7 Appendix

7.1 Mask-Based Sample Generation

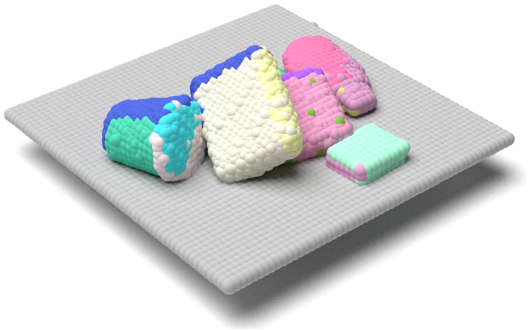


Figure 5. The mask-centric point cloud representation. Each mask is rendered in a distinct color. Points that belong to multiple masks are rendered with only one color.

To obtain object masks on a point cloud, all RGB-D images are first cropped to focus on the workspace area. The Segment Anything Model [44] is then applied to each RGB image to acquire 2D segmentation masks of all objects. Given the pixel-wise correspondence between the RGB image, masks, and depth map, we can map these 2D masks onto the point cloud using the depth information and transformation matrices. Note that each point $\mathbf{p}_i$ may belong to multiple masks, such as $\mathbf{p}_i \in \mathbf{M}_a$ and $\mathbf{p}_i \in \mathbf{M}_b$, where $\mathbf{M}_a$ and $\mathbf{M}_b$ are different masks. By combining the point clouds and mask information from multiple cameras, a raw mask-centric point cloud representation is reconstructed. After preprocessing and filtering, we finally obtain the refined mask-centric point cloud scene, illustrated in Figure. 5.

In our implementation, we use the centers of the individual object masks as center points to construct B_i for generating grasp poses during training data collection. This method provides more even point sampling compared to the FPS-based method, ensuring a more uniform distribution of grasp points. Additionally, it avoids generating many unstable poses on the edges and corners of objects. Although FPS is still used to construct B_i during inference, we found that our model effectively reduces the probability of grasping the edges or corners of objects.

7.2 Model Architecture

The detailed architecture of EquiFormerV2, as mentioned in Section 4.5, is illustrated in Figure. 6. Compared to the original structure, we have made several key modifications. The original SO(3) embedding has been replaced with a single linear layer that takes the point coordinates and normal directions as input directly. This embedding is then used in subsequent blocks. After the first equivariant graph attention block, we apply FPS to downsample the point cloud. For each downsampled point, we use KNN to find its neighbors and build edges between them. During upsampling, we reverse these edges from each downsampling block by swapping the source and destination of these edges. This allows us to gradually transfer information from the downsampled points to the points in the upsampling blocks. The theoretical foundation and mathematical proof of EquiFormerV2 can be found in [17, 36, 37].

Figure. 7 visualizes the procedure for evaluating grasp poses on the *orbit* of point p in the point cloud. Given the point normal vector n_p and the Fourier coefficients $\mathcal{F}_{l,p}^m$ output from the network, the spherical harmonics basis function is first multiplied with the coefficients to reconstruct the spherical harmonics signals on S^2. This corresponds to Equation. 1. These signals can be used as the grasp quality function, i.e., $f_p \colon S^2 \to \mathbb{R}$. The orbit grasp sampler then takes the normal vector as input and outputs a set of approach vectors $\{r_3 \in \overline{O}_p = \{r_3 \in S^1 : n_p^\top r_3 = 0\}\}$. These vectors are used to query f_p by Equation. 1 to yield the grasp quality of each approach vector.

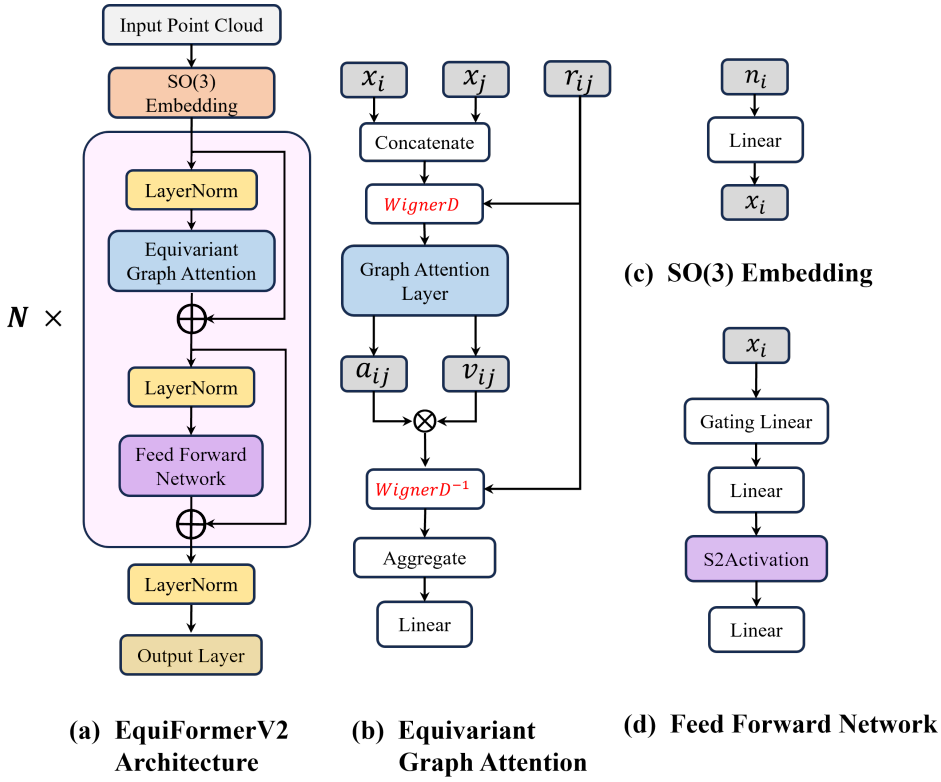


Figure 6. **Overview of the EquiFormerV2 architecture.** (a) shows the overall structure of EquiFormerV2, while (b), (c), and (d) illustrate the submodules of (a). Multiple EquiFormerV2 blocks, incorporating FPS and KNN layers for connectivity, are stacked to form our UNet-style architecture.

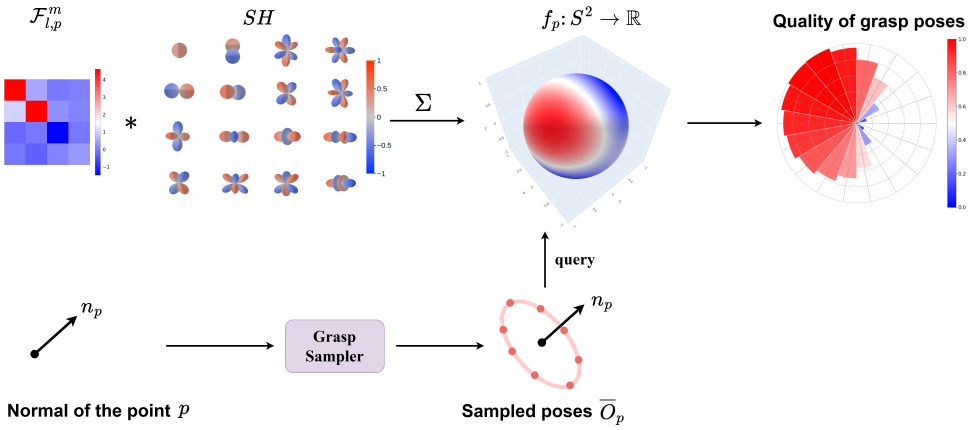
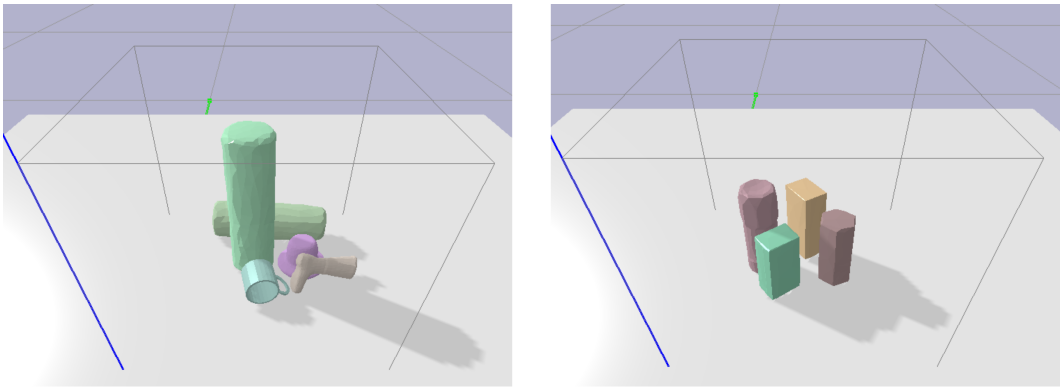


Figure 7. **Visualization of grasp poses sampling and evaluation.** To enhance clarity, this procedure is illustrated using one point p in the point cloud. The Fourier coefficients of p, denoted as $\mathcal{F}_{l,p}^{m}$, and the spherical harmonics basis functions SH are used to reconstruct the grasp quality function $f_p: S^2 \to \mathbb{R}$ on the 2-sphere S^2, as detailed in Equition.1. The circle on the top right, viewed along the normal direction of point p, shows the grasp quality of each sampled pose, with redder colors indicating higher quality.

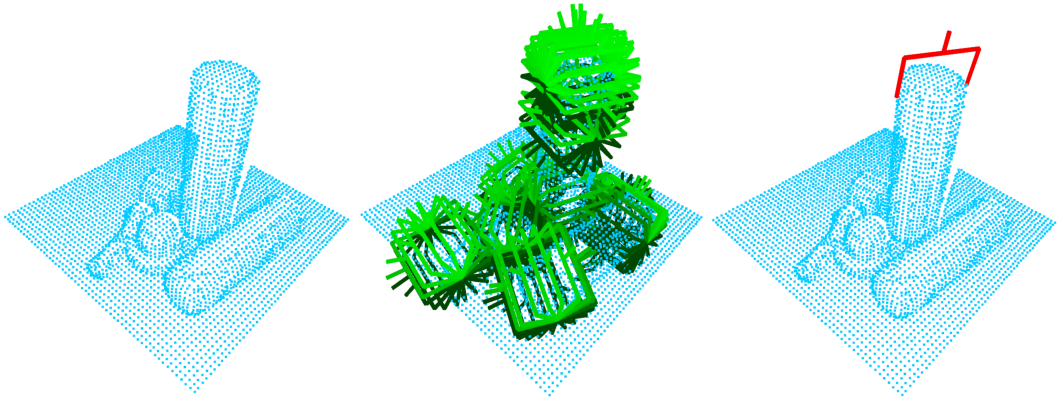
7.3 Simulation Additional Details

We provide several figures (Figure. 8, 9) to give more information about the simulation environment and the grasp pose evaluation process.



(a) (b)

Figure 8. (a) and (b) illustrate examples of "pile" and "packed" scenes, respectively.



(a) (b) (c)

Figure 9. (a) The downsampled point cloud. (b) We show sampled grasp poses at the points with the highest grasp quality scores in each of the 10 $\mathcal{B}_i$. A more intense green color indicates higher quality. For simplicity, only 18 of the 36 sampled poses per point are displayed. (c) The best grasp poses among all $\mathcal{B}_{1,\ldots,n}$.

7.4 Physical Experiments Additional Details

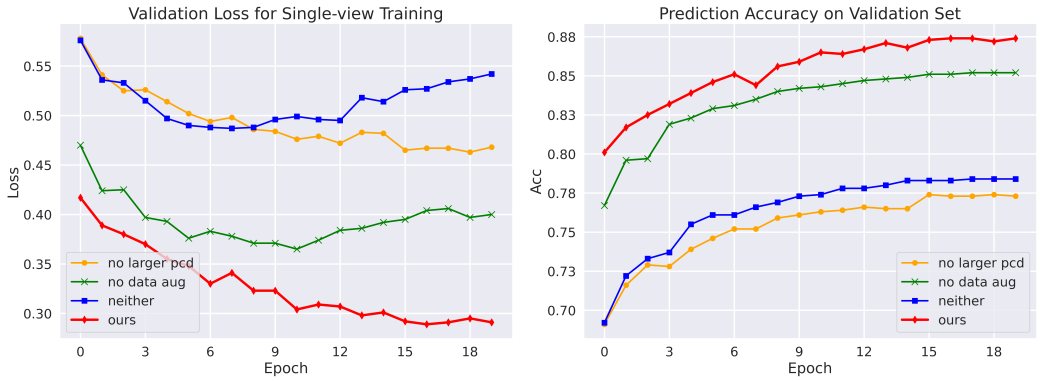
Implementation Details. Although our design initially selects the grasp pose with the highest Z value after filtering, we observed that this highest Z value can sometimes be unstable on the physical robot, unlike in simulations. Therefore, because our method generates a series of grasping poses for each point, we also consider poses within a 3cm range below the highest grasp pose. If a pose within this range has the highest grasp quality (i.e., it is at least better than the highest pose), we select it instead. This approach effectively reduces failures caused by weak grasping and mitigates the sim-to-real gap.

Failure Mode Analysis. In the single-view setting, we observe that the GSR of our method for the ***Packed*** scene is lower than that for the ***Pile*** scene, which is inconsistent with the simulation results. The primary failure mode $(5/8)$ involves a white bottle that blends with the table mat color. This blending results in inaccurate depth and point cloud shape estimation by the camera, preventing

the gripper from being inserted deeply enough to provide sufficient friction. For the ***Pile*** task, the primary reason for failure is the thickness of the objects, where thin objects cause their point cloud to merge with that of the table, distorting the shape of objects. This distortion leads to inaccurate normal estimation and unreasonable poses generated by our model. Additionally, the smooth surface and specific shape of some objects lead to insufficient friction, causing objects like stones to slip out of the gripper's hand. In the multi-view setting, the overall tendency of failure is similar to the single-view setting. While the point clouds from different perspectives help mitigate the distortion problem, introducing more cameras also introduces calibration errors between them. These errors, in turn, transfer to noises that appear in the point cloud.

7.5 Full Ablation Results

Larger Point Cloud as Input and Data Augmentation. As mentioned in Section 4.3, we emphasize the importance of using a larger point cloud $B_i = \mathcal{N}(c_i, m)$ instead of just the local point cloud $\mathcal{B}_i = \mathcal{B}(c_i, r_l)$ is crucial for eliminating boundary effects by providing more context. To evaluate this, we compare the performance of these two input formats. Besides, despite our network being $\mathrm{SE}(3)$-equivariant, overfitting remains a concern. Therefore, we also assess the impact of data augmentation by comparing results with and without random rotations of the point cloud in $\mathrm{SE}(3)$ space before each SGD step. The results, shown in Figure. 10, indicate that using the larger point cloud as input significantly improves performance, as evidenced by lower loss and higher prediction accuracy. Moreover, we also find that without data augmentation, validation loss increases in the later stages of training, regardless of other strategies used. These findings underscore the necessity of larger point clouds for providing context and the importance of data augmentation.



(a). The validation loss trends.

(b). The predicted accuracy of all validation poses.

Figure 10. The ablation study results of larger point cloud as input and data augmentation.

Performance Comparison Between Mask-Based and FPS-Based Training Data. We compared the effects of mask-based versus FPS-based training data. Although both methods perform similarly in simulations, differences appear in real-world experiments. As shown in Figure. 11, with the same input, the grasp quality distribution from the mask-based trained network is more uniform and centered around the object's center of mass (e.g., the banana, hammer, and shoe). This indicates that the mask-based trained network can evaluate a wide range of grasp poses more effectively. In contrast, the FPS-based trained network tends to produce grasps biased towards the object's edges or specific small regions. These edge-focused poses are relatively unstable than those around the center of mass. We interpret this as a result of FPS-based training lacking object-centric awareness, which causes the network to focus on specific areas, and consequently, it struggles to effectively assess grasps at other positions. Mask-based training data, however, incorporates object-centric information. This enables the network to evaluate grasp poses across the entire object. Therefore, even when FPS is used for input during inference, the network maintains enough robustness to handle unseen geometric information.

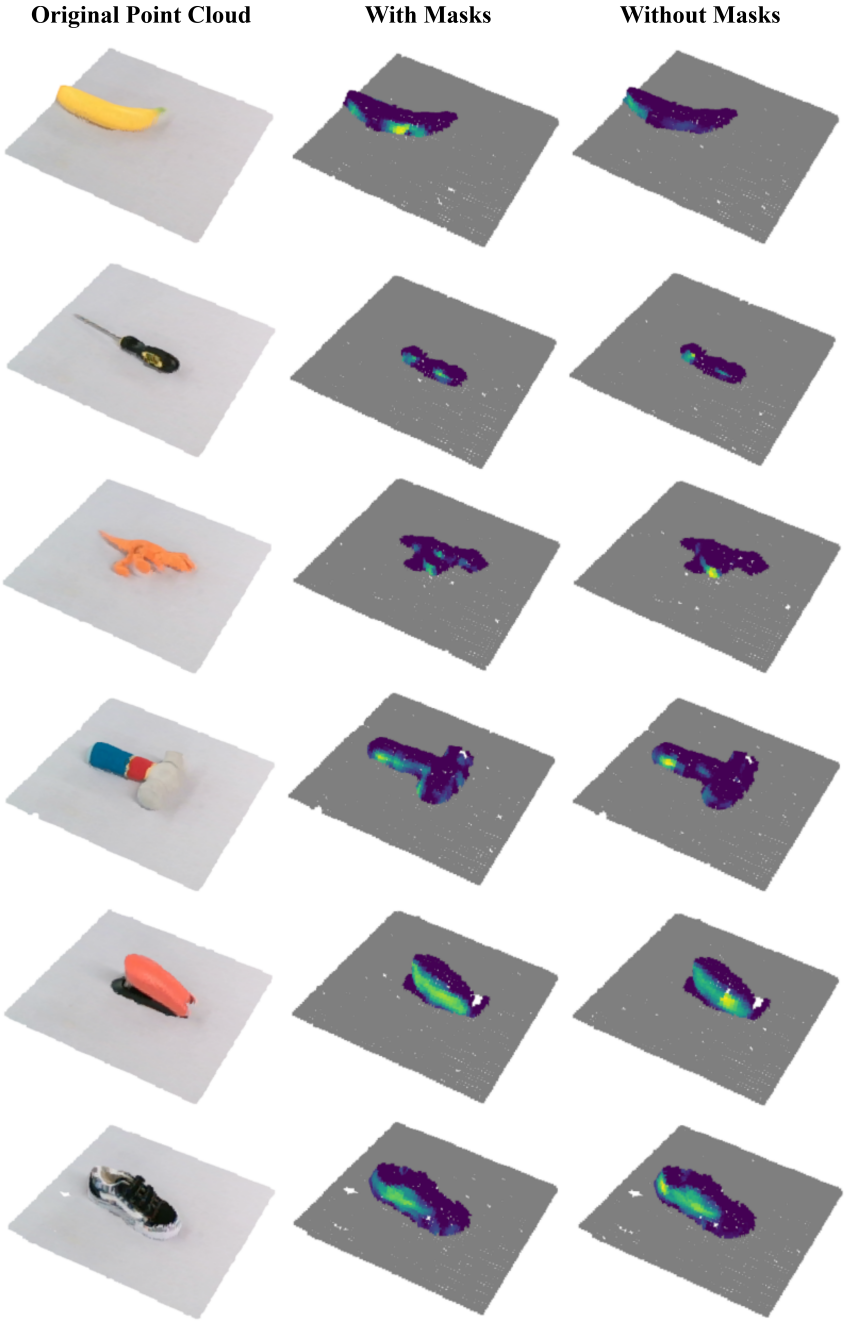


Figure 11. **Grasp quality distribution for different training data generation strategies.** The highest grasp quality of all sampled poses at each point represents that point's quality. The left column displays the original point cloud. The middle column shows predictions from the network trained with mask-based data. The right column shows predictions from the network trained with FPS-based data.