Decomposition. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *International Conference on Machine Learning (ICML)*, 2024. 7, 15

Elisabetta Fedele, Boyang Sun, Leonidas Guibas, Marc Pollefeys, and Francis Engelmann. SuperDec: 3D Scene Decomposition with Superquadric Primitives. In *International Conference on Computer Vision (ICCV)*, 2025. 2, 3, 5, 6, 16

Elisabetta Fedele, Francis Engelmann, Ian Huang, Or Litany, Marc Pollefeys, and Leonidas Guibas. SpaceControl: Introducing Test-Time Spatial Control to 3D Generative Modeling. *International Conference on Learning Representations (ICLR)*, 2026. 3

Aditya Ganeshan, Matheus Gadelha, Thibault Groueix, Zhiqin Chen, Siddhartha Chaudhuri, Vladimir Kim, Wang Yifan, and Daniel Ritchie. Residual Primitive Fitting of 3D Shapes with SuperFrusta. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026. 2

Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky T. Q. Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. Discrete Flow Matching. In *International Conference on Neural Information Processing Systems (NeurIPS)*, 2024. 4, 15

Ales Jaklic, Ales Leonardis, and Franc Solina. *Segmentation and Recovery of Superquadrics*, volume 20. Springer Science & Business Media, 2000. 2

Seungyeon Kim, Young Hun Kim, Yonghyeon Lee, and Frank C. Park. Leveraging 3D reconstruction for mechanical search on cluttered shelves. In *Conference on Robot Learning (CoRL)*, volume 229 of *Proceedings of Machine Learning Research*, 2023. 1, 3

Young Hun Kim, Seungyeon Kim, Yonghyeon Lee, and Frank C. Park. T^2SQNet: A recognition model for manipulating partially observed transparent tableware objects. In *Conference on Robot Learning (CoRL)*, 2024. 3

Tianhong Li and Kaiming He. Back to basics: Let denoising generative models denoise. *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026. 5, 14

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022. 3

Weixiao Liu, Yuwei Wu, Sipu Ruan, and Gregory S Chirikjian. Robust and Accurate Superquadric Recovery: A Probabilistic Approach. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2, 5, 6, 15, 16

Weixiao Liu, Yuwei Wu, Sipu Ruan, and Gregory S Chirikjian. Marching-primitives: Shape abstraction from signed distance function. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8771–8780, 2023. 2

Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-voxel cnn for efficient 3d deep learning. *International Conference on Neural Information Processing Systems (NeurIPS)*, 2019. 4, 14

Adithyavairavan Murali, Balakumar Sundaralingam, Yu-Wei Chao, Jun Yamada, Wentao Yuan, Mark Carlson, Fabio Ramos, Stan Birchfield, Dieter Fox, and Clemens Eppner. GraspGen: A Diffusion-based Framework for 6-DOF Grasping with On-Generator Training. In *International Conference on Robotics and Automation (ICRA)*, 2026. 8, 18

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. 2

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. DINOv2: Learning Robust Visual Features without Supervision. *Transactions on Machine Learning Research Journal*, 2024. 4, 14

Despoina Paschalidou, Ali Osman Ulusoy, and Andreas Geiger. Superquadrics revisited: Learning 3d shape parsing beyond cuboids. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2, 3

Despoina Paschalidou, Luc Van Gool, and Andreas Geiger. Learning unsupervised hierarchical part decomposition of 3d objects from a single rgb image. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1060–1070, 2020. 3

Despoina Paschalidou, Angelos Katharopoulos, Andreas Geiger, and Sanja Fidler. Neural parts: Learning expressive 3d shape abstractions with invertible neural networks. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 3

Sipu Ruan, Xiaoli Wang, and Gregory S. Chirikjian. Collision detection for unions of convex bodies with smooth boundaries using closed-form contact space parameterization. *IEEE Robotics and Automation Letters*, 7(4):9485–9492, 2022. doi: 10.1109/LRA.2022.3190629. 3

Etai Sella, Hao Phung, Nitay Amiel, Or Litany, Or Patashnik, and Hadar Averbuch-Elor. Prox-e: Fine-grained 3d shape editing via primitive-based abstractions. *ACM SIGGRAPH 2026 Conference Papers*, 2026. 1, 3

Yawar Siddiqui, Duncan Frost, Samir Aroudj, Armen Avetisyan, Henry Howard-Jenkins, Daniel DeTone, Pierre Moulon, Qirui Wu, Zhengqin Li, Julian Straub, et al. Shaper: Robust conditional 3d shape generation from casual captures. *arXiv preprint arXiv:2601.11514*, 2026. 2, 3, 6

Franc Solina and Ruzena Bajcsy. Recovery of parametric models from range images: The case for superquadrics with global deformations. *IEEE transactions on pattern analysis and machine intelligence*, 1990. 2

Balakumar Sundaralingam, Siva Kumar Sastry Hari, Adam Fishman, Caelan Garrett, Karl Van Wyk, Valts Blukis, Alexander Millane, Helen Oleynikova, Ankur Handa, Fabio Ramos, et al. curobo: Parallelized collision-free minimum-jerk robot motion generation. *arXiv preprint arXiv:2310.17274*, 2023. 16

Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *International Conference on Neural Information Processing Systems (NeurIPS)*, 2020. 4

Gabriel Tavernini, Elisabetta Fedele, Tiago Novello, Leonidas Guibas, Marc Pollefeys, and Francis Engelmann. SuperFlex: Deformable Superquadrics for Point Cloud Decomposition. In *European Conference on Computer Vision (ECCV)*, 2026. 2, 3, 4, 5, 6, 14, 16

Xun Tu and Karthik Desingh. SuperQ-GRASP: Superquadrics-based grasp pose estimation on larger objects for mobile-manipulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2025. arXiv:2411.04386. 1, 3, 8, 18

Shubham Tulsiani, Hao Su, Leonidas J Guibas, Alexei A Efros, and Jitendra Malik. Learning shape abstractions by assembling volumetric primitives. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

Giulia Vezzani, Ugo Pattacini, and Lorenzo Natale. A Grasping Approach Based on Superquadric Models. In *International Conference on Robotics and Automation (ICRA)*, 2017. 3

Boyuan Wang, Yue Zhang, Xutao Xue, Xueyu Song, and Yu Sun. Tableverse: A large-scale tabletop dataset with real-world grounded layouts for generalizable manipulation. *arXiv preprint arXiv:2607.21017*, 2026. 5, 8, 16

Ruicheng Wang, Sicheng Xu, Yue Dong, Yu Deng, Jianfeng Xiang, Zelong Lv, Guangzhong Sun, Xin Tong, and Jiaolong Yang. Moge-2: Accurate monocular geometry with metric scale and sharp details. In *International Conference on Neural Information Processing Systems (NeurIPS)*, 2025a. 3, 5

Yuhan Wang, Weikai Chen, Zeyu Hu, Runze Zhang, Yingda Yin, Ruoyu Wu, Keyang Luo, Shengju Qian, Yiyan Ma, Hongyi Li, et al. Light-SQ: Structure-aware Shape Abstraction with Superquadrics for Generated Meshes. In *ACM SIGGRAPH Asia 2025 Conference Papers*, 2025b. 2

Yuwei Wu, Weixiao Liu, Zhiyang Liu, and Gregory S. Chirikjian. Learning-free grasping of unknown objects using hidden superquadrics. In *Robotics: Science and Systems (RSS)*, 2023. 3

Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. Structured 3D Latents for Scalable and Versatile 3D Generation. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. 2, 3

Yifeng Xu, Fan Zhu, Ye Li, Sebastian Ren, Xiaonan Huang, and Yuhao Chen. RGB-SQ Grasp: Inferring local superquadric primitives from single RGB image for graspability-aware bin picking. *arXiv preprint arXiv:2503.02387*, 2025. 1

Kaizhi Yang and Xuejin Chen. Unsupervised learning for cuboid shape abstraction via joint segmentation from point clouds. *ACM Transactions On Graphics (TOG)*, 2021. 3

Jason Yim, Andrew Campbell, Andrew YK Foong, Michael Gastegger, José Jiménez-Luna, Sarah Lewis, Victor Garcia Satorras, Bastiaan S Veeling, Regina Barzilay, Tommi Jaakkola, et al. Fast protein backbone generation with SE (3) flow matching. *arXiv preprint arXiv:2310.05297*, 2023. 14

Longwen Zhang, Ziyu Wang, Qixuan Zhang, Qiwei Qiu, Anqi Pang, Haoran Jiang, Wei Yang, Lan Xu, and Jingyi Yu. CLAY: A Controllable Large-scale Generative Model for Creating High-quality 3D Assets. *ACM Transactions on Graphics (TOG)*, 2024. 2, 3

Haocheng Zhao, Lukas Brunke, Oliver Lagerquist, Siqi Zhou, and Angela P Schoellig. Sq-cbf: Signed distance functions for numerically stable superquadric-based safety filtering. *arXiv preprint arXiv:2602.11049*, 2026a. 1

Haocheng Zhao, Lukas Brunke, Oliver Lagerquist, Siqi Zhou, and Angela P. Schoellig. SQ-CBF: Signed distance functions for numerically stable superquadric-based safety filtering. *arXiv preprint arXiv:2602.11049*, 2026b. 3

Zibo Zhao, Zeqiang Lai, Qingxiang Lin, Yunfei Zhao, Haolin Liu, Shuhui Yang, Yifei Feng, Mingxin Yang, Sheng Zhang, Xianghui Yang, et al. Hunyuan3d 2.0: Scaling Diffusion Models for High Resolution Textured 3D Assets Generation. *arXiv preprint arXiv:2501.12202*, 2025. 3

Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 14

A ADDITIONAL DETAILS

Implementation Details. Our DiT has 12 blocks, $D = 384$, 6 heads, and operates on $P{=}16$ superquadric slots. The conditioning points are encoded with a PVCNN point encoder (Liu et al., 2019), initialized from the SuperFlex (Tavernini et al., 2026) point-cloud reconstruction weights and fine-tuned jointly with the DiT. The rotation is presented to the network through the continuous 6D representation of Zhou et al. (2019). DINOv2 (Oquab et al., 2024) features use ViT-S/14 on 224×224 images, and we keep it frozen during training. Each object is conditioned on 1024 points and supervised against the predictions of SuperFlex (Tavernini et al., 2026) on a complete point cloud. During training we sample $t \sim \mathcal{U}(0, 1)$. We set $\lambda_e{=}1$ and use per-group weights w_g of 1 for translation, scale and shape exponents, 0.5 for the 6D rotation, 0.7 for tapering and bending scales, and 0.2 for the bending angles, lowered to 0.05 when fine-tuning on ASE.

Time Step Encoding. As in a standard DiT, the current time step t is mapped by a sinusoidal embedder to a vector $\mathbf{T}_t \in \mathbb{R}^D$, which conditions the transformer blocks through adaLN-Zero modulation of their self-attention and feed-forward layers.

Multimodal Flow Given the different nature of the superquadric parameters, for each of them we need to define ad-hoc normalizations, prior p_0 distributions, and interpolations between prior and clean samples. Below, we explain the design choices for each of the,.

Parameter	Normalization	$\mathbf{p_0}$	Interpolation
Translation $\mathbf{t}$	Mean and variance	$\mathcal{N}(\mathbf{0}, I)$	Linear
Rotation R	–	$\mathcal{U}(SO(3))$	$SO(3)$ geodesic
Scale $\mathbf{s}$	log, mean and variance	$\mathcal{N}(\mathbf{0}, I)$	Linear
Shape $\boldsymbol{\epsilon}$	Rescale to $[-1, 1]$	$\mathcal{N}(\mathbf{0}, I)$	Linear
Tapering $\boldsymbol{\tau}$	Rescale to $[-1, 1]$	$\mathcal{N}(\mathbf{0}, I)$	Linear
Bending scales $\boldsymbol{\beta}_k$	log, mean and variance	$\mathcal{N}(\mathbf{0}, I)$	Linear
Bending angles $\boldsymbol{\beta}_\alpha$	–	$\mathcal{U}(S^1)$	S^1 geodesic

Tab. 7: **Parameters normalization and interpolation.** Each superquadric parameter is treated according to its type. We report the normalization, initial noise distribution $\mathbf{p}_0$, and interpolation.

Supervision of Continuous Flow. As we have discussed in the main, our network F_Θ predicts the clean parameters $\hat{\mathbf{x}}_1 = F_\Theta(\mathbf{x}_t, t, \mathbf{c})$, we optimize the continuous parameters in velocity space (v-loss), following JiT (Li & He, 2026). We convert $\hat{\mathbf{x}}_1$ into an estimated velocity $\hat{\mathbf{v}}_t$ by rearranging the interpolation path. More specifically, for quantities in the Euclidean manifold, we compute $\hat{\mathbf{v}}_t$ as:

$$\hat{\mathbf{v}}_t(\mathbf{x}_t, t) = \frac{\hat{\mathbf{x}}_1 - \mathbf{x}_t}{1 - t}, \qquad \mathbf{v}_t(\mathbf{x}_t, t) = \mathbf{x}_1 - \mathbf{x}_0 . \tag{3}$$

For rotations, we follow (Yim et al., 2023) and we first convert the network's 6D output into a clean rotation matrix $\hat{R}_1$ via Gram-Schmidt, and then we compute the estimated and ground truth velocities in the tangent space as:

$$\hat{\mathbf{v}}_t^R = \frac{\log_{R_t}(\hat{R}_1)}{1 - t}, \qquad \mathbf{v}_t^R = \frac{\log_{R_t}(R_1)}{1 - t} . \tag{4}$$

For bending angles, we first compute the predicted angle $\hat{\mathbf{v}}^\alpha$ on S^1, and then we take the difference along the shorter arc:

$$\hat{\mathbf{v}}_t^\alpha = \frac{\log_{\alpha_t}(\hat{\alpha}_1)}{1 - t}, \qquad \mathbf{v}_t^\alpha = \frac{\log_{\alpha_t}(\alpha_1)}{1 - t} . \tag{5}$$

We then sum the weighted squared velocity errors over all parameter groups,

$$\mathcal{L}_{\text{geom}} = \mathbb{E}_{t, \mathbf{x}_0, \mathbf{x}_1} \left[\sum_{g \in \mathcal{G}} w_g \left\| \hat{\mathbf{v}}_t^g - \mathbf{v}_t^g \right\|^2 \right], \tag{6}$$

where $\mathcal{G}$ ranges over all parameter groups and the w_g are per-group weights. To avoid the $1/(1-t)$ factor blowing up near $t{=}1$ we clip it at a fixed threshold. Notice that the loss is computed over all P slots, including the inactive ones.

Supervision of Discrete Flow. Unlike the continuous parameters, which are noised by interpolation toward their prior, the discrete existence token is noised by *masking*, following masked discrete flow matching (Campbell et al., 2024; Gat et al., 2024). During training, we corrupt the clean existence state $\mathbf{x}_d \in \{0,1\}^P$ by revealing each slot's true state independently with probability $\kappa(t)$ and replacing it with MASK otherwise; we use the linear schedule $\kappa(t) = t$, so all slots are masked at $t{=}0$ and fully revealed at $t{=}1$. The DiT reads this partially revealed state and outputs, for every slot i, a posterior logit ℓ_i over the clean state $\{\text{exist}, \text{not-exist}\}$; no mass is ever placed on MASK, which appears only in the input. We supervise the posterior with a binary cross-entropy against $\mathbf{x}_d$ and define $\mathcal{L}_{\text{exist}}$ as:

$$\mathcal{L}_{\text{exist}} = \mathbb{E}_{t,\mathbf{x}_0,\mathbf{x}_1}\left[\frac{1}{P}\sum_{i=1}^{P}\text{BCE}\big(\sigma(\ell_i),\, \mathbf{x}_{d,i}\big)\right]. \tag{7}$$

Training objective. The total training objective is defined as:

$$\mathcal{L} = \mathcal{L}_{\text{geom}} + \lambda_e\, \mathcal{L}_{\text{exist}}\,. \tag{8}$$

B Additional Results

B.1 Interactive 3D Scene Generation

By obtaining our representation directly from images, we can use superquadrics as an agent- and user-friendly interface for 3D generation. We generate an image of a living room from a text prompt with Stable Diffusion 3.5 (Esser et al., 2024), and PixelDec decomposes every object in it into superquadrics (Fig. 10a,b). Since each primitive is described by a handful of interpretable parameters (position, orientation, size, shape and deformation), a VLM can read the scene as plain numbers and edit it by writing new ones. In Fig. 10 we ask Claude Opus 5.5 to *"move the left armchair to the right of the table so that it still faces the table"*, the agent identifies the parameters of the correct armchair, rotates and slides it towards the table, keeping it on the floor and inside the camera frame. Then, we also asked to *"add a lamp on the table top, and a cat tree with a cat where the chair was"*, it composes the three new objects from primitives it writes itself, placing each one on the surface that supports it (Fig. 10c). Edits of this kind are hard to express in pixel space or on dense meshes, but reduce to a few parameter changes on superquadrics.



Fig. 10: **Interactive 3D scene generation.** PixelDec decomposes a generated image (a) into superquadrics (b), which an agent can easily edit from language instructions (c). In (c), we visualize with colors moved or added objects, untouched ones are grey. Both the decompositions are visualized from a novel view to emphasize the generated geometry of unseen input parts.

B.2 Qualitative Results

Fig. 11 repeats Fig. 3 with the optimization-based EMS (Liu et al., 2022).

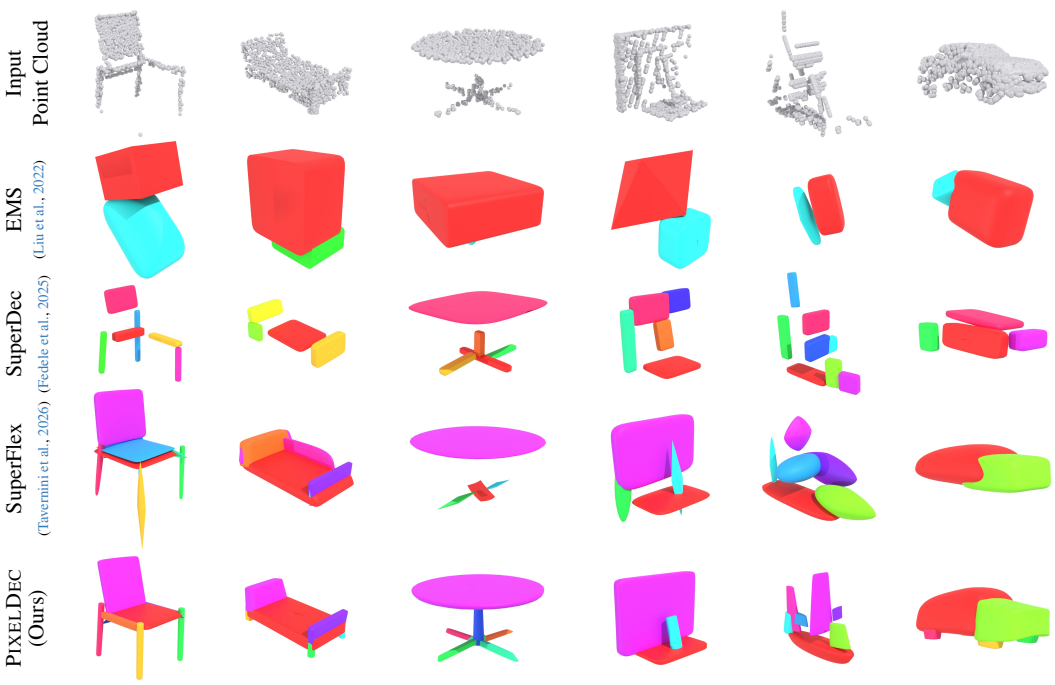

Fig. 11: **Qualitative Results on ShapeNet, including EMS.** *Top row:* input partial point clouds. *Below:* EMS, SuperDec, SuperFlex and our PIXELDEC. Different colors indicate different primitives.

C ROBOTICS EXPERIMENTS

Task and simulation setup. We evaluate on a dual-arm decluttering task that we build on the tabletop scenes of TableVerse (Wang et al., 2026). Each scene places 4–15 household objects (mean 7.0) on a table in front of a dual-arm robot with two Franka Research 3 arms, each carrying a Robotiq 2F-85 parallel-jaw gripper. We add the arms, the table and one bin per arm to the scene; the robot observes it from a single head-mounted RGB-D camera, receives ground-truth instance masks, and has to carry every object into the bin on its side. An episode proceeds in pick-and-place cycles: the scene is re-observed with fresh sensor noise, the representation is recomputed for the remaining objects, one grasp is selected and executed, and the cycle repeats until every object is removed, 3 consecutive attempts fail, or the number of attempts exceeds the object count by 2. Trajectories are planned with cuRobo Sundaralingam et al. (2023) and executed in MuJoCo 3.12.

Statistics. We use 50 scenes drawn from the held-out test list of TableVerse and run every configuration with three seeds. The scene and object placement are fixed by the dataset; a seed drives the camera's depth noise and the sampling in the grasp sampler, and cuRobo itself is not deterministic on the GPU, so the spread across seeds reflects run-to-run variation of the whole pipeline rather than variation of the scenes. We report the mean over seeds. The declutter rate (DR) is the fraction of objects that end up in the bin. The grasp success rate (SR) is the fraction of executed grasps that lift the object off the table, so it measures grasp quality independently of how many attempts an episode needs. Time is the wall-clock time of one prediction on the first frame of a scene, from the segmented points to the set of collision-checked grasps. The number of obstacles (# Obst.) is the size of the collision world that cuRobo checks against in that frame: the 11 static cuboids of the table and bins plus, for every object, either the occupied voxels of its point cloud at 1 cm resolution or the superquadrics of its decomposition together with a 1 cm voxel patch covering the observed points that no primitive comes within 1 cm of.

Representations. All methods receive the same segmented single-view point cloud, rendered at 320×240 with 2 mm of depth noise. *PC* uses this cloud directly. *SQ-SuperFlex* fits superquadrics to the partial cloud of each object with SuperFlex Tavernini et al. (2026), using the teachers checkpoint finetuned on tableverse. *SQ*-PIXELDEC generates the decomposition of each object from the single view with our model, drawing 10 samples with 8 flow-matching steps per object and keeping the

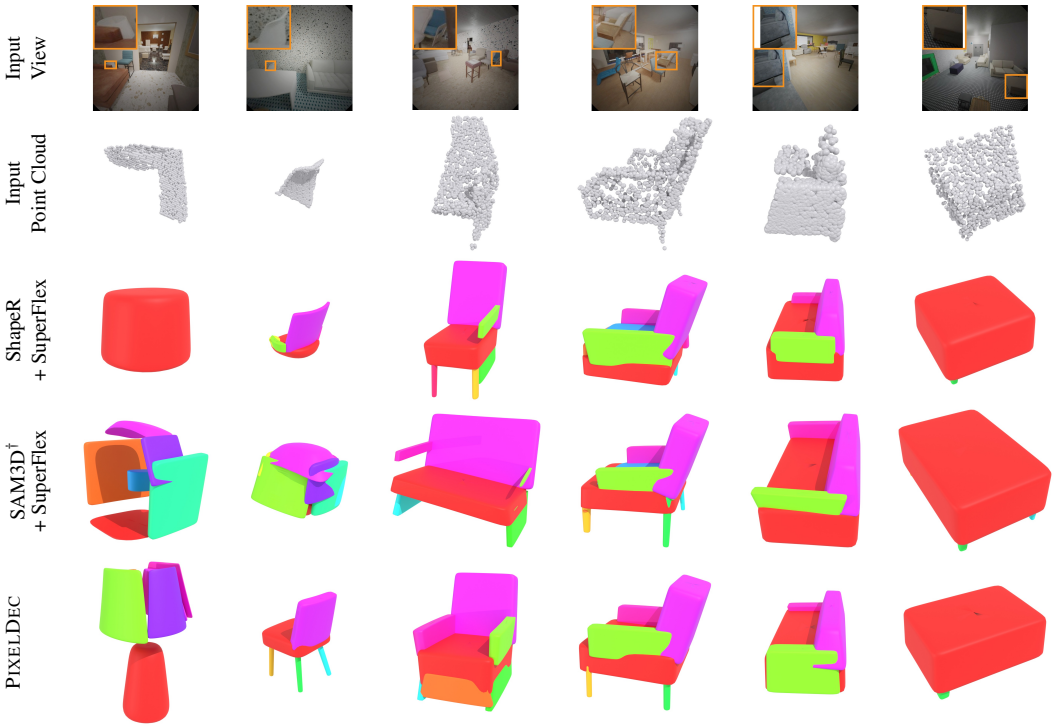

Fig. 12: **Qualitative Results on ASE.** *First row:* the input frame, with the instance boxed and the crop the model is conditioned on inset. *Second row:* the 1024 points every method receives. *Below:* the two reconstruct-then-abstract baselines – SAM3D and ShapeR, each abstracted by SuperFlex, SAM3D after the rigid fit (†) – and our PIXELDEC, which predicts the superquadrics directly from the points. Different colors indicate different primitives, and the three decompositions of a column are rendered from the frame's own viewpoint.

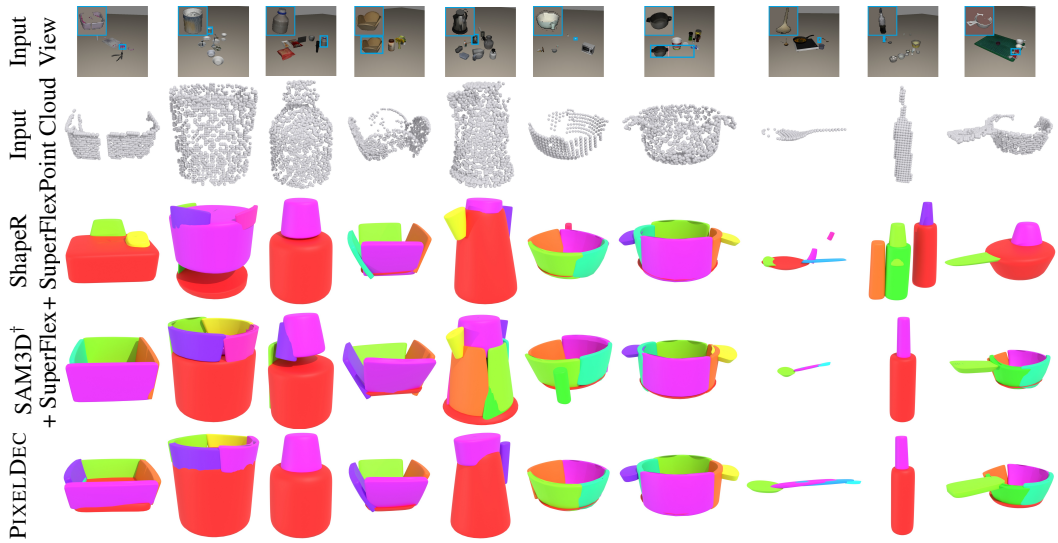

Fig. 13: **Qualitative Results on TableVerse.** *First row:* the input view, with the instance highlighted and the crop the model is conditioned on inset. *Second row:* the 1024 points every method receives. *Below:* ShapeR and SAM3D reconstructed and then abstracted by SuperFlex, SAM3D after the rigid fit (†), and our PIXELDEC. Different colors indicate different primitives.

highest-likelihood sample; it is additionally conditioned on a 640×480 RGB render of the same view. In both superquadric variants the planner treats the primitives, plus the patch above, as the collision geometry of the object.

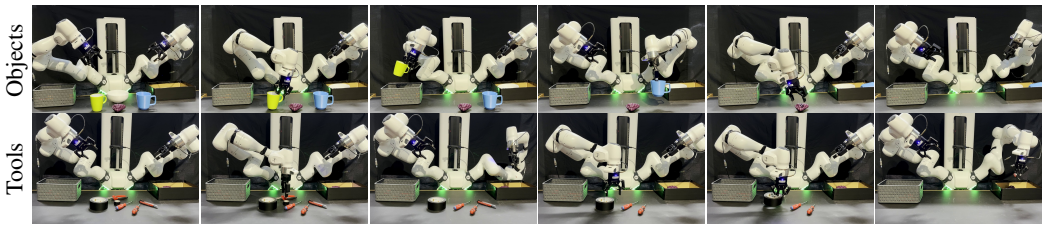

Fig. 14: **Real-world decluttering rollouts.** Each row shows one rollout from left to right. *Top:* General objects are picked and placed into the boxs: initial scene, grasping the yellow mug, placing it into the crate, grasping the blue mug, grasping the grapes, and the cleared table. *Bottom:* tools are cleared: initial scene, grasping a screwdriver, placing into the box, grasping the tape, lifting it into the box, and clearing the last screwdriver.

Surface point clouds for the learned sampler. GraspGen Murali et al. (2026) expects a point cloud of the object. For the point-cloud baseline this is the observed cloud. For the superquadric representations we sample points on the surface of every primitive, so that the learned sampler sees the completed object rather than its visible side. Each primitive is sampled at a spacing of 5 mm, the observed points are kept alongside the sampled ones, and the union is subsampled to the 8192 points GraspGen consumes. GraspGen then draws 200 grasps per object and returns the best 100; the released checkpoint is used unchanged.

Geometric sampler. Our geometric sampler builds on SuperQ-GRASP Tu & Desingh (2025), which places antipodal grasps on the principal axes of each superquadric. We make three changes so that the sampler exploits complete primitives. First, grasps are proposed on every side of a primitive whose width fits between the open gripper fingers, not only on the two smallest axes, which matters for elongated and box-like primitives that are approached from above. Second, each grasp is slid along its approach direction as far as the open hand remains free of the reconstructed scene, so that the fingers close as deep on the object as the clutter allows. Third, candidates are collision-checked against the superquadrics of all other objects and the table, and the remaining grasps are ranked by the product of five factors: how far the pads overlap the primitive, how close the approach is to vertical, how well the pose sits within the arm's reach, how square the contact is against the surface, and how much of the gripper's stroke the bite leaves in reserve. We keep the best 16 per object from a pool of 64. The sampler is purely geometric: it reads the primitives only, never the camera points.