LieGrasPFormer: Point Transformer-based 6-DOF Grasp Detection with Lie Algebra Grasp Representation

Jianjie Lin, Markus Rickert, and Alois Knoll

***Abstract*— With the significant advancements made in the 6-DOF Grasp learning network, grasp selection for unseen objects has garnered much attention. However, most existing approaches rely on complex sequence pipelines to generate potential Grasp, which can be challenging to implement. In this work, we propose an end-to-end grasp detection network that can create a diverse and accurate 6-DOF grasp posture based solely on pure point clouds. We utilize the hierarchical PointNet++ with a skip-connection point transformer encoder block to extract contextual local region point features, which we refer to as LieGrasPFormer. This network can efficiently generate a distribution of 6-DoF parallel-jaw grasps directly from a pure point cloud. Moreover, we introduce two different grasp detection loss functions, which give the neural network the ability to generalize to unseen objects, such as generators. These loss functions also enable a continuously differentiable property for the network. We trained our LieGrasPFormer using the synthesized grasp dataset ACRONYM, which contains 17 million parallel-jaw grasps, and found that it generalized well with an actual scanned YCB dataset consisting of 77 objects. Finally, we conducted experiments in the PyBullet simulator, which showed that our proposed grasp detection network can outperform most state-of-the-art approaches with respect to the grasp success rate.**

I. Introduction

Grasp detection is a critical task in robotic manipulation, particularly in unstructured environments such as warehouses or households. Analytical-based grasp approaches can achieve high grasp quality when provided with an accurate object model [1]. However, they still face difficulties in generalizing to unseen objects, and most approaches are limited to simulations. Data-driven approaches, especially machine-learning tools, can complement analytical-based grasp approaches to significantly reduce the required amount of information of object models. In recent work, grasped objects are represented as RGB images, allowing convolutional-based networks to generate grasp configurations. These works are classified as 3/4-DOF (degree of freedom) grasping and constrain the gripper vertically to the objects (top-down grasp). This type of grasping can dramatically simplify the pick and place task problem. However, the constraints imposed by 3/4-DOF limit the potential for integrating motion planning, where motion planning algorithms explore all possible directions to generate a good motion trajectory. This limitation has led researchers to study a more general approach (6-DOF grasp) [2], [3], [4], [5], enabling grasping in an arbitrary direction.

Jianjie Lin, Markus Rickert, Alois Knoll are with Robotics, Artificial Intelligence and Real-time Systems, School of Computation, Information and Technology, Technische Universität München, Munich, Germany {jianjie.lin, rickert, knoll}@in.tum.de

The proposed approaches in [2], [3] sample the grasp configuration in terms of the region of interest (ROI) and evaluate each sampled grasp configuration separately. However, the entire process is time-consuming, as the number of generated grasp configurations increases. In [5], [6], the variational auto-encoder (VAE) approach with an evaluation network is introduced for generating diverse grasp configurations. This VAE replaces the heuristic geometry sampler in [2], [3], significantly simplifying the generation process. An evaluation network further refines the candidate grasp configuration. We classify such approaches in [5], [6] as a two-stage approach.

To simplify the grasp detection algorithm further, [7], [4], [8] proposed a one-stage strategy that directly outputs the grasp quality and grasp configuration simultaneously. The proposed approaches in [7], [4] consider the grasp detection problem as a one-to-one pose regression, heuristically selecting the ground truth grasp configurations. The coarse-to-fine approach introduced in [8] takes a step further by considering the grasp detection as a one-to-multiple pose regression problem and requires manual quantization of the grasp orientation for refinement.

In this work, we adopt the "one-to-one grasp pose regression" strategy proposed in [7], [4], as it allows for refinement of the grasp orientation outside of the deep learning framework. We assume that there is an infinite number of grasp configurations that can successfully grasp an object and that the similarity between a candidate grasp configuration and the ground truth depends on the distance between the two grasp points. Based on this assumption, we propose a grasp quality loss function that follows a Gaussian distribution.

Moreover, most related work represents the orientation via Euler angles and subsequently converts it to a rotation matrix, which can cause discontinuity. We address this issue by using Lie algebra to represent a transformation matrix, which enables us to integrate the rotation and translation loss functions into one. We also generate diverse grasp configurations by downsampling the point cloud into a predefined number of points, considering each downsampled point as a grasp point, and generating the grasp configuration based on its corresponding grasp point.

Our proposed 6-DOF grasp detection algorithm follows the deep learning framework and is trained with the synthetic ACRONYM dataset [9]. We evaluate the algorithm with the scanned YCB dataset [10], which contains partial point clouds of various objects captured by a commodity depth camera. We do not specify the object category for training,

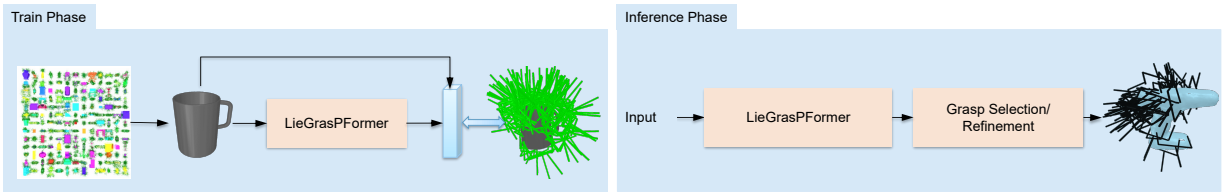


Fig. 1: The visualization of the overall network pipeline. On the left side, one object from ACRONYM [9] is randomly chosen for training the LieGrasPFormer and outputs diverse grasp configurations and grasp quality for regressing. A point cloud is taken for generating the grasping posture at the inference phase on the right side.

instead using the proposed network for the generalization of unseen objects. We refer to the grasp network as LieGrasP-Former for two reasons: first, we modify the hierarchically PointNet++ with a stacked skip-connected point transformer encoder as the backbone, and second, the proposed network can directly output the Lie algebra vector for representing the grasping posture.

During inference, we rank the grasp quality and select the grasp configuration with the highest grasp quality as the final candidate. We also follow the concept of GraspIt! [11] by moving the potential grasp configuration in the approach direction until the gripper contacts the object. The overall structure of the network process is illustrated in Fig. 1.

II. Related Work

Grasp selection or grasp estimation is the process of estimating the grasp posture in the world coordinate system based on input sensor data, which can be in the form of images or point clouds. There are two main categories of grasp selection approaches: analytical-based and data-driven-based. The analytical-based approach [12], [13] relies on precise geometric and physical models of objects, typically available in CAD, which may not be easily obtained in unknown or complex environments. Additionally, surface properties and friction parameters are important in analytical-based approaches and pose additional challenges. Approaches, such as the use of particle filters for surface parameter estimation, have been proposed to address these challenges [14].

In previous work [1], the authors proposed a novel approach that combines Gaussian Process Implicit Surfaces (GPIS) and Bayesian optimization to compute a grasp configuration. This approach can model objects with noise implicitly using GPIS. However, this approach is still limited to simulations and can be error-prone and requiring CAD model.

Deep learning has inspired researchers to utilize neural network structures for grasp detection. There are two main categories of deep learning-based grasp approaches: 2D planar grasp and 6-DOF grasp. In 2D planar grasp, researchers use a five-dimensional vector to represent robotic grasps posture, which are rectangles with position, orientation, and size: (x, y, θ, h, w). In the deep learning framework, there are typically three ways to obtain an oriented rectangle-based grasp configuration: classification-based, regression-based, and detection-based approaches. However, the 2D planar grasp restricts the grasp in the top-down direction.

The 6-DOF grasp provides more possibilities to interact with objects and considers the flexibility of the robotic manipulator in an environment. Grasp Pose Detection (GPD), proposed in [2], samples diverse candidate grasps around the region of interest (ROI). The generated grasp posture is then fed into an adopted CNN, which is classified from the perspective of grasp score. PointNetGPD [3] considers 3D geometry in a different way by taking the point cloud directly as input for PointNet to learn a grasp quality network. The geometry-based grasp sampler is critical for these two approaches.

In [5], an adopted Variational Auto-Encoder (VAE) is proposed to train a grasp sampler for generating diverse sets of grasps, and the gradient of the evaluation network is utilized to refine the generated grasp. Furthermore, an improved version of 6-DOF GraspNet is proposed in [6] by introducing a learned collision checker conditioned on the gripper information and the raw point cloud of the scene. Contact-GraspNet, introduced in [15], reduces the dimensionality of grasp representation to 4-DOF to facilitate the learning process.

In contrast to the two-stage approaches, a single-shot strategy is proposed in [4] to regress the grasp configuration by using PointNet++. With the same concept, PointNet++Grasp is introduced in [7], which can directly predict the poses, categories, and scores of all the grasps. These two approaches consider grasp detection as a regression and classification problem, reducing the diversity of grasp distribution.

III. Algorithm

A. Grasp Problem Formulation

The goal of LieGrasPFormer is to determine a set of possible 6-DOF grasp configurations for picking up an object using a parallel-jaw gripper from any direction, given the object's point cloud in Cartesian world coordinates. The grasp configuration is denoted as $g = [c, \mathbf{T}]$, as shown in Fig.3b. The grasp quality is represented as c, and the grasp pose is defined as a transformation matrix. Most grasp detection networks use a 6-dimensional vector $(x, y, z, r_x, r_y, r_z) \in \mathbb{R}^6$ to represent the grasp pose, whereas in this work, we represent the grasp transformation matrix using Lie algebra $\mathfrak{se}(3) \in \mathbb{R}^6$. Therefore, the grasp configuration in this work is denoted as $g = [c, \mathbf{v}]$, where $\mathbf{v} = [\mathbf{t}^{\mathrm{T}}, \boldsymbol{\omega}^{\mathrm{T}}]^{\mathrm{T}} \in \mathcal{R}^6$ is a 6-dimensional

vector of coordinates in the Lie algebra $\mathfrak{se}(3)$. The $\mathfrak{se}(3)$ vector $\mathbf{v}$ comprises two separate 3-dimensional vectors: $\boldsymbol{\omega}$, which determines the rotation, and $\mathbf{t}$, which determines the translation.

B. Network Structure Design

Similar to most deep learning-based grasp network structures, we utilize PointNet++ [16] as our primary framework to extract point features. To enhance the capability of PointNet++, we modify its set abstraction structure by incorporating a skip-connection point transformer encoder block [17]. The point transformer encoder [18], [19] has been demonstrated to possess a strong ability to attend to important features. Furthermore, the point transformer encoder structure with a skip connection for aggregating the feature can propagate larger gradients to initial layers, allowing for faster learning in deeper networks. The main advantage of using the hierarchically skip-connection transformer encoder is to learn an attention map between the point feature after the furthest point sampling [16], which was previously aggregated only using max pooling, neglecting the geometric relationship within the features. Additionally, the attention mechanism is shown to be permutation-invariant and is therefore useful for point cloud applications. We also follow the idea from [20], which replaces self-attention with offset-attention to enhance the transformer encoder. Offset-attention originates from Graph convolution networks [21], which demonstrate the benefits of using a Laplacian matrix $\mathbf{L} = \mathbf{D} - \mathbf{E}$ to replace the adjacency matrix $\mathbf{E}$, where $\mathbf{D}$ is the diagonal degree matrix. After extracting the network's primary skeleton, two additional branches are deployed to compute grasp quality and grasp configuration separately. These two branches consist of MLP layers for regression. The grasp quality in the ground truth is normalized between $[0, 1]$. We use the sigmoid operator instead of softmax for regression, since softmax is suitable for predicting probabilities. Finally, the network is fed a point cloud of size $\mathbb{R}^{2000\times3}$ and predicts 256 grasp configurations to ensure the inference fits in GPU memory.

C. Grasp Loss Design

We adopt the approach of [4] and treat grasp detection as a regression task for both grasp pose and quality. In Fig. 2, we show how we use two MLPs to predict the grasp quality and configuration simultaneously.

1) Gaussian-based Grasp Quality Loss: We approach the problem of predicting grasp quality as a regression task. In [8], the authors used focal loss to compute the classification loss by assigning each subsampled grasp point with a predefined score, checking whether the grasp point is enclosed by the gripper using the ground truth grasp posture. In contrast, we consider the assignment of grasp quality from a different perspective. We make the assumption that a predicted grasp quality follows a Gaussian distribution. The closer the predicted grasp configuration is to the assigned ground truth grasp configuration, the higher the grasp quality.

To regress the point-wise 6-DoF grasp pose, we assume that each point q in the point set has its best grasp configuration g corresponding to the ground truth. Unlike most existing works that consider the regression unique and fixed, we use the Gaussian-distributed grasp quality to expand the grasp configuration, allowing the predicted point to deviate slightly from the ground truth. This point is denoted as the grasp center g_t, as assumed in most works. We use the k-nearest neighbors (KNN) to find the nearest grasp point inside the ground truth grasp point set for each selected point. We can then compute the ground truth grasp quality at this grasp point, which can be formulated as follows:

$$c_{j,g} = \exp(-\frac{||\mathbf{q}_j - \mathbf{q}_g||^2}{\delta^2}) \tag{1a}$$

$$L_{\text{score}} = \frac{1}{|S|}\sum_j ||c_{j,p} - c_{j,g}||^2\,, \tag{1b}$$

We select a grasp point $\mathbf{q_j}$ and find the corresponding ground truth grasp point $\mathbf{q_g}$ using the KNN. The computed grasp quality at the grasp point $\mathbf{q}_j$ is represented by the score $c_{j,g}$, while the predicted grasp quality from the network is denoted by $c_{j,p}$, the variable $|S|$ is the number of grasp configuration. This process is simplified by using Gaussian distribution quality assignment, which differs from the predefined parameter used in most other works to assign grasp success [5], [6].

2) Lie Algebra Transformation Loss: The widely used L_1 or L_2 metrics for computing the distance between Euler angles have been proven [4] to suffer from discontinuity and ambiguity in deep learning applications. Two common approaches are typically employed to mitigate the discontinuity when learning a rotation loss. The first one is to convert the Euler angle into a rotation matrix and use the Frobenius distance to obtain the loss, such as $||\mathbf{I}_{3\times3} - \mathbf{R}^{\mathrm{T}}\mathbf{R}||_F$. Another approach utilizes normalized quaternions to represent a rotation matrix. Both expressions are still discontinuous. The rotation expression in [4] introduces a 6D representation of the 3D rotation matrix to remedy the problem of discontinuity, but this increases the complexity. Importantly, all approaches mentioned above consider rotation and translation separately. For normal rotation classification, the separation can simplify the loss function. However, in grasp planning, the rotation matrix and translation vector should be coupled. Therefore, we use Lie algebra to overcome the discontinuity and ambiguity issue. We consider the grasped object to be composed of a set of charts, and we define each grasp configuration on one chart, as illustrated in Fig. 3d. Based on the manifold theory, specifically Lie group theory, SE(3) is a continuous group. Furthermore, SE(3) is a differentiable manifold. Additional details on the benefits of using Lie algebra to train an SE(3) neural network can be found in SE(3)-TrackNet [22].

As shown in Section III-A, we use Lie algebra $\mathfrak{se}(3)$ for representing a transformation matrix with $\mathbf{v} = [\mathbf{t}^{\mathrm{T}}, \boldsymbol{\omega}^{\mathrm{T}}]^{\mathrm{T}} \in \mathcal{R}^6$. According to the Lie algebra theory [23], [24], the exponential map, which maps elements from the algebra

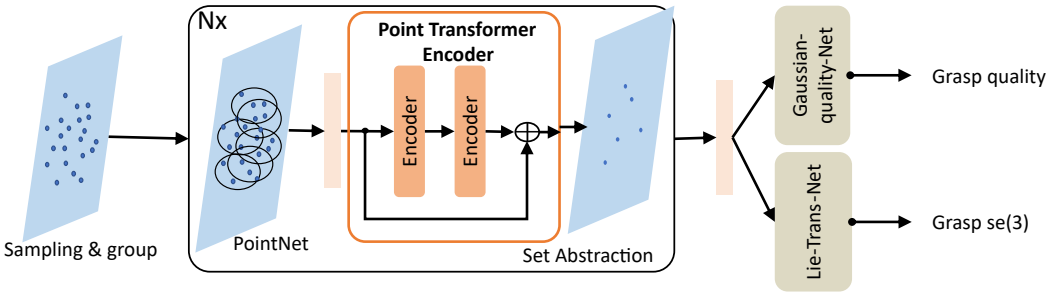


Fig. 2: Demonstration of the overall structure of LieGrasPFormer. The point cloud is first input into a PointNet++ for extracting point features. The set abstraction module then provides these features to a skip-connected point transformers encoder for learning semantic features with the multi-head attention mechanism. The final contextual global feature is then separated into two MLP networks to learn the Lie algebra-based grasp representation and the Gaussian distributed-based grasp quality.

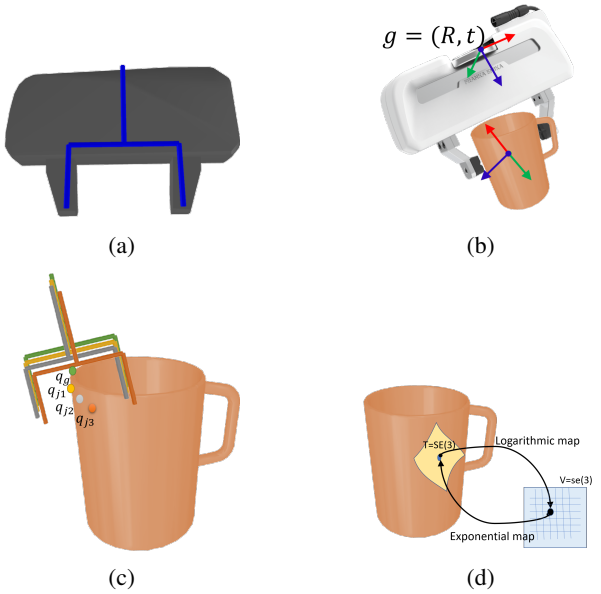


Fig. 3: Visualization of the parallel Franka Emika gripper setting. (a) A simplified parallel gripper (blue line) with its Franka Emika gripper collision mesh. (b) Franka Emika grasp configuration with respect to the mug. (c) A set of translation-shifted grasp configurations at different grasp points with respect to the ground truth grasp posture. (d) A grasp configuration chart defined on the mug with its corresponding Lie algebra map.

to the manifold and determines the local structure of the manifold, is used to express the transformation matrix as

$$e^{\mathbf{v}} = \begin{bmatrix} e^{\boldsymbol{\omega}^{\wedge}}, & \boldsymbol{V}\mathbf{t} \\ \mathbf{0}, & 1 \end{bmatrix}, \tag{2}$$

where $\boldsymbol{\omega}^{\wedge}$ is the skew-symmetric matrix of $\boldsymbol{\omega}$, and

$$e^{\boldsymbol{\omega}^{\wedge}} = \boldsymbol{I} + \frac{\sin\theta}{\theta}\boldsymbol{\omega}^{\wedge} + \frac{1-\cos\theta}{\theta^2}(\boldsymbol{\omega}^{\wedge})^2 \tag{3a}$$

$$\boldsymbol{V} = \boldsymbol{I}_{3\times 3} + \frac{1-\cos\theta}{\theta^2}(\boldsymbol{\omega}^{\wedge}) + \frac{\theta-\sin\theta}{\theta^3}(\boldsymbol{\omega}^{\wedge})^2 \tag{3b}$$

The scale value θ is the norm of $\boldsymbol{\omega}$. By using the exponential map, we can easily convert the 6-dimensional vector coordinate to SE(3). Similarly, using the logarithm map with $\boldsymbol{\omega} = (\mathrm{Log}(\mathbf{R}))^{\vee}$ and $\mathbf{t} = \boldsymbol{V}^{-1}\mathrm{trans}$, we can convert SE(3) to $\mathfrak{se}(3)$. The operator $(\vee)$ converts a skew-symmetric matrix to a vector, which is the reciprocal of the operator $(\wedge)$. In this work, the proposed network produces a Lie algebra $\mathfrak{se}(3)$ value for each selected grasp point, denoted as $\mathbf{v}_{j,\mathfrak{se}(3)}$. Using the same strategy introduced in Section III-C.1, we can obtain the corresponding ground truth grasp configuration, indicated as $\mathbf{T}_{j,g}$. Additionally, we add the translation $\mathbf{q}_j - \mathbf{q}_g$ to $\mathbf{T}_{j,g}$, denoted as $\mathbf{T}_{j,g'}$. (Fig. 3c). The logarithm map and group operation are employed to define the geodesic distance as Euclidean norm by

$$p\left(\mathrm{M}_1, \mathrm{M}_2\right) = \left\|\log\left(\mathrm{M}_1^{-1}\mathrm{M}_2\right)\right\| \tag{4}$$

where $M_i \in$ Lie group, defined on a differentiable manifold. The sum of the squared geodesic distance, is given as

$$L = \sum_{i=1}^{N} \rho^2\left(\boldsymbol{\beta}\left(f_i\right), \mathrm{M}_i\right). \tag{5}$$

The Baker-Campbell-Hausdorff (BCH) formula formulate the logarithm $\log\left(e^{\mathrm{M}_1}e^{\mathrm{M}_2}\right)$ as a Lie algebra element using only Lie bracket for noncommutative Lie groups. A first order approximation to the BCH [25] is

$$\log\left(e^{\mathrm{M}_1}e^{\mathrm{M}_2}\right) = \mathrm{M}_1 + \mathrm{M}_2 + \frac{1}{2}\left[\mathrm{M}_1, \mathrm{M}_2\right] + O\left(\mathrm{M}_1^2, \mathrm{M}_2^2\right) \tag{6}$$

where M_1 and M_2 are $\mathrm{m}_1 = \log\left(\mathrm{M}_1\right)$ and $\mathrm{m}_2 = \log\left(\mathrm{M}_2\right)$, respectively. The geodesic distance can be approximated by [25]

$$\begin{aligned} \rho\left(\mathrm{M}_1, \mathrm{M}_2\right) &= \left\|\log\left(\mathrm{M}_1^{-1}\mathrm{M}_2\right)\right\| = \left\|\log\left(e^{-\mathrm{m}_1}e^{\mathrm{m}_2}\right)\right\| \\ &= \left\|\mathrm{m}_2 - \mathrm{m}_1 + 0.5\left[-\mathrm{m}_1,\ \mathrm{m}_2\right] + O\left(\ \mathrm{m}_1^2,\ \mathrm{m}_2^2\right)\right\| \\ &\approx \left\|\mathrm{m}_2 - \mathrm{m}_1\right\| \end{aligned} \tag{7}$$

The approximation is sufficient for regression as long as the training samples lie in a small neighborhood of the identity.

Therefore, the sum of squared geodesic distance can be represented as follows:

$$L_{\mathrm{se}(3)} \approx \sum_{i=1}^{N} \left\| \log\left(\mathrm{M}_i\right) - \log\left(\beta\left(f_i\right)\right) \right\|^2 = \frac{1}{|S|} \sum_{0}^{j} \left\| \mathbf{v}_{j,\mathfrak{se}(3)} - \mathrm{Log}(\mathbf{T}_{j,g'}) \right\|^2 \tag{8}$$

It can be easily proven that if $\mathbf{v}_{j,\mathfrak{se}(3)}$ is equal to $\mathrm{Log}(\mathbf{T}_{j,g'})$, the predicted grasp configuration is aligned to ground truth.

3) Grasp Planning Loss: Finally, we combine the grasp quality and transformation loss using weighting coefficients to create a grasp loss

$$L_{\mathrm{total}} = w_1 L_{\mathrm{score}} + w_2 L_{\mathfrak{se}(3)}\,. \tag{9}$$

D. Grasp Configuration Selection and Refinement

LieGrasPFormer generates various grasp configurations during the inference phase by feeding a point cloud obtained via a sensor. In order to eliminate all infeasible grasps, we perform collision checking [26]. We then arrange the grasp qualities in descending order, as higher quality grasps provide more reliable grasping of the object. Finally, we select the grasp configuration with the highest grasp quality. Our grasp configuration is independent of the parallel gripper used. Therefore, during the inference phase, we need to adjust the grasp configuration based on the chosen gripper setting. In this work, we use the Franka Emika Panda gripper to evaluate the grasp. The simplified version of the gripper is shown in blue in Fig. 3a, while the collision mesh is shown in gray. As stated in [9], the predicted grasp configuration is known as the pre-grasp, which can result in a stable grasp when the fingers are closed. To improve the grasp configuration, we can further refine it based on the pre-grasp by approaching the gripper in the direction of the approach direction. The grasp approach direction is defined as the direction along the gripper handle, as illustrated in Fig. 6. The approaching process ends when the gripper makes contact with the object. For more information about this approach strategy, please refer to Graspit! platform [11].

IV. Experimental Results

A. Experimental Settings

1) Grasping Data Set: We use the large-scale grasp dataset ACRONYM [9], [15] for training and evaluate it using the YCB dataset [10]. ACRONYM is a grasp dataset for robot grasp planning based on the physics simulation FleX, containing 17.7 million parallel-jaw grasps spanning 8,872 objects from 262 different categories, each labeled with the grasp result obtained from a physics simulator. The objects in the ACRONYM dataset [9] are from ShapeNetSem [27] and are assumed to have uniform density and identical friction. We extend the ACRONYM dataset by assigning each grasp configuration with a deterministic grasp point. In contrast to most state-of-the-art approaches that train and evaluate the network on the same dataset, we evaluate the network using the YCB dataset. The YCB dataset includes objects from daily life with different shapes, sizes, textures, weights, and rigidities, aiming to evaluate the generalization of the proposed grasp detection network.

2) Grasping Tasks: The experimental environment is set up inside PyBullet [28], which includes a Franka Emika robotic manipulator, a Franka Emika parallel gripper, a table, objects to grasp, and a depth camera. To ensure the reachability of the robotic manipulator, we randomly place a single YCB object on the table. We use a depth camera to extract the point cloud by applying the projection and view matrices. The experiments are performed in two steps. First, we infer the grasp configuration by feeding the point cloud, and then we apply inverse kinematics to obtain the joint values for the robotic manipulator. Finally, we close the fingers to grasp the object.

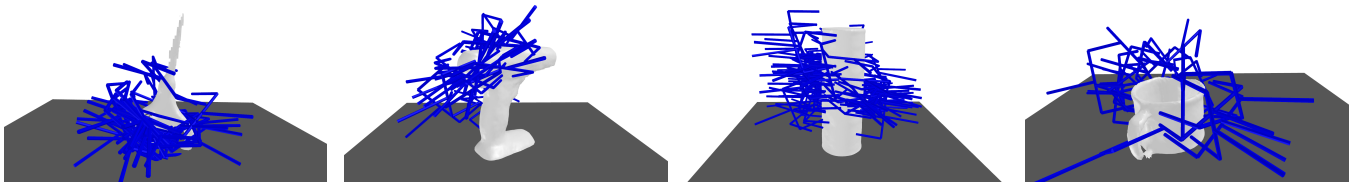
3) Baselines: We compare our approach to two state-of-the-art, open-sourced baselines: GPD [2] and PointNetGPD [3]. GPD uses a geometry sampler to evaluate many grasp candidates, while PointNetGPD replaces the geometry sampler with PointNet++ to learn a grasp quality network. We adopt the default network settings from the original papers.

B. Quantitative Results

We present the success rate results of our grasp generation method in Table I, which indicates the percentage of successful grasps for single-object grasping. The experiments involve randomly translating and rotating each object in the z orientation. We use a range of objects with varying sizes, from big components like the pudding box to small objects like the plum. We trained our network using the ACRONYM data set and evaluated it with the entire YCB data set. We have summarized some of the results in Table I. Based on these results, we can conclude that our approach can be applied to previously unseen objects that are not included in the ACRONYM data set.

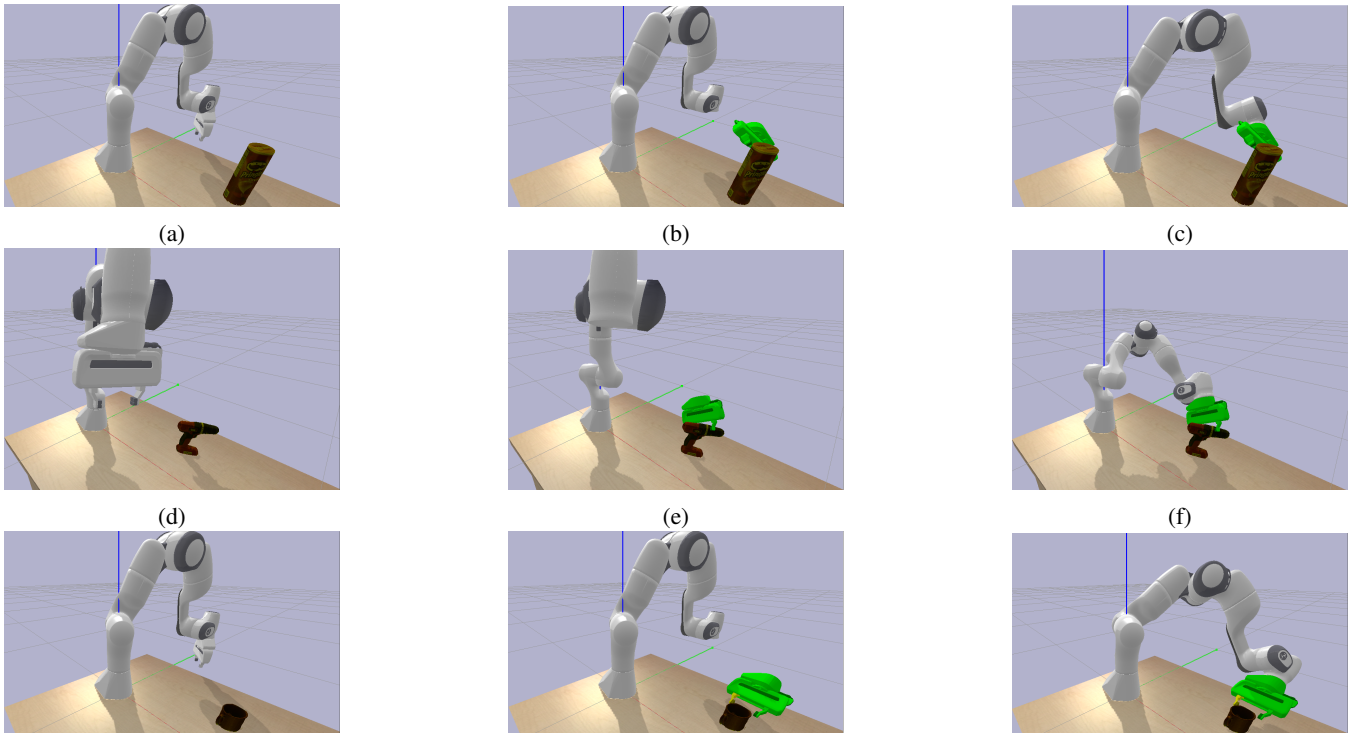
C. Qualitative Results

In this section, we present the results obtained from our proposed LieGrasPFormer. The results are illustrated qualitatively in Fig.4. LieGrasPFormer generates 256 grasp configurations and their corresponding grasp qualities, where the number 256 is a hyperparameter that can be changed based on the available GPU memory. Subsequently, the generated grasps undergo collision checking, and the grasp configurations that are collision-free with high grasp quality are selected for grasp refinement. In the grasp refinement phase, the object is assumed to be placed on a table. We also demonstrate the results in a PyBullet environment (Fig. 5) with different initial robot joint configurations. The execution of a grasp configuration involves two steps. In the first step, we select the collision-free grasp configuration with the highest grasp quality from our LieGrasPFormer. Then, an inverse kinematics solver is applied to obtain the joint configuration. Based on the qualitative results shown in Fig. 4 and 5, we can conclude that our proposed LieGrasPFormer can generate diverse grasp configurations to satisfy various initial grasp configurations, enabling the application of motion planning.



(a) (b) (c) (d)

Fig. 4: Visualization of results from the network after collision checking. The blue lines are the grasp configurations generated by the LieGrasPFormer. The meshes are some examples from the YCB data set.



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Fig. 5: The PyBullet simulation platform is used to simulate grasping objects, which involves two steps. First, LieGrasPFormer generates a grasp configuration at the palm with respect to the point cloud. Then, the end effector of a robotic manipulator is aligned with the palm pose of the grasp configuration. An inverse kinematics solver is used to compute the joint configuration, and finally, a point-to-point motion is executed to grasp the object. The first column demonstrates the initial setting in the scenario. The second column shows the chosen grasp configuration. The last column demonstrates the solution obtained by using inverse kinematics to approach the object.

TABLE I: Results (in %) of a single object grasping experiment. Success is defined as at least one feasible grasp configuration for each object. Our LieGrasPFormer can outperform other state-of-the-art approaches.

Method	Avg.	tomato soup can	pudding box	potted meat can	orange	plum	scissors	e-cups
GPD [2]	86.3	81.6	92.9	78.8	74.8	92.6	97.7	85.8
PointNetGPD [3]	86.0	75.0	94.1	68.6	80.1	94.3	98.4	90.8
LieGrasPFormer (Ours)	94.8	90.0	100.0	96.3	92.3	98.0	100.0	96.8

TABLE II: Ablation study of skip-connection transformer encoder in LieGrasPFormer (in %).

Method	Avg.	tomato soup can	pudding box	potted meat can	orange	plum	scissors	e-cups
LieGrasPFormer (without transformer)	92.7	85.3	100.0	92.3	85.3	93.0	100.0	92.8
LieGrasPFormer	94.8	90.0	100.0	96.3	92.3	98.0	100.0	96.8

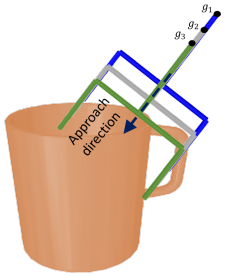


Fig. 6: Depicting the refinement process of our proposed grasping algorithm. The generated grasp pose is visualized as it moves in the direction of the gripper approach until it meets the predefined contact condition, resulting in a successful grasp.

D. Ablation Study

The skip-connection point transformer encoder block is integrated into the PointNet++ structure for extracting more attractive features for grasping. This section aims to study the effect of the transformer encoder block by using pure PointNet++ as the main skeleton. We compare the results in Table II. The results show that the introduced modification of PointNet++ can help the LieGrasPFormer improve performance by extracting more valuable features.

V. Conclusion

This work proposes a new grasp detection neural network based on PointNet++ with a hierarchically skip-connection transformer encoder. Two different grasp loss functions, a Gaussian distribution-based quality score loss and a grasp Transform Loss in terms of Lie algebra, are introduced for remedying the discontinuity problem due to rotation loss in form of Euler angles and quality score loss in form of an indicator function. Our network is trained with the synthetic data set ACRONYM [9] and also works well in the real-world YCB data set. Furthermore, the experimental results show that our framework can detect diverse grasps with a higher convergence on the ground truth grasps and that it can generalize to unknown objects, as the YCB data set [10] is different from ACRONYM [9].

References

[1] J. Lin, M. Rickert, and A. Knoll, "Grasp planning for flexible production with small lot sizes using Gaussian process implicit surfaces and Bayesian optimization," in *Proceedings of the IEEE International Conference on Automation Science and Engineering*, 2021.

[2] A. ten Pas, M. Gualtieri, K. Saenko, and R. Platt, "Grasp pose detection in point clouds," *The International Journal of Robotics Research*, 2017.

[3] H. Liang, X. Ma, S. Li, M. Görner, S. Tang, B. Fang, F. Sun, and J. Zhang, "PointNetGPD: Detecting grasp configurations from point sets," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2019.

[4] Y. Qin, R. Chen, H. Zhu, M. Song, J. Xu, and H. Su, "S4G: Amodal single-view single-shot SE(3) grasp detection in cluttered scenes," in *Proceedings of the Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, L. P. Kaelbling, D. Kragic, and K. Sugiura, Eds., 2020.

[5] A. Mousavian, C. Eppner, and D. Fox, "6-DOF GraspNet: Variational grasp generation for object manipulation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019.

[6] A. Murali, A. Mousavian, C. Eppner, C. Paxton, and D. Fox, "6-DOF grasping for target-driven object manipulation in clutter," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2020.

[7] P. Ni, W. Zhang, X. Zhu, and Q. Cao, "PointNet++ grasping: Learning an end-to-end spatial grasp generation algorithm from sparse point clouds," arXiv:2003.09644 [cs.RO], 2020.

[8] K.-Y. Jeng, Y.-C. Liu, Z. Y. Liu, J.-W. Wang, Y.-L. Chang, H.-T. Su, and W. H. Hsu, "GDN: A coarse-to-fine (C2F) representation for end-to-end 6-DoF grasp detection," arXiv:2010.10695 [cs.RO], 2020.

[9] C. Eppner, A. Mousavian, and D. Fox, "ACRONYM: A large-scale grasp dataset based on simulation," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2021.

[10] B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar, "Benchmarking in manipulation research: Using the Yale-CMU-Berkeley object and model set," *IEEE Robotics Automation Magazine*, vol. 22, no. 3, 2015.

[11] A. Miller and P. Allen, "Graspit! a versatile simulator for robotic grasping," *IEEE Robotics & Automation Magazine*, 2004.

[12] V.-D. Nguyen, "Constructing force-closure grasps," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 1986.

[13] A. Bicchi and V. Kumar, "Robotic grasping and contact: A review," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2000.

[14] L. Zhang and J. C. Trinkle, "The application of particle filtering to grasping acquisition with visual occlusion and tactile sensing," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2012.

[15] M. Sundermeyer, A. Mousavian, R. Triebel, and D. Fox, "Contact-GraspNet: Efficient 6-DoF grasp generation in cluttered scenes," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2021.

[16] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Proceedings of the International Conference on Neural Information Processing Systems*, 2017.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of the International Conference on Neural Information Processing Systems*, 2017.

[18] J. Lin, M. Rickert, A. Perzylo, and A. Knoll, "PCTMA-Net: Point cloud transformer with morphing atlas-based point generation network for dense point cloud completion," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2021.

[19] H. Zhao, L. Jiang, J. Jia, P. Torr, and V. Koltun, "Point transformer," arXiv:2012.09164 [cs.CV], 2020.

[20] M.-H. Guo, J.-X. Cai, Z.-N. Liu, T.-J. Mu, R. R. Martin, and S.-M. Hu, "PCT: Point cloud transformer," *Computational Visual Media*, vol. 7, 2021.

[21] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," in *Proceedings of the International Conference on Learning Representations*, 2014.

[22] B. Wen, C. Mitash, B. Ren, and K. E. Bekris, "se(3)-TrackNet: Data-driven 6D pose tracking by calibrating image residuals in synthetic domains," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020.

[23] J. M. Selig, "Lie groups and lie algebras in robotics," in *Computational Noncommutative Algebra and Applications*, ser. NATO Science Series II: Mathematics, Physics and Chemistry, J. Byrnes, Ed. Springer, 2004, vol. 136, pp. 101–125.

[24] J. L. Blanco Claraco, "A tutorial on SE(3) transformation parameterizations and on-manifold optimization," University of Malaga, Tech. Rep. 012010, Sept. 2010.

[25] F. M. Porikli, "Regression on lie groups and its application to affine motion tracking," 2016.

[26] J. Pan, S. Chitta, and D. Manocha, "FCL: A general purpose library for collision and proximity queries," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2012.

[27] M. Savva, A. X. Chang, and P. Hanrahan, "Semantically-enriched 3D models for common-sense knowledge," in *Proceedings of the CVPR Workshop on Functionality, Physics, Intentionality and Causality*, 2015.

[28] E. Coumans and Y. Bai, "PyBullet, a Python module for physics simulation for games, robotics and machine learning," http://pybullet.org, 2016–2021.