MVB-Grasp: Minimum-Volume-Box Filtering of Diffusion-based Grasps for Frontal Manipulation

Bibek Poudel*, Abdul Basit*, Muhammad Shafique
eBRAIN Lab, Division of Engineering New York University (NYU) Abu Dhabi, Abu Dhabi, UAE
{bp2376, abdul.basit, muhammad.shafique}@nyu.edu

***Abstract*—State-of-the-art 6-DoF grasp generators excel on tabletop benchmarks with overhead cameras but struggle in frontal grasping scenarios on low-cost manipulators with constrained workspaces, where kinematic limits and approach-direction constraints cause high failure rates. We address this challenge for the Unitree Z1 arm by proposing MVB-Grasp, a novel grasping stack that injects a Minimum Volume Bounding Box (MVBB) geometric prior into diffusion-based grasp generation to dramatically improve success rates in frontal, workspace-constrained settings. Our key scientific contributions are threefold: (i) an MVBB-based geometric filter that exploits oriented bounding-box face normals to reject grasps approaching through the table or misaligned with accessible object faces in $\mathcal{O}(N)$ time; (ii) a combined re-scoring function that blends learned discriminator scores with face-alignment geometry ($\alpha = 0.85$), specifically calibrated for the Z1's frontal workspace and kinematic constraints; and (iii) a systematic MuJoCo evaluation protocol measuring grasp success across object types, distances, lateral positions, and pitch orientations to validate embodiment-specific performance. We implement MVB-Grasp on a Unitree Z1 arm with an Intel RealSense D405 camera, integrating YOLOv8 object detection, GraspGen for candidate generation, Principal Component Analysis (PCA)-based MVBB fitting, and inverse-kinematics trajectory planning. Experiments across 81 MuJoCo episodes (cylinder, asymmetric box, waterbottle) demonstrate that MVB-Grasp achieves 59.3% success versus 24.7% for vanilla GraspGen, a 2.4× improvement, by filtering geometrically infeasible candidates and prioritizing face-aligned grasps suited to the Z1's frontal approach constraints. Real-world trials confirm that the MVBB prior substantially improves grasp reliability on constrained, low-cost manipulators without requiring model retraining.**

***Index Terms*—Robotic grasping, diffusion-based grasp synthesis, MVBB, geometric filtering, frontal grasp planning.**

I. Introduction

Grasping is the primary interface between a robot and its environment: without reliable grasp acquisition, manipulation skills such as rearrangement, tool use, and pick–place are infeasible in everyday settings [1]. Recent advances in 6-DoF grasp synthesis have produced diverse deep models operating on point clouds and RGB–D data, including sampling-based methods (GPD, Contact-GraspNet) [2], [3], dense predictors (AnyGrasp) [4], and diffusion-based generators (GraspGen) [5]. These systems achieve strong performance on tabletop and cluttered-scene benchmarks, typically under top-down or mildly oblique viewpoints.

However, deploying such powerful 6-DoF grasp generators on low-cost manipulators with constrained workspaces remains challenging. Most state-of-the-art methods are trained and evaluated in scenarios with overhead cameras and gravity-aligned approaches, but affordable arms such as the Unitree Z1 often operate in *frontal grasping* settings with side-mounted cameras and tight kinematic constraints. In these scenarios, many generated grasps are geometrically infeasible, approaching through the support surface, deeply penetrating object faces, or requiring joint configurations beyond the arm's limits. The embodiment-agnostic nature of general-purpose generators means they cannot anticipate which approach directions are accessible for a specific manipulator, leading to high failure rates despite strong learned discriminators.

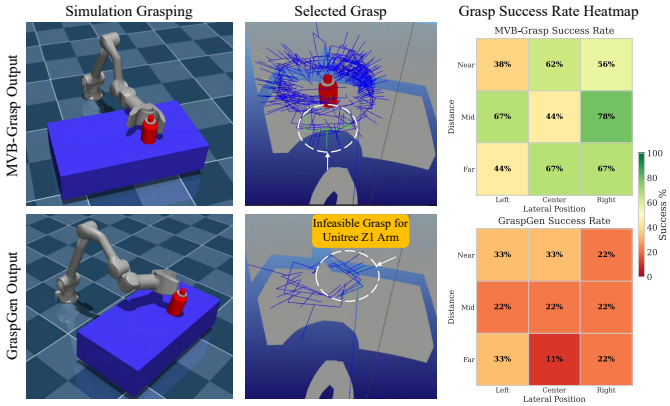


Fig. 1. Motivation for MVB-Grasp. Top: the proposed MVB-Grasp selects stable frontal grasps that succeed in simulation and achieve consistently high success rates across the workspace. Bottom: the vanilla GraspGen baseline often selects less robust grasps, yielding lower and spatially inconsistent success. From left to right we show the simulated grasp execution, the distribution of sampled and selected grasps, and the empirical success-rate heatmaps over distance–lateral bins for each method.

A. Motivational case study: frontal grasping failures on Z1

We study tabletop frontal grasping with the Unitree Z1 arm using a side-mounted RGB–D camera, integrating GraspGen with both MuJoCo simulation and a calibrated real setup. In its vanilla form, GraspGen samples K grasp candidates, ranks them with a learned discriminator, and applies collision checks before execution.

Two observations motivate this work. First, many Z1 failures, such as through-table approaches, deep penetrations, or joint-infeasible poses, are geometrically evident from the point cloud, yet are ranked highly by GraspGen's embodiment-agnostic discriminator. Second, simple geometric cues, including object face normals and principal axes, often suffice to distinguish

* Authors have equal contributions.

accessible from infeasible approaches in the Z1's frontal workspace (Fig. 1).

In this setting, a PCA-fitted oriented bounding box (OBB), which we use as a practical MVBB approximation, captures key object geometry, including extents, support plane, and dominant face normals. Many invalid grasps can be rejected using OBB/MVBB alignment alone, without physics simulation. Thus, improving frontal 6-DoF grasping on Z1 reduces to three key questions: how to design a geometric prior that filters infeasible grasps while preserving valid ones, how to integrate this prior with general-purpose generators like GraspGen without retraining, and how to characterize when such geometry helps, or hurts, grasp success.

B. Scientific challenges targeted in this paper

Towards this, we formalize three scientific challenges. First, *geometric priors for embodiment-specific filtering*: how simple object geometry (e.g., OBB/MVBB) can remove grasps that are infeasible for a frontal, workspace-limited arm while preserving sufficient grasp diversity. Second, *training-free embodiment adaptation*: how to re-score candidates from a general grasp generator such as GraspGen so that they favor the Z1's frontal workspace without retraining the generator or its discriminator. Third, *when geometry helps, or hurts*: under what object shapes and viewing conditions geometric priors improve grasp success, and when they over-constrain generation, especially for asymmetric or close-range objects.

C. Our contributions

We propose a training-free alignment strategy that augments an off-the-shelf 6-DoF grasp generator (GraspGen) for frontal grasping with the Unitree Z1 arm. Our contributions are:

1) **OBB-based alignment prior for frontal 6-DoF grasping:** We apply a PCA-based OBB, used as a lightweight MVBB prior, to extract face normals, filter misaligned grasps, and re-score candidates by combining geometric alignment with GraspGen's discriminator. This training-free prior integrates with a diffusion-based generator and improves frontal grasp success on a workspace-constrained arm.
2) **Structured frontal-grasp evaluation on Unitree Z1:** We introduce an 81-scenario MuJoCo benchmark and compare vanilla GraspGen with the OBB-augmented pipeline, analyzing success rates, failure modes, and candidate statistics across distance, lateral position, occlusion, and object orientation.
3) **Geometry–learning trade-off analysis:** We study per-object, distance, and orientation effects, identifying when alignment priors help (e.g., symmetric objects) and when they over-constrain generation (e.g., asymmetric objects at close range).

Overall, injecting this OBB prior into GraspGen yields a 2.4× improvement in simulated success (59.3% vs. 24.7%) with only 6.78 ms additional latency, making the approach practical for real-time deployment. Although our experiments focus on GraspGen and the Z1 arm, the OBB module is architecture-agnostic and provides a template for embodiment-specific tuning of general-purpose grasp perception systems.

II. Related work and background

Robotic grasping has evolved from analytic planning to learned vision-based synthesis. Early methods [6]–[9] predict planar grasp rectangles in image space assuming top-down approaches with overhead cameras. While highly successful for bin-picking and similar setups, these approaches are fundamentally limited to near-planar (4-DoF) grasps and do not directly support arbitrary 6-DoF poses or frontal approaches.

Modern 6-DoF systems address this through three main paradigms. (1) *Sampling-based generators* such as 6-DoF GraspNet and Contact-GraspNet sample grasp poses in $SE(3)$ and refine them using learned quality metrics and collision checks [3], [10]. They offer strong success in clutter but are designed for general scenarios rather than specific embodiments. (2) *Dense prediction methods* including AnyGrasp predict grasp scores densely over the visible scene in a single feed-forward pass [4], enabling temporally smooth execution and robustness to depth noise, but still requiring downstream collision checking and planning specific to the robot. (3) *Generative models* such as diffusion-based GraspGen model the distribution of successful grasps conditioned on point clouds [5]. These models can generate diverse grasps across gripper types, but are typically deployed as general modules without embodiment-specific tuning or explicit latency-aware design.

Table I summarizes representative frameworks along two axes that are central to this work: (i) the grasp representation (planar vs. 6-DoF), and (ii) the dominant approach mode (top-down vs. frontal/general). Most frameworks are optimized for top-down or general 6-DoF scenarios; none explicitly target frontal grasping with a Z1-like arm.

TABLE I
Comparison of representative vision-based grasp frameworks.

Framework	Grasp DoF	Approach mode
Lenz *et al.* [6]	Planar (4-DoF)	Top-down
Redmon & Angelova [7]	Planar (4-DoF)	Top-down
GG-CNN [8]	Planar (4-DoF)	Top-down / wrist
Dex-Net 2.0 [9]	Planar (4-DoF)	Top-down bin picking
6-DoF GraspNet [10]	6-DoF samples	General 6-DoF
Contact-GraspNet [3]	6-DoF dense	General 6-DoF
AnyGrasp [4]	6/7-DoF dense	Multi-view, mostly top-down
GraspGen [5]	6-DoF diffusion	General 6-DoF
MVBB grasping [11], [12]	Planar (4-DoF)	Various
GoalGrasp [13]	6-DoF	General 6-DoF
MVB-Grasp (Ours)	**6-DoF + MVBB**	**Frontal, Z1-specific**

Grasp DoF: grasp representation (planar vs. full 6-DoF). Approach mode: dominant camera viewpoint and approach direction in evaluations.

A complementary line of work exploits geometric primitives as priors for grasp planning. Hübner *et al.* [11] decompose objects into MVBBs for planar grasping in simulation, and Geidenstam *et al.* [12] learn 2D strategies from box-based approximations. More recent methods use oriented boxes with machine learning [14], [15], but still target planar grasps. In 6-DoF settings, GoalGrasp [13] uses 3D box priors for user-specified objects under occlusion, and AnyGrasp or Contact-GraspNet employ geometry mainly for cropping and simple clearance checks [3], [4]. In these systems, geometric reasoning primarily serves to remove obviously invalid candidates or

localize objects; the core scoring models remain generic and are not tuned to a specific arm's workspace, preferred approach directions (e.g., frontal vs. vertical), or latency budget.

Most prior work therefore optimizes grasp *accuracy and generality* across diverse robots and scenarios, while embodiment-specific constraints and end-to-end latency are treated as downstream filtering problems. Recent efforts that encode physical or geometric priors for 6-DoF grasping [16], [17] show that injecting structure can improve robustness, but they do not specifically address the *frontal grasping* challenge on workspace-constrained, low-cost manipulators like the Z1.

III. Methodology

This section presents the design and implementation of **MVB-Grasp**, a latency-aware grasping pipeline that integrates a MVBB prior with a state-of-the-art diffusion-based grasp generator. We begin with a system overview, then describe each component in detail: perception, grasp generation, the MVBB module, grasp filtering and re-scoring, and Z1 arm execution.

A. System Overview

Fig. 2 illustrates the complete MVB-Grasp pipeline. Given an RGB-D observation from a RealSense D405 camera mounted on or near the Unitree Z1 arm, our system performs the following stages:

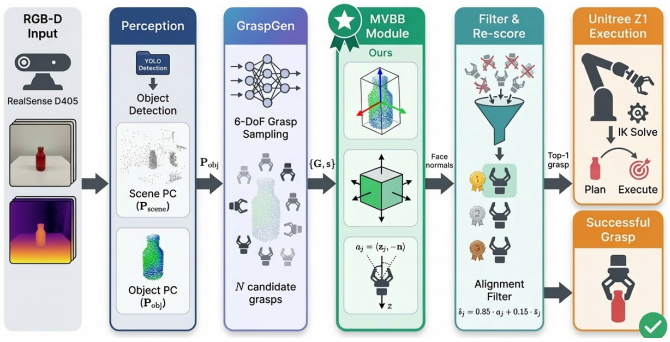


Fig. 2. Overview of MVB-Grasp: RGB-D images are processed to obtain an object point cloud, GraspGen samples 6-DoF candidate grasps, our MVBB module extracts face normals to filter and re-score candidates for frontal alignment, and the top-ranked grasp is executed on the Unitree Z1.

1) **Perception**: Acquire RGB-D frames, detect the target object via YOLO, and segment the object point cloud $\mathcal{P}_{\text{obj}}$ from the scene point cloud $\mathcal{P}_{\text{scene}}$.
2) **Grasp Generation**: Feed $\mathcal{P}_{\text{obj}}$ into GraspGen [5] to sample N candidate 6-DoF grasp poses $\{(\mathbf{G}_j, s_j)\}_{j=1}^{N}$, where $\mathbf{G}_j \in SE(3)$ and $s_j \in [0,1]$ is the discriminator confidence.
3) **MVBB Module**: Fit an OBB to $\mathcal{P}_{\text{obj}}$, extract face normals, and compute alignment scores between each grasp's approach axis and the nearest OBB faces.
4) **Filtering & Re-scoring**: Reject grasps that do not align with any accessible face; re-score remaining grasps by combining geometric alignment with the original confidence.
5) **Execution**: Select the top-ranked grasp, solve inverse kinematics, plan a collision-free trajectory, and execute on the Z1 arm.

The key innovation is that the MVBB-based geometric prior (stages 3–4) can aggressively prune infeasible grasps in $\mathcal{O}(N)$ time, often eliminating the need for expensive physics-based collision checking that scales as $\mathcal{O}(N \cdot |\mathcal{P}_{\text{scene}}|)$.

B. Perception Pipeline

1) RGB-D Acquisition: We use an Intel RealSense D405 depth camera operating at 640×480 resolution at 30 FPS. The depth stream is aligned to the color stream to ensure pixel correspondence. Let $I_{\text{rgb}} \in \mathbb{R}^{H \times W \times 3}$ denote the RGB image and $I_{\text{depth}} \in \mathbb{R}^{H \times W}$ the aligned depth map (in meters).

2) Object Detection: We employ YOLOv8 [18] for real-time object detection. Given I_{rgb}, the detector outputs a set of bounding boxes $\mathcal{B} = \{(x_1^{(i)}, y_1^{(i)}, x_2^{(i)}, y_2^{(i)}, c^{(i)})\}$ for objects of interest, where $c^{(i)}$ is the detection confidence. In our experiments, we typically target a single object (e.g., a bottle) and select the highest-confidence detection.

3) Point Cloud Generation and Segmentation: We convert the depth image to a 3D point cloud using the camera intrinsics (f_x, f_y, c_x, c_y):

$$\mathbf{p}_{u,v} = \begin{bmatrix} (u - c_x) \cdot d_{u,v} / f_x \\ (v - c_y) \cdot d_{u,v} / f_y \\ d_{u,v} \end{bmatrix}, \quad d_{u,v} = I_{\text{depth}}(u, v) \tag{1}$$

where (u, v) are pixel coordinates. Points with invalid depth ($d_{u,v} \leq 0$ or $d_{u,v} > d_{\max}$) are discarded.

The scene point cloud is: $\mathcal{P}_{\text{scene}} = \{\mathbf{p}_{u,v} \mid d_{u,v} \in (0, d_{\max}]\}$

The object point cloud is segmented by retaining only points whose pixel coordinates fall within the detected bounding box:

$$\mathcal{P}_{\text{obj}} = \{\mathbf{p}_{u,v} \in \mathcal{P}_{\text{scene}} \mid x_1 \leq u \leq x_2,\ y_1 \leq v \leq y_2\} \tag{2}$$

To remove outliers and noise from $\mathcal{P}_{\text{obj}}$, we apply:

- **Depth percentile clipping**: Remove points outside the $[1\%, 99\%]$ depth percentile to eliminate spurious spikes.
- **Statistical outlier removal**: Remove points whose mean distance to k nearest neighbors exceeds $\mu + 2\sigma$.
- **Radius outlier removal**: Remove points with fewer than $n_{\min}$ neighbors within radius r.

C. GraspGen Integration

We integrate GraspGen [5], a diffusion-based 6-DoF grasp generator that operates on point clouds. GraspGen models the distribution of successful grasps conditioned on the object geometry using a denoising diffusion probabilistic model (DDPM) coupled with a PointTransformerV3 backbone.

1) Grasp Representation: Each grasp pose is represented as a rigid transformation $\mathbf{G} \in SE(3)$, parameterized as a 4×4 homogeneous matrix:

$$\mathbf{G} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix}, \quad \mathbf{R} \in SO(3),\ \mathbf{t} \in \mathbb{R}^3 \tag{3}$$

where $\mathbf{R}$ is the grasp orientation and $\mathbf{t}$ is the grasp position. The grasp frame convention follows the Robotiq 2F-140 gripper: the z-axis points along the approach direction (into the palm),

the y-axis is parallel to the finger opening direction, and the x-axis completes the right-handed frame.

2) Sampling and Scoring: Given $\mathcal{P}_{\text{obj}}$, we invoke GraspGen to sample N candidate grasps: $\{(\mathbf{G}_j, s_j)\}_{j=1}^{N} = \textsc{GraspGen}(\mathcal{P}_{\text{obj}}, N)$ where $s_j \in [0,1]$ is the discriminator score indicating grasp quality. In our experiments, we use $N = 800$ grasps per scene. The scores are normalized to $[0,1]$:

$$\bar{s}_j = \frac{s_j - s_{\min}}{s_{\max} - s_{\min} + \epsilon} \tag{4}$$

where $s_{\min} = \min_j s_j$, $s_{\max} = \max_j s_j$, and $\epsilon = 10^{-6}$ prevents division by zero.

D. MVBB Module

The core contribution of MVB-Grasp is the MVBB module, which exploits the geometry of an oriented bounding box fitted to the object point cloud to filter and re-score grasp candidates.

1) Oriented Bounding Box Fitting: Given $\mathcal{P}_{\text{obj}} = \{\mathbf{p}_i\}_{i=1}^{M}$, we compute the MVBB using PCA followed by convex hull optimization.

First, we compute the centroid: $\boldsymbol{\mu} = \frac{1}{M}\sum_{i=1}^{M} \mathbf{p}_i$

The covariance matrix is:

$$\boldsymbol{\Sigma} = \frac{1}{M-1}\sum_{i=1}^{M}(\mathbf{p}_i - \boldsymbol{\mu})(\mathbf{p}_i - \boldsymbol{\mu})^\top \tag{5}$$

Eigendecomposition yields the principal axes:

$$\boldsymbol{\Sigma} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^\top, \quad \mathbf{V} = [\mathbf{v}_1 \mid \mathbf{v}_2 \mid \mathbf{v}_3] \tag{6}$$

where $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ are the eigenvectors ordered by decreasing eigenvalue $\lambda_1 \geq \lambda_2 \geq \lambda_3$.

The OBB is defined by:

- **Rotation matrix**: $\mathbf{R}_{\text{obb}} = [\mathbf{v}_1 \mid \mathbf{v}_2 \mid \mathbf{v}_3] \in SO(3)$
- **Center**: $\mathbf{c}_{\text{obb}} \in \mathbb{R}^3$ (refined from $\boldsymbol{\mu}$)
- **Extents**: $\mathbf{e} = (e_1, e_2, e_3)$ giving half-lengths along each principal axis

The transformation from the world frame to the OBB frame:

$$\mathbf{T}_{\text{obb}} = \begin{bmatrix} \mathbf{R}_{\text{obb}} & \mathbf{c}_{\text{obb}} \\ \mathbf{0}^\top & 1 \end{bmatrix} \tag{7}$$

2) Face Extraction: The OBB has six faces, each defined by a face normal $\mathbf{n}$ and face center $\mathbf{c}$. For each principal axis $i \in \{1,2,3\}$ and sign $\sigma \in \{-1,+1\}$:

$$\mathbf{n}_{i,\sigma} = \sigma \cdot \mathbf{v}_i \tag{8}$$

$$\mathbf{c}_{i,\sigma} = \mathbf{c}_{\text{obb}} + \frac{e_i}{2} \cdot \mathbf{n}_{i,\sigma} \tag{9}$$

This yields six face descriptors:

$$\mathcal{F}_{\text{all}} = \{(\mathbf{n}_{i,\sigma}, \mathbf{c}_{i,\sigma}) \mid i \in \{1,2,3\}, \sigma \in \{-1,+1\}\} \tag{10}$$

While the above equations define the MVBB module analytically, Fig. 3 provides a procedural view of how it operates. Starting from the segmented object point cloud, the left column shows OBB fitting and extraction of the six face planes, followed by the selection of the k faces closest to the camera. The right column illustrates how, for each GraspGen candidate, we compute the approach axis, evaluate its alignment to the selected faces, reject misaligned grasps, and assign the combined score $\hat{s}_j$ that blends geometric alignment with the normalized GraspGen confidence. This flow directly corresponds to the MVBB-related calls in Algorithm 1.

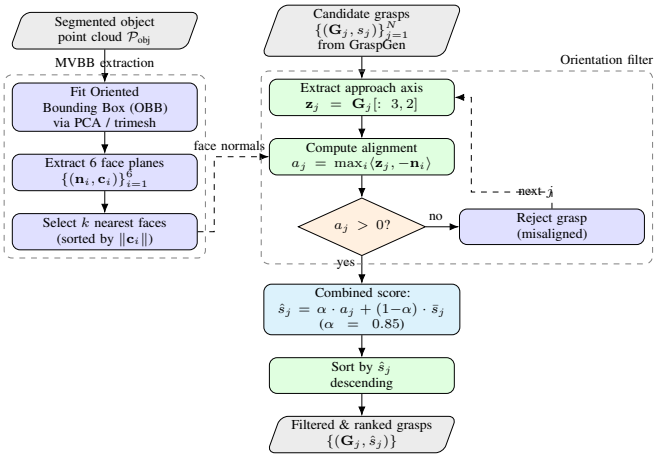


Fig. 3. MVBB-based grasp filtering and scoring. Left: MVBB extraction from the object point cloud. Right: Each candidate grasp is checked for alignment with the nearest OBB faces; aligned grasps receive a combined score mixing geometric alignment ($\alpha = 0.85$) and the original GraspGen confidence.

3) Face Selection: Not all faces are accessible for frontal grasping on a tabletop. We select the k nearest faces (typically $k = 2$) by sorting face centers by their distance to the camera origin (or world origin): $\mathcal{F} = \textsc{TopK}\,(\mathcal{F}_{\text{all}}, k, \text{key} = \|\mathbf{c}_{i,\sigma}\|)$

This heuristic prioritizes faces that are closer to the robot (and thus more likely to be reachable), while also implicitly avoiding the bottom face resting on the table.

E. Grasp Filtering and Re-scoring

The MVBB module enables two key operations: (1) filtering out geometrically infeasible grasps, and (2) re-scoring surviving grasps to favor those aligned with accessible object faces.

1) Alignment Computation: For each grasp $\mathbf{G}_j$, we extract the approach axis (the z-column of the rotation matrix): $\mathbf{z}_j = \mathbf{G}_j[:, 3, 2] = \mathbf{R}_j \mathbf{e}_z$ where $\mathbf{e}_z = [0,0,1]^\top$ is the unit vector along the local z-axis.

A grasp approaching a face should have its approach axis pointing *into* the face, i.e., opposite to the outward face normal. The alignment score with face $(\mathbf{n}, \mathbf{c})$ is: $a(\mathbf{z}_j, \mathbf{n}) = \langle \mathbf{z}_j, -\mathbf{n} \rangle = -\mathbf{z}_j^\top \mathbf{n}$. The best alignment over all selected faces is:

$$a_j = \max_{(\mathbf{n},\mathbf{c}) \in \mathcal{F}} a(\mathbf{z}_j, \mathbf{n}) = \max_{(\mathbf{n},\mathbf{c}) \in \mathcal{F}} \left(-\mathbf{z}_j^\top \mathbf{n}\right) \tag{11}$$

Note that $a_j \in [-1, 1]$, where $a_j = 1$ indicates perfect alignment (grasp approaches perpendicular to a face), and $a_j \leq 0$ indicates the grasp is pointing away from all selected faces.

2) Hard Filtering: We reject grasps that do not align with any accessible face: $\mathcal{G}_{\text{pass}} = \{(\mathbf{G}_j, s_j) \mid a_j > 0\}$. This simple threshold eliminates grasps that approach through the table, from behind the object, or at extreme tangential angles. In practice, this filter typically removes 15–35% of candidates, depending on object shape and pose.

3) Combined Re-scoring: For grasps that pass the hard filter, we compute a combined score that blends the geometric alignment with the original GraspGen confidence: $\hat{s}_j = \alpha \cdot a_j + (1-\alpha) \cdot \bar{s}_j$ where $\alpha \in [0,1]$ controls the trade-off between geometric alignment and learned quality. We use $\alpha = 0.85$ in all experiments, emphasizing geometric feasibility for the Z1 arm's constrained workspace.

4) Ranking: The filtered grasps are ranked by descending combined score: $\mathcal{G}_{\text{ranked}} = \text{SORTDESCENDING}(\mathcal{G}_{\text{pass}}, \text{key} = \hat{s})$. The top-$k$ grasps (typically $k = 1$ or $k = 3$) are passed to the execution stage.

F. Z1 Arm Execution

1) Coordinate Frame Transformations: The grasp poses from GraspGen are expressed in the camera frame $\{C\}$. To execute on the Z1 arm, we transform to the robot base frame $\{B\}$: $\mathbf{G}_j^B = \mathbf{T}_{BC} \cdot \mathbf{G}_j^C$, where $\mathbf{T}_{BC} \in SE(3)$ is the camera-to-base extrinsic calibration.

2) Inverse Kinematics: The Z1 arm has 6 revolute joints. Given a target end-effector pose $\mathbf{G}_{\text{target}}^B$, we solve for joint angles $\mathbf{q} \in \mathbb{R}^6$ using iterative inverse kinematics:

$$\mathbf{q}^* = \arg\min_{\mathbf{q}} \|\text{FK}(\mathbf{q}) - \mathbf{G}_{\text{target}}^B\|_{SE(3)}^2 + \lambda \|\mathbf{q} - \mathbf{q}_{\text{home}}\|^2 \quad (12)$$

subject to joint limits $\mathbf{q}_{\min} \leq \mathbf{q} \leq \mathbf{q}_{\max}$, where $\text{FK}(\cdot)$ is the forward kinematics and λ is a regularization term favoring solutions near the home configuration.

If IK fails for the top-ranked grasp, we proceed to the next candidate in $\mathcal{G}_{\text{ranked}}$.

3) Trajectory Planning and Execution: Once a valid IK solution is found, we plan a trajectory from the current configuration $\mathbf{q}_{\text{curr}}$ to the pre-grasp pose (offset along the approach axis), then to the final grasp pose:

1) Move to pre-grasp: $\mathbf{G}_{\text{pre}} = \mathbf{G}_{\text{target}}^B \cdot \mathbf{T}_{\text{offset}}$, where $\mathbf{T}_{\text{offset}}$ translates $-d_{\text{pre}}$ along the z-axis.
2) Approach: Linear motion from $\mathbf{G}_{\text{pre}}$ to $\mathbf{G}_{\text{target}}^B$.
3) Close gripper.
4) Lift: Retract along the z-axis by d_{lift}.

4) Optional Collision Checking: For deployments where additional safety is required, we optionally perform collision checking between the gripper mesh and $\mathcal{P}_{\text{scene}}$. For each grasp candidate $\mathbf{G}_j$, we transform the gripper collision mesh vertices $\mathcal{V}_{\text{gripper}}$ to the world frame and check for penetration:

$$\text{collision}(\mathbf{G}_j) = \exists \mathbf{v} \in \mathcal{V}_{\text{gripper}}^{\mathbf{G}_j} : \min_{\mathbf{p} \in \mathcal{P}_{\text{scene}}} \|\mathbf{v} - \mathbf{p}\| < \tau_{\text{coll}} \quad (13)$$

where τ_{coll} is the collision threshold (typically 2 mm). However, a key hypothesis of MVB-Grasp is that the MVBB filtering makes collision checking optional for many scenarios, substantially reducing latency.

G. Complexity Analysis

The computational complexity of each pipeline stage depends on the input size:

- **Perception**: Depth-to-pointcloud conversion and YOLO detection scale as $\mathcal{O}(HW)$ with image resolution.
- **GraspGen inference**: Scales as $\mathcal{O}(M \cdot N)$ where M is the object point cloud size and N is the number of generated grasps.
- **OBB fitting**: PCA-based fitting is $\mathcal{O}(M)$ linear in point cloud size.
- **Alignment filtering**: Checking N grasps against k faces is $\mathcal{O}(N \cdot k)$, which is very fast since $k \ll N$.
- **Collision checking**: The bottleneck in vanilla pipelines, scaling as $\mathcal{O}(N \cdot |\mathcal{V}| \cdot |\mathcal{P}_{\text{scene}}|)$ where $|\mathcal{V}|$ is the number of gripper mesh vertices.

The key insight is that MVBB-based filtering ($\mathcal{O}(N \cdot k)$) is orders of magnitude faster than physics-based collision checking. Detailed latency measurements are presented in Section V.

The same MVB-Grasp stack is used both for systematic evaluation in MuJoCo and for online execution on the physical Z1. Fig. 4 visualizes this dual use: the left branch shows the offline simulation and calibration loop, and the right branch shows the online runtime pipeline deployed on the robot.

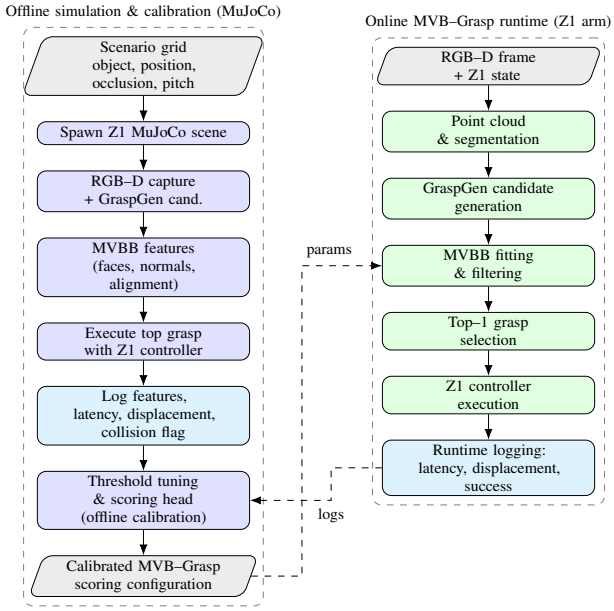


Fig. 4. Offline simulation and calibration pipeline (left) and online MVB–Grasp runtime on the Z1 arm (right). Solid arrows indicate forward execution; dashed arrows indicate feedback from runtime logs and calibrated parameters.

IV. Experimental Setup

A. Simulation Setup (MuJoCo)

Environment and scenario grid: All simulation experiments are run in MuJoCo with a physics-accurate Unitree Z1 Pro and parallel-jaw gripper in a tabletop workspace (Fig. 5). Each episode contains a single object (cylinder, asymmetric box, or water bottle) placed on the table according to a structured grid with: *distance* (Near, Mid, Far), *lateral offset* (Left, Center, Right), and *pitch* ($-45°$, $0°$, $+45°$), yielding 81 distinct configurations detailed in Algorithm 2.

Algorithm 1 MVB-Grasp: Latency-aware 6-DoF grasping with MVBB prior

Require: $\mathcal{C}, \mathcal{G}, \mathcal{A}, \mathcal{D}, \mathcal{H}, \alpha, k, N$
Ensure: $\mathbf{G}^*$
$(I_{\text{rgb}}, I_{\text{depth}}) \leftarrow \mathcal{C}.\textsc{Capture}()$
$\mathcal{P}_{\text{obj}} \leftarrow \textsc{SegmentObject}(\textsc{DepthToPointCloud}(I_{\text{depth}}), \mathcal{D}.\textsc{Detect}(I_{\text{rgb}}))$
$\{(\mathbf{G}_j, s_j)\}_{j=1}^{N} \leftarrow \mathcal{G}.\textsc{Sample}(\mathcal{P}_{\text{obj}}, N)$
$\bar{s}_j \leftarrow \textsc{Normalize}(\{s_j\})$
$\mathcal{F} \leftarrow \textsc{FitMVBBAndNearestFaces}(\mathcal{P}_{\text{obj}}, k)$
$\mathcal{G}_{\text{filtered}} \leftarrow \{(\mathbf{G}_j, \alpha a_j + (1-\alpha)\bar{s}_j) \mid a_j > 0\}$
  where $a_j = \max_{(\mathbf{n},\cdot)\in\mathcal{F}} \langle \mathbf{G}_j[:3,2], -\mathbf{n} \rangle$
$\mathcal{G}_{\text{ranked}} \leftarrow \textsc{SortDescending}(\mathcal{G}_{\text{filtered}})$
for $(\mathbf{G}, \hat{s}) \in \mathcal{G}_{\text{ranked}}$ **do**
  $\mathbf{q} \leftarrow \textsc{IKSolve}(\mathcal{A}, \mathbf{G})$
  if $\mathbf{q} \neq$ None **and** (**not** $\textsc{CollisionCheckEnabled}$ **or not** $\textsc{CheckCollision}(\mathcal{H}, \mathbf{G}, \mathcal{P}_{\text{scene}})$) **then**
    $\mathbf{G}^* \leftarrow \mathbf{G}$; **break**
$\textsc{PlanAndExecute}(\mathcal{A}, \mathbf{q}, \mathbf{G}^*)$
return $\mathbf{G}^*$

Algorithm 2 Systematic grasp evaluation using MuJoCo

Require: Objects $\mathcal{O}$; grid $\mathcal{X} = \{\text{Near, Mid, Far}\} \times \{\text{Left, Center, Right}\}$; occlusion levels $\mathcal{L}$; pitches Θ; MuJoCo scene $\mathcal{S}$; camera pose $\mathbf{T}_{\text{cam}}$; GraspGen model $\mathcal{G}$; methods $\mathcal{M} = \{\textsc{Vanilla}, \text{MVB-}\textsc{Grasp}\}$
Ensure: Result log $\mathcal{R}$
$\mathcal{R} \leftarrow \emptyset$
for $o \in \mathcal{O}$, $(x, y) \in \mathcal{X}$, $\ell \in \mathcal{L}$, $\theta \in \Theta$ **do**
  $\textsc{ResetScene}(\mathcal{S})$; $\textsc{SpawnObject}(\mathcal{S}, o, x, y, \theta, \ell)$
  $(I_{\text{rgb}}, I_{\text{depth}}) \leftarrow \textsc{RenderRGBD}(\mathcal{S}, \mathbf{T}_{\text{cam}})$
  $P_{\text{obj}}, P_{\text{scene}} \leftarrow \textsc{SegmentPointClouds}(I_{\text{depth}}, o)$
  $\{(G_j, s_j)\}_{j=1}^{N} \leftarrow \mathcal{G}.\textsc{Sample}(P_{\text{obj}}, N)$
  for $m \in \mathcal{M}$ **do**
    if $m = \textsc{Vanilla}$ **then**
      $G_{\text{ranked}} \leftarrow \textsc{SortByScore}(\textsc{CollisionFilter}(\{(G_j, s_j)\}, P_{\text{scene}}))$
    else ▷ MVB-Grasp
      $F \leftarrow \textsc{ComputeOBBFaces}(P_{\text{obj}})$
      $G_{\text{ranked}} \leftarrow \text{MVBB}\textsc{FilterAndRescore}(\{(G_j, s_j)\}, F)$
    success $\leftarrow \textsc{ExecuteTopK}(\mathcal{S}, G_{\text{ranked}}, k, \tau_{\text{lift}})$
    Append $(o, x, y, \ell, \theta, m,$ **success**$, |G_{\text{ranked}}|)$ to $\mathcal{R}$
return $\mathcal{R}$

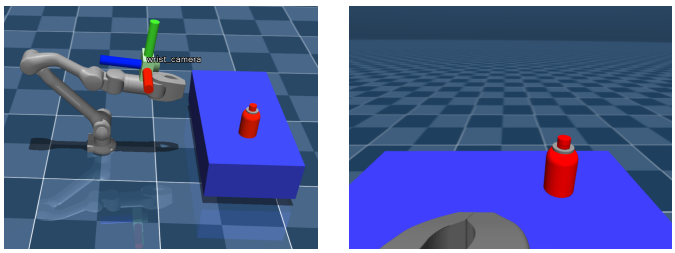


Fig. 5. MuJoCo simulation setup: (a) Unitree Z1 Pro arm in a tabletop scene, (b) wrist-mounted RGB–D view used for point cloud reconstruction.

Perception and candidate generation: From a fixed home configuration $q_{\text{home}} = [0, 1.085, -0.261, -0.523, 0, 1.57]$ rad, we render synchronized RGB–D images, back-project to a scene point cloud, and segment the target object using the known mask. The segmented point cloud is passed to GraspGen to generate K 6-DoF grasp candidates per episode (Fig. 6).

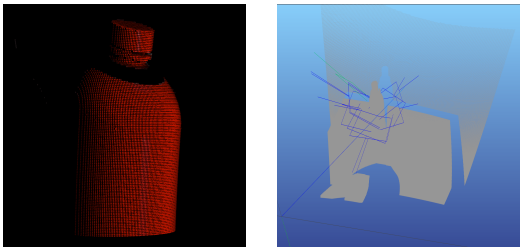


Fig. 6. Perception and grasp candidates in simulation: (a) segmented object point cloud, (b) example GraspGen 6-DoF grasp proposals.

MVBB filtering and execution: An oriented MVBB is fitted to the object point cloud to obtain face centers and normals (Fig. 7). The k faces closest to the camera are selected, grasp candidates are filtered by approach alignment to these faces, and surviving grasps are re-scored using the MVBB alignment score and GraspGen confidence. The top-ranked grasp is transformed to the robot base frame and executed via a Cartesian trajectory (LERP position, SLERP orientation). In simulation, a grasp is counted as successful if the object is lifted and stably retained after the closing and lift motion.

Platform and calibration: Real-world experiments use a Unitree Z1 Pro with a wrist-mounted Intel RealSense D405

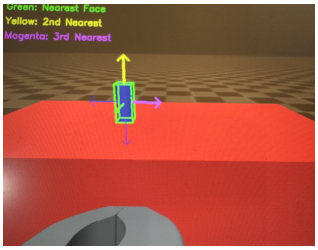


Fig. 7. MVBB in simulation: fitted oriented box and three selected nearest faces used for alignment-based filtering.

RGB–D camera (Fig. 8). Camera intrinsics and hand–eye calibration are obtained using a ChArUco board, and the table and camera poses are arranged to closely match the MuJoCo setup for sim-to-real comparison.

B. Real-World Setup (Z1 Hardware)

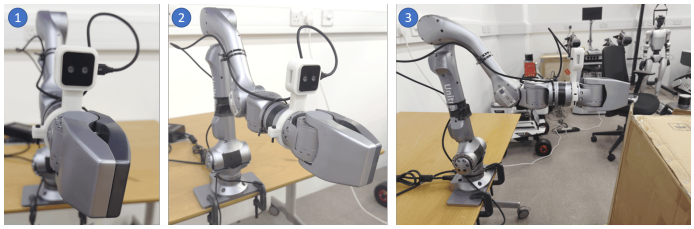
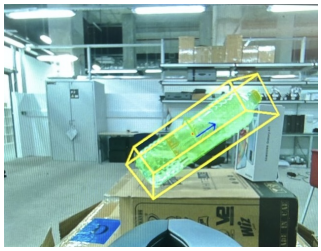


Fig. 8. Hardware setup: (1, 2) Unitree Z1 Pro arm with Intel RealSense D405 camera, (3) tabletop view mirroring the MuJoCo configuration.

Object placement and execution. We consider a single target object (water bottle) placed within the reachable workspace with pose sampled from the same distance–lateral–pitch grid as in simulation. RGB–D frames from the calibrated camera are back-projected to a scene point cloud, the bottle is segmented, and MVB-Grasp runs the same GraspGen + MVBB pipeline as in MuJoCo (Fig. 9). The top-ranked grasp is executed with conservative approach and lift distances to maintain table clearance and avoid joint limits.

Evaluation protocol and sim-to-real consistency. For each real-world trial we record grasp success, visible failure modes (table collisions, excessive penetration, IK failure), and whether the object remains stably held during lift and small translations



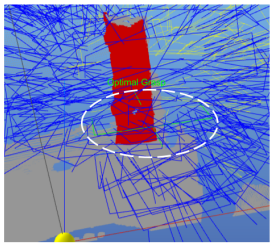


Fig. 9. Real-world MVBB pipeline: (a) MVBB fitted to the segmented bottle point cloud, (b) filtered grasp set with selected grasp shown in green.

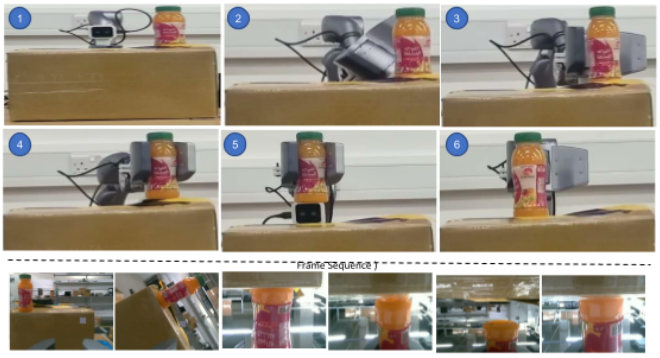


Fig. 10. Example real-world run: MVB-Grasp executing a frontal grasp on the Z1 arm.

(Figs. 10 and 11). Object placements are chosen to mirror the MuJoCo scenario grid, enabling direct comparison of success rates and failure modes between simulation and hardware.

Fig. 11. Qualitative slanted-object grasps with MVB-Grasp in the real world.

V. Results and Discussion

a) Overall performance.: Across all 81 MuJoCo scenarios, MVB-Grasp more than doubles success on the Z1 arm (59.3% vs. 24.7%; Table III) while using the *same* GraspGen candidates. Per-object gains in Table II are substantial: cylinders ($25.9 \rightarrow 85.1\%$), asymmetric boxes ($37.0 \rightarrow 55.5\%$), and bottles ($11.1 \rightarrow 40.0\%$). MVBB filtering also reduces the number of candidates forwarded downstream by 10–20%, improving the quality of the shortlist.

b) Distance and orientation effects.: Table V shows that MVB-Grasp consistently improves performance at all distances; for cylinders at *Far* range, success rises from 11.1% to 88.8%. Orientation trends in Table VI and Fig. 12 indicate that MVBB is particularly effective for symmetric objects (cylinder: $11.1 \rightarrow 100\%$ at $45°$), gives moderate but configuration-dependent gains for asymmetric boxes, and significantly helps but does not fully solve thin bottle geometry.

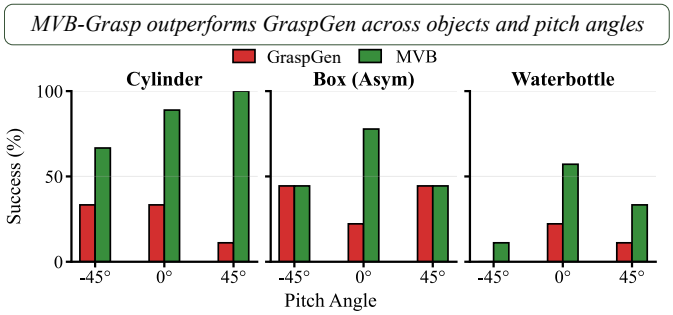


Fig. 12. Success rate by object and pitch angle (see Table VI).

c) Latency and dataset statistics.: As summarized in Table IV, MVBB adds only 6.78 ms of selection time (2947.4 ms vs. 2940.6 ms total), confirming that the geometric prior improves reliability with negligible latency overhead. Over 81 episodes per method (Table III), this leads to more than twice as many successful trials (48 vs. 20), supporting MVB-Grasp as a practical embodiment-aware adapter for existing 6-DoF grasp generators.

TABLE II
Z1 MuJoCo results aggregated over all positions, occlusions and pitch angles.

Object	Method	#Ep.	Succ. [%]	#Cand.	#After MVB
Cylinder	GraspGen	27	25.9	35	–
	MVB-Grasp	27	**85.1**	35	31.0
BoxAsym	GraspGen	27	37.0	110	–
	MVB-Grasp	27	**55.5**	110	92.2
Waterbottle	GraspGen	27	11.1	142	–
	MVB-Grasp	27	**40.0**	142	112.1

TABLE III
Dataset statistics for all logged Z1 MuJoCo episodes.

Method	#Ep.	#Cand. grasps	#Succ. ep.	#Fail ep.
GraspGen	81	7459	20	61
MVB-Grasp	81	7459	**48**	33

VI. Conclusion

We studied how to deploy a state-of-the-art 6-DoF grasp generator on a low-cost, workspace-constrained arm in a frontal grasping setting. While GraspGen performs well under standard top-down benchmarks, our Z1 experiments showed many grasps are infeasible for a side-mounted, frontal configuration.

TABLE IV
LATENCY BREAKDOWN OF GRASPGEN AND MVB-GRASP OVER ALL Z1 MUJOCO SCENARIOS.

Method	t_{gen} **[ms]**	t_{coll} **[ms]**	t_{sel} **[ms]**	t_{total} **[ms]**
GraspGen	463.34	2477.28	0.02	2940.64
MVB-Grasp	463.34	2477.28	6.78	2947.40

TABLE V
EFFECT OF DISTANCE AND OCCLUSION ON Z1 MUJOCO GRASPING PERFORMANCE.

Object	**Dist.**	**Method**	**Succ. [%]**	**#Cand.**	**#After MVB**
Cylinder	Near	GraspGen	33.3	35.0	–
	Near	**MVB-Grasp**	**88.8**	35.0	34.3
	Mid	GraspGen	33.3	21.1	–
	Mid	**MVB-Grasp**	**77.7**	21.1	18.2
	Far	GraspGen	11.1	48.7	–
	Far	**MVB-Grasp**	**88.8**	48.7	40.4
BoxAsym	Near	GraspGen	55.5	75.8	–
	Near	**MVB-Grasp**	44.4	75.8	66.6
	Mid	GraspGen	33.3	89.2	–
	Mid	**MVB-Grasp**	**77.7**	89.2	79.7
	Far	GraspGen	22.2	163.8	–
	Far	**MVB-Grasp**	**44.4**	163.8	130.3
Waterbottle	Near	GraspGen	0.0	135.3	–
	Near	**MVB-Grasp**	**14.3**	135.3	104.1
	Mid	GraspGen	0.0	117.0	–
	Mid	**MVB-Grasp**	**33.3**	117.0	97.2
	Far	GraspGen	33.3	173.0	–
	Far	**MVB-Grasp**	**44.4**	173.0	133.1

TABLE VI
ORIENTATION ROBUSTNESS: EFFECT OF OBJECT PITCH FOR ASYMMETRIC OBJECTS.

Object	**Pitch**	**Method**	**Succ. [%]**	**#Cand.**	**#MVB**
Cylinder	0°	GraspGen	33.3	45.2	–
	0°	**MVB**	**88.9**	45.2	43.1
	45°	GraspGen	11.1	26.0	–
	45°	**MVB**	**100.0**	26.0	22.4
	-45°	GraspGen	33.3	33.5	–
	-45°	**MVB**	**66.7**	33.5	27.4
BoxAsym	0°	GraspGen	22.2	115.7	–
	0°	**MVB**	**77.8**	115.7	92.4
	45°	GraspGen	44.4	103.2	–
	45°	**MVB**	44.4	103.2	87.8
	-45°	GraspGen	44.4	109.8	–
	-45°	**MVB**	44.4	109.7	96.2
Bottle	0°	GraspGen	22.2	126.1	–
	0°	**MVB**	**57.1**	126.1	109.7
	45°	GraspGen	11.1	127.1	–
	45°	**MVB**	**33.3**	127.1	96.0
	-45°	GraspGen	0.0	170.0	–
	-45°	**MVB**	**11.1**	170.0	130.0

The key idea of this paper is to use a simple, embodiment-aware geometric prior, an MVBB/OBB over the segmented object point cloud, to filter and re-score GraspGen candidates without any retraining. The proposed *MVB-Grasp* stack uses MVBB face normals to reject grasps that approach through inaccessible directions and to prioritize candidates aligned with faces reachable from the Z1's frontal workspace. On an 81-scenario MuJoCo benchmark across three objects, distances, and orientations, MVB-Grasp improves overall success from 24.7% to 59.3% (2.4×), with particularly strong gains on symmetric cylinders (25.9% → 85.1%) and challenging 45° orientations (11.1% → 100%), while adding only 6.78 ms to the selection stage. MVB-Grasp is not universally better and requires embodiment-specific tuning, but our results show that thin, geometric adapters can substantially improve the practical reliability of powerful but generic grasp generators on constrained, real-world manipulators.

ACKNOWLEDGMENT

This work was supported in part by the NYUAD Center for Artificial Intelligence and Robotics (CAIR), funded by Tamkeen under the NYUAD Research Institute Award CG010.

REFERENCES

[1] R. Newbury *et al.*, "Deep learning approaches to grasp synthesis: A review," 2023.
[2] A. ten Pas *et al.*, "Grasp pose detection in point clouds," 2017.
[3] M. Sundermeyer *et al.*, "Contact-graspnet: Efficient 6-dof grasp generation in cluttered scenes," 2021.
[4] H.-S. Fang *et al.*, "Anygrasp: Robust and efficient grasp perception in spatial and temporal domains," 2023.
[5] A. Murali *et al.*, "Graspgen: A diffusion-based framework for 6-dof grasping with on-generator training," 2025.
[6] I. Lenz *et al.*, "Deep learning for detecting robotic grasps," 2014.
[7] J. Redmon *et al.*, "Real-time grasp detection using convolutional neural networks," 2015.
[8] D. Morrison *et al.*, "Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach," 2018.
[9] J. Mahler *et al.*, "Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics," 2017.
[10] A. Mousavian *et al.*, "6-dof graspnet: Variational grasp generation for object manipulation," 2019.
[11] K. Huebner *et al.*, "Minimum volume bounding box decomposition for shape approximation in robot grasping," in *2008 IEEE International Conference on Robotics and Automation*, 2008, pp. 1628–1633.
[12] S. Geidenstam *et al.*, "Learning of 2d grasping strategies from box-based 3d object approximations," in *Robotics: Science and Systems V*. The MIT Press, 07 2010.
[13] S. Gui *et al.*, "Goalgrasp: Grasping goals in partially occluded scenarios without grasp training," 2025.
[14] S. Lin *et al.*, "Robot grasping based on object shape approximation and lightgbm," *Multimedia Tools and Applications*, vol. 83, pp. 1–17, 06 2023.
[15] V. D. Cong *et al.*, "Improving robotic grasping accuracy through oriented bounding box detection with yolov11-obb," *Heliyon*, vol. 11, no. 12, p. e43512, 2025.
[16] H. Ma *et al.*, "Generalizing 6-dof grasp detection via domain prior knowledge," 2024.
[17] S. Chen *et al.*, "Efficient heatmap-guided 6-dof grasp detection in cluttered scenes," 2024.
[18] G. Jocher *et al.*, "Ultralytics yolov8," 2023.