Geometry Matching for Multi-Embodiment Grasping

Anonymous Author(s)
Affiliation
Address
email

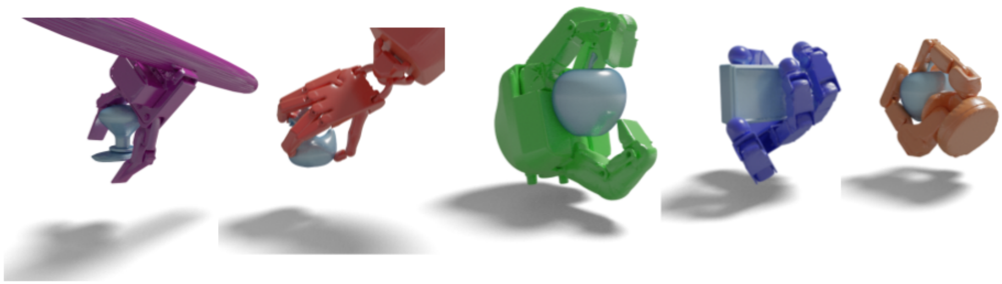


Figure 1: **GeoMatch**: Our method enables multi-emobdiment grasping by conditioning the grasp selection on end-effector and object geometry.

Abstract: While significant progress has been made on the problem of generating grasps, many existing learning-based approaches still concentrate on a single embodiment, provide limited generalization to higher DoF end-effectors and cannot capture a diverse set of grasp modes. In this paper, we tackle the problem of grasping multi-embodiments through the viewpoint of learning rich geometric representations for both objects and end-effectors using Graph Neural Networks (GNN). Our novel method - *GeoMatch* - applies supervised learning on grasping data from multiple embodiments, learning end-to-end contact point likelihood maps as well as conditional autoregressive prediction of grasps keypoint-by-keypoint. We compare our method against 3 baselines that provide multi-embodiment support. Our approach performs better across 3 end-effectors, while also providing competitive diversity of grasps. Examples can be found at geo-match.github.io.

Keywords: Multi-Embodiment, Dexterous Grasping, Graph Neural Networks

1 Introduction

Dexterous grasping remains an open and important problem for robotics manipulation. Many tasks where robots are involved, from the simplest to the most complex ones, at their core come down to some form of interacting with objects in their environment. This in turn, results in grasping objects with all kinds of different geometries. In addition, the large variety of robot and end-effector types necessitates that grasping should also be achievable with new and arbitrary end-effector geometries. However, the cross-embodiment gap between grippers does not permit simply applying grasping policies from one end-effector to another, while domain adaptation i.e. "translating" actions from one embodiment to another, is also not straightforward. In comparison, humans are extremely versatile: they can adapt the way they grasp objects based on what they know about object geometry even if the object class or instance is new to them, and they can do this in more than one ways efficiently.

There has been much research in grasping thus far, with many works focusing on one embodiment at a time [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11] and fewer looking at the multi-embodiment problem [12, 13, 14]. Methods are divided between hand-agnostic or hand-aware, and experiment with different representations for grasping, such as contact maps [12], contact points [13] or even root pose and z-offset [14]. Existing multi-embodiment approaches either require explicit representation of joint limits that becomes exponentially harder in higher DoF end-effectors, or expect heavy manual work to adapt to new end-effectors, or showcase mixed rates of success across different embodiments with some gripper/hand morphologies performing significantly better than others.

Drawing inspiration from how humans seem to be adapt their grasps easily and successfully based on priors they have learned about 3D geometry of both objects in space and their own hands, we propose endowing robotic agents with a similar sense of geometry via a generalized geometry embedding that is able to represent both objects and end-effectors. This embedding can be used to predict grasps that demonstrate stability, diversity, generalizability, and robustness. More specifically, we propose to rely on Graph Neural Networks (GNN) to encode meaningful geometry representations and use them to predict keypoint contacts on the object surface in an autoregressive manner.

In summary, our contributions are as follows:

a) We propose formulating robot grasp learning as a geometry matching problem through learning correspondences between geometry features, where both end-effector and object geometries are encoded in rich embedding spaces.

b) To solve the above problem formulation, we introduce a novel method, namely GeoMatch, that is trained end-to-end to learn expressive geometric embeddings, and autoregressive keypoint contacts via teacher forcing.

c) We demonstrate that our method is competitive against baselines without any extra requirements to support higher DoF end-effectors and while also showcasing high performance across multiple embodiments.

2 Related Work

Dexterous Grasping. Many dexterous grasping works do not look into multi-embodiment and instead focus on diversity of objects using a single end-effector. Many grasping methods support 2-finger parallel grippers [2, 3, 4, 5, 6, 7, 8, 15] with several others looking into high-DoF dexterous manipulation [9, 10, 1, 16]. Some work has also been conducted towards multi-embodiment grasping. Several of those address the problem from the differentiable simulation grasp synthesis point of view [17, 11, 18]. GenDexGrasp [12] advocate for hand-agnostic contact maps generated by a trained cVAE [19] with a newly introduced align distance, and optimize matching of end-effectors against produced contact maps via a specialized Adam optimization method. This is the most recent work to our knowledge, attempting to tackle multi-embodiment grasping without extra steps to support higher DoF grippers. In contrast to them, we choose to operate on hand-specific contact maps as we are interested in learning both object and embodiment geometry conditioned grasps, and empirically found our method to perform more evenly well multi-embodiments. Intuitively, our work is closest to UniGrasp [13]. UniGrasp operates on object and end-effector point clouds to extract features and ultimately output contact points which are then fed into an Inverse Kinematics (IK) solver, similarly to us. Their encoder of choice is PointNet++ and the contact prediction is done through a Point Set Selection Network (PSSN) [20]. Their proposed architecture adds one stage per finger, which means supporting more than 3 finger grippers requires manually adding another stage. As a result, adapting the method to more than 2-finger and 3-finger grippers requires significant work, while also the need for explicit representation of boundary configurations can explode exponentially on higher DoF end-effectors. In contrast, we rely on learned geometry features to identify viable configurations as opposed to explicitly encoding them through joint limit representation, as well as on a small number of user-selected keypoints, same for all end-effectors, which disentangles the dependency between number of fingers and applicability of our method. Similarly to UniGrasp, EfficientGrasp [21] also uses PointNet++ and a PSSN model for contact point prediction and further generates a pose with RL. TAX-Pose [22] is another recent work that shares some high level concepts. Instead of encoding the end-effector, authors look at the problem of tasks involving objects that interact with each other in a particular way. They proceed with encoder objects or object parts using DGCNN and learn a cross-attention model that predicts relative poses of objects that accomplish a task. AdaGrasp [14] uses 3D Convolutional Neural Networks to learn a scoring function for possible generated grasps, and finally executes the best one. Many of the methods mentioned, rely on deterministic solvers which can result in decreased diversity of generated grasps. Even though we also rely on a deterministic solver, we address this issue by leveraging the scoring we obtain by

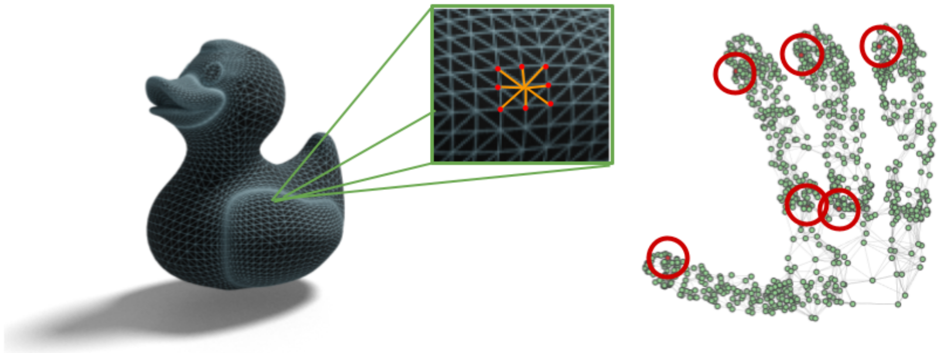


Figure 2: **Object and end-effector inputs.** Objects are initially represented as regularly sampled point clouds which are converted into a graph for further processing. End-effector geometries are given as meshes and converted to coarser graphs by randomly sampling points from the mesh as an intermediate step. User-selected keypoints are highlighted in red.

the learned full unnormalized distribution of contacts to select a first keypoint that will guide the remaining contact point prediction. This permits higher diversity without having to sample a large number of grasps.

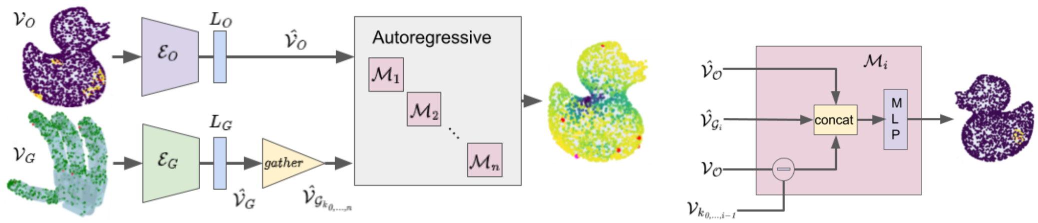
Graph Neural Networks. Graph Neural Networks were first introduced by Scarselli et al. [23] as a proposed framework to operate on structured graph data. Since then, many advancements have been made towards extending their capabilities and expressivity [24]. Specifically in the grasping literature, there have also been multiple instances of use of GNN. More specifically, Huang et al. [25] propose learning a GNN to predict 3D stress and deformation fields based on finite element method based grasp simulations. The use of GNNs for end-effector parameterization has been proposed before in [26] where tactile sensor data is fed into a GNN to represent the end-effector as part of grasp stability prediction, however we propose applying GNN as a more general geometry representation that encompases both objects and end-effectors jointly. Lou et al. [27] leverage GNN to represent the spacial relation between objects in a scene and suggest optimal 6-DoF grasping poses. Unlike previous methods, we aim to use GNN as a general geometry representation for any rigid body, including both objects and end-effectors. For the purposes of this work, we leverage the GNN implementation by [28] due to the readily available and easily adaptable code base.

Geometry-Aware Grasping. In the topic of geometry-aware grasping, several works have advocated for the importance of geometry in the grasping problem. Yan et al. [29] encodes RGBD input via generative 3D shape modeling and 3D reconstruction, then based on this learned geometry-aware representation grasping outcomes are predicted with solutions coming out of an analysis-by-synthesis optimization. In the same vein, Van et al. [30] proposed leveraging learned 3D reconstruction as a means of understanding geometry, and further rely on this for grasp success classification as an auxiliary objective function for grasp optimization and boundary condition checking. Bohg et al. [31] introduced a supervised learning method where a classifier trained on labeled images predicts grasps via shape context based representations. Finally, Jiang et al. [6] learn grasp affordances and 3D reconstruction as an auxiliary task, through the use of implicit functions. Unlike these works, we suggest looking at geometry itself directly from 3D as a feature representation without imposing any 3D reconstruction constraints.

3 Method

In this work, we aim to learn robust and performant grasping prediction via embeddings of geometry for both objects and end-effectors. We are given point cloud representations of object and end-effector geometries. These are converted into graphs which allows to utilize GNNs to learn features across both.

Assume an object geometry represented as a graph $\mathcal{G}_O = (\mathcal{V}_O, \mathcal{E}_O)$ and an end-effector geometry also represented as a graph $\mathcal{G}_G = (\mathcal{V}_G, \mathcal{E}_G)$ where $\mathcal{V}_O$, $\mathcal{V}_G$, $\mathcal{E}_O$, $\mathcal{E}_G$ are the object and end-effector vertices and edges respectively. The edges are represented by adjacency matrices Adj_O, Adj_G for the object and end-effector graphs respectively, which are row normalized symmetric binary matrices



(a) Full overview of GeoMatch. (b) Autoregressive modules.

Figure 3: **GeoMatch architecture.** The object and gripper graphs are passed through the two encoders followed by linear layers. The gripper keypoint embeddings are gathered and are passed as input along with the object embeddings in the autoregressive modules.

with a unitary diagonal. Given $\mathcal{G}_O$ and $\mathcal{G}_G$, we seek to learn feasible and stable contact points between a subset of $\mathcal{V}_O$ and $\mathcal{V}_G$.

3.1 Object and End-effector Representations

Each end-effector is represented by its surface geometry in the form of a graph. Additionally, we require a small number of canonical user-selected keypoints that will be the ones matched with object vertices when calculating contacts. We select these once for each end-effector we are working with visually, and store them. It is recommended to selected keypoints having good coverage of each gripper with respect to its morphology and its grasping behavior. To construct the graph, we sample a point cloud of 1000 surface points from the end-effector mesh in a canonical rest pose:

$$q_{\text{rest}} = (t, R, \theta_0, ..., \theta_{N-1})_{\text{rest}}, \tag{1}$$

where $t \in \mathbb{R}^3$ is the root translation, $R \in \mathbb{R}^6$ is the root rotation in continuous 6D representation as introduced in [32], and $\theta_0, \ldots, \theta_{N-1}$ are the joint angles of the end-effector. We chose the rest pose to be a vector with all joint angles in middle range of their respective joint limits, zero root translation, and identity root rotation. For creating the graph, we consider each of the points a vertex and create edges between each point and its K closest points.

For our experiments, we empirically chose $K = 8$ to capture local geometry. This is a hyperparameter that depends on point cloud density and object structure. The canonical keypoints were selected manually at the rest pose of each hand to represent points of contact. We empirically chose 6 so that for all end-effectors in our dataset, each finger and the palm is represented by at least one keypoint. Choosing the same number of keypoints for different embodiments is technically not a requirement as lower degree of freedom end-effectors may have good enough coverage with less, but we chose to use a constant number multi-embodiments to simplify our training process. A sample of the object and end-effector representations is shown in Fig. 2.

Each object is also represented by its surface geometry in the form of a graph. More specifically, the same process is utilized to convert a object point cloud of 2048 points to a graph. The point cloud and adjacency matrix together describe the graph. For the purposes of this work, we use a subset of the MultiDex dataset introduced by [12] and used by them to train the CMap-VAE model of their approach. The dataset is comprised of 5 end-effectors - one 2-finger, two 3-finger, one 4-finger, and one 5-finger, as well as 58 common objects from YCB [33] and ContactDB [34]. It contains 50,802 diverse grasps over the set of hands and objects, each represented by an object name, an end-effector name, and the end-effector pre-grasp pose in the form of Eq. 1.

3.2 Learning Setup

At the core of our hypothesis is that learning rich geometry features for objects and end-effectors jointly can be a powerful tool for dexterous and diverse grasp prediction multi-embodiments. Thus, we seek an architecture that can embed local geometry information well. From the architecture choices that demonstrate such properties, we chose Graph Neural Neural Networks (GNN) [28].

Our overall model architecture can be seen in Fig. 3a and is designed to learn a) an independent unnormalized prior distribution of contacts between each object and gripper keypoint, and b) marginal distributions of contact for each keypoint, conditioned upon the above likelihood map and previously predicted keypoint contacts.

3.2.1 Independent unnormalized priors of object-keypoint contacts

Ideally, what we would like to calculate is the full joint distribution of vertex-to-vertex contact, i.e. $P(v_i, v_j)$ for all vertices of the object v_i, and all vertices of the end-effector v_j. This is typically intractable. We reduce the complexity by only focusing on n landmark keypoints $k_0, \dots, k_n$ on the end-effector and we try to approximate $P(v_o, k_0, ..., k_n)$ through learning a set of factorizations by applying the Bayes rule. This yields

$$P(v_o, k_0, ..., k_n) = \Pi_{i=1}^{n} P_{M_i}(v_o, k_i | k_0, ..., k_{i-1}) = \Pi_{i=1}^{n} P_{M_i}(v_o, k_i | \mathbf{k}_{<i})), \tag{2}$$

where $v_o \in \mathcal{V}_{\mathcal{O}}$, $(k_0, ..., k_n) \subset \mathcal{V}_{\mathcal{G}}$, and $P_{M_i}(v_o, k_i | k_0, ..., k_{i-1})$ are the factorized marginals to be learned in an autoregressive manner, as discussed in the following subsection. As a first step, we aim to associate a likelihood of contact for a sparse set of keypoints k_i per each object vertex v_o. We first pass the object and end-effector graphs through GNN encoders that output the same number of features. The embeddings obtained are L2 normalized. We then gather the embeddings on the canonical user-selected keypoints as the vertices of interest on the hand. It is noted that we still compute the embedding for all hand vertices even though for contact areas, we focus on the embedding of the canonical keypoints. This unnormalized likelihood map of object-keypoint contacts intuitively represents a score that a given object vertex is in contact with a given gripper keypoint and is given by

$$P_{I_i}(v_o, k_i) = E_O(v_o) \cdot E_G(v_g)[k_i]. \tag{3}$$

This is optimized against the dot product of the hand-specific object contact map $C_O(v_o, k_i)$ via a binary cross-entropy loss

$$\mathcal{L}_{P_{I_{0,\dots,n}}} = \Sigma_{i=1}^{n} \mathrm{BCE}_{\lambda_a}(P_{I_i}(v_o, k_i), C_O(v_o, k_i)), \tag{4}$$

where λ_a is the positive weight hyperparameter used to address the class imbalance.

3.2.2 Autoregressive marginals with teacher forcing

As discussed in the previous paragraph, we seek to estimate the joint distribution of contacts by estimating a set of factorizations. We further proceed with the estimation of factors:

$$P_{M_i}(v_o, k_i | k_0, ..., k_{i-1}) \quad \forall i \in [0, n). \tag{5}$$

Both, object and gripper embeddings are projected down to a lower dimension with a simple linear layer without bias, and passed into 5 layers, each responsible for predicting the index of the object vertex v_{o_n} where keypoint k_n makes contact, given keypoints $k_{0\dots(n-1)}$.

Each layer n concatenates the embedding of the n-th keypoint of the end-effector along with the object embedding. Then, it calculates the *relative* distance map of each object vertex to each of the $n-1$ object vertices where the previous $n-1$ keypoints make contact. Note that is done via teacher forcing: instead of using the predictions of each $n-1$ layer, we use the previous $n-1$ ground truth contact points. This avoids error propagation during training. The relative distance maps are stacked and concatenated with the object and n-th keypoint embeddings. This constitutes the input to an MLP that predicts a binary classification prediction over the object vertices that indicates the predicted n-th contact point.

This is again optimized against the ground truth binary contact map label of the n-th gripper keypoint, contributing to a second binary cross-entropy loss term

$$\mathcal{L}_{P_{M_{0,\dots,n}}} = \Sigma_{i=1}^{n} \mathrm{BCE}_{\lambda_b}(P_{M_i}(v_o, k_i | k_0, ..., k_{i-1}), C_O(v_o, k_i)), \tag{6}$$

where, similarly, λ_b is the positive weight hyperparameter used to address the class imbalance. A visual representation of the autoregressive layers can be seen in 3b. Note that for $i = 0$, $P_{I_0}(v_o, k_0)$ constitutes the first marginal for k_0 and thus: $P_{I_0}(v_o, k_0) = P_{M_0}(v_o, k_0)$.

3.3 Likelihood Maps

In order to learn the above, we assumed access to grouth truth likelihood maps used for supervised learning which we obtain as follows. For each grasp in our dataset, instead of an object contact map, we generate a $(2048, 6)$ per-gripper-keypoint proximity map where the nearest areas are calculated as a fixed number of M closest points in Euclidean distance, to each of the canonical keypoints:

$$P_o(v_o, k_i) = \begin{cases} 1, & v_o \in \arg\min_M ||\mathcal{V}_O - \mathcal{V}_G(k_i)||^2 \text{ for each } k_i, \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

We also generate a gripper contact map for the selected keypoints where the contacts are defined as the keypoints closer than a given threshold, to the object point cloud:

$$C_g(k_i) = \begin{cases} 1, & \exists v_o,\ ||\mathcal{V}_O - \mathcal{V}_G(k_i)||^2 < \text{threshold}, \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

where $P_o(v_o, k_i)$ is the object proximity map, $C_g(k_i)$ is the gripper contact map, $O(v_o)$ is the object point on index v_o and $G(k_i)$ is the gripper point on the canonical keypoint index k_i. For this work, we empirically assumed $M = 20$ and a threshold of 0.04. Finally, the hand-specific object contact map can be obtained as $C_O(v_o, k_i) = P_o(v_o, k_i) \cdot C_g(k_i)$.

For training speed considerations, we preprocess the dataset prior to training, and save each grasp in this new form.

We finally represent our full training objective with the total loss being:

$$\mathcal{L}_{\text{total}} = \alpha \cdot \mathcal{L}_{P_{I_{0,\dots,n}}} + \beta \cdot \mathcal{L}_{P_{M_{0,\dots,n}}} \tag{9}$$

For our experiments, we used the Graph Convolutional Networks (GCN) implementation by Kipf et al. [28] with 3 hidden layers of size 256, $\lambda_a = 500$, $\lambda_b = 200$, 512 output embedding dimension, one for objects and one for end-effectors. The linear projection for each encoder was of size 64 without bias. We also used $\alpha = \beta = 0.5$.

3.4 Grasp Prediction at Inference

At test time, the independent unnormalized distribution for $k = 0$ is leveraged to sample keypoint 0 which will commence the autoregressive inference. More specifically, we use the 0-th dimension as a scoring mechanism for sampling high likelihood points where keypoint 0 makes contact. This is then passed into the model as the previous contact of keypoint 1. At inference time, at the n-th step, teacher forcing is substituted with passing in the $(n-1)$ predicted contact vertices. Finally, the end result is a tensor of 6 coordinates of the object graph. As previously mentioned, grasping is a multi-modal distribution and our model should be able to sample from the various modes. In our method, this can be achieved straightforwardly by sampling a variety of starting top-K points for keypoint 0. The intuition behind this is that diverse, yet likely starting points for keypoint 0 will condition subsequent predicted points differently, and ultimately yield different grasp modes. For our experiments, we sampled 4 such top-K points, namely top-0, 20, 50, 100, in order to explore the capacity of our method to generate diverse grasps. A more sophisticated sampling algorithm such as Beam search, could be applied here, however we empirically achieved sufficient diversity through multimodal sampling of keypoint 0. Here it should be noted that this autoregressive representation does present some limitations. More specifically, the ordering with which the keypoint contacts are being learning and ultimately selected could vastly change the result. However, we refrain from experimenting with all possible combinations of keypoint ordering in the scope of this work.

The end-effector joint angles are then inferred by feeding the predicted contact points into an Inverse Kinematics (IK) solver. For our purposes, we used SciPy's Trust Region Reflective algorithm (TRF) [35]. The initial pose given to IK is a heuristic pose calculated by applying a rotation/translation that aligns the palm with the closest object vertex while keeping all non-root joints at their rest pose configuration. It should be noted that any other IK solution and initial pose guess strategy could be leveraged instead. Further implementation details can be found in Appendix B.

4 Experiments

We evaluate our method through the lens of a number of research questions.

Q1: How successful is the model at producing stable and diverse grasps for various embodiments? We train our method with a training set containing samples of 5 end-effectors and 38 objects.

Figure 4: **Qualitative results.** Generated grasps using GeoMatch on **unseen** objects with ezgripper, barrett, robotiq-3finger, allegro and shadowhand. For each grasp, another perspective is included where the GeoMatch predicted keypoints on each object are marked with purple and the gripper user selected keypoints matching these, are marked with yellow.

We then generate grasps on each of the 5 end-effectors but 10 new unseen objects, and evaluate them in Isaac Gym [36], specifically the Isaac Gym based environment proposed by [12]. Similarly, we apply a consistent $0.5ms^{-2}$ acceleration on the object from all xyz directions sequentially, for 1 second each. If the object moves more than $2cm$ after every such application, the grasp is declared as a failure. We also follow the same contact-aware refinement paradigm, which applies force closure via a single step of Adam with step size 0.05. In addition, we provide calculated diversity as the standard deviation of the joint angles of all successful grasps, comparably to [12]. We compare our method to GenDexGrasp [12], AdaGrasp (initOnly as it is the closest setup to our task) [14], and DFC [17]. Results can be found in Tab. 1. In addition, we provide a number of qualitative results in Fig. 4.

Method	**Success (%) ↑**				**Diversity (rad) ↑**		
	ezgripper	barrett	shadowhand	**Mean**	ezgripper	barrett	shadowhand
DFC [17]	58.81	85.48	72.86	72.38	0.3095	0.3770	0.3472
AdaGrasp [14]	60.0	80.0	-	70.0	0.0003	0.0002	-
GenDexGrasp [12]	43.44	71.72	**77.03**	64.01	0.238	0.248	0.211
GeoMatch (Ours)	**75.0**	**90.0**	72.5	**79.17**	0.188	0.249	0.205

Table 1: **Success and diversity comparisons.** GeoMatch performs more evenly well across end-effectors with a varied DoF number while maintaining diversity of grasp configurations.

In our experiments, we observed that GeoMatch is performing slightly worse (-2%) on the 5-finger gripper Shadowhand than the best performing baseline, however performance for the 2-finger and 3-finger grippers **increases by 5-30%** compared to other methods. Diversity remains competitive to other methods. Overall, the minimum performance observed for GeoMatch is significantly higher than baselines and the average performance multi-embodiments beats all baselines we compared against.

Q2: Is the multi-embodiment model performing better than a model trained on individual embodiments? We hypothesize that training our method on data containing a variety of end-effectors will result in learning better geometry representations. To investigate this, we train our method on each single embodiment separately by filtering our dataset for each given end-effector. We then compare against the multi-embodiment model. Each of the single end-effector models is trained only on grasp instances of that gripper while the multi-embodiment model is trained on all 5 end-effectors and objects in the training set. The validation set in all cases contains 10 unseen objects. We provide results in Tab. 2. The model trained on multi-embodiment data is indeed performing **20%-35%** better than single end-effector models which advocates for the value of multi-embodiment grasping policies as opposed to single model policies trained on more data.

Q3: How robust is the learned model under relaxed assumptions? While our method demonstrates compelling results, it has been trained on full point clouds. Acknowledging that this is often a strict assumption, especially when considering real-world environments, we evaluate robustness of the approach under conditions more similar to real-world robotic data. We experiment with grasp generation using: a) noisy point clouds, b) partial point clouds, and c) partial point clouds including

Method	Success (%) ↑			Diversity (rad) ↑		
	ezgripper	barrett	shadowhand	ezgripper	barrett	shadowhand
Single embodiment	40.0	70.0	40.0	0.157	0.175	0.154
Multi embodiment	**75.0**	**90.0**	**72.5**	0.188	0.249	0.205

Table 2: Comparisons between the Multi-embodiment model and models trained on individual grippers.

noise. For each of these, we perturbed the object point clouds accordingly, and collected grasps using our method **zero-shot**. Success rate was **77.5%**, **66.7%**, and **67.5%** across end-effectors for each type of augmentation respectively. As demonstrated, our method shows reasonable robustness. Experiment details and a breakdown of numbers can be found at Appendix A.

Q4: How important are various components of the design? Finally, we investigate the design decisions of our approach and how they affect performance. More specifically, we perform two ablations:

<u>PointNet++ as the encoder of choice instead of GNN.</u> We evaluate our choice towards GNN by swapping out the two GNN encoders with PointNet++[20], a popular encoder architecture for point clouds. Our results show that GNN was indeed a good choice as it performs better than the PointNet++ ablation, by **10%** averaging across end-effectors. In addition, we empirically observed a 12x slow down when using PointNet++ due to the difference in model parameters number, which also makes GNN more light weight and fast. A breakdown per end-effector can be found in Appendix A.

<u>Non-shared weights between keypoint encoders.</u> We hypothesize that a shared encoder among all end-effectors is beneficial for learning features that represent local geometry and this subsequently, informs autoregressive prediction of keypoints. To validate this hypothesis, we conducted an ablation where we separated the end-effector encoder to 6 separate identical encoders, one per keypoint. Our main model with shared weights across all end-effectors and keypoints outperforms the split encoders by **9%**. Further analysis per end-effector can be found in Appendix A.

5 Limitations

While this method showcased that grasp learning can benefit from multi-embodiment data in terms of generalization to new objects as well as robustness, obtaining large amounts of such multi-embodiment grasping data, especially in real world setups can be challenging, time consuming and expensive. However, given that a single embodiment grasping policy was shown to require more data to perform comparably, we argue that spending resources on a multi-embodiment dataset to yield a policy that performs well across a variety of grippers is a better choice. Lastly, our method relies on the robustness of the IK solution. We empirically observed cases where there was a reasonable grasp solution for a set of predicted keypoints, however the chosen IK solution terminated in some suboptimal configuration.

6 Conclusion

This work presented a novel multi-embodiment grasping method that leverages GNN to learn powerful geometry features for object and embodiment representation. Our approach demonstrates that a joint encoder trained on multiple embodiments can better embed geometry in a generalizable fashion and ultimately result in higher grasping success rate on unseen objects. The proposed framework also showcased robustness to more realistic point cloud inputs. Diversity of generated grasps remains competitive while producing such diverse grasps is as simple as conditioning with a different high likelihood starting contact point for the first keypoint. Code and models will be released on acceptance.

References

[1] A. Nguyen, D. Kanoulas, D. G. Caldwell, and N. G. Tsagarakis. Detecting Object Affordances with Convolutional Neural Networks. In *International Conference on Intelligent Robots and Systems (IROS)*, 2016.

[2] H.-S. Fang, C. Wang, M. Gou, and C. Lu. GraspNet-1Billion: A Large-Scale Benchmark for General Object Grasping. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[3] A. Mousavian, C. Eppner, and D. Fox. 6-dof graspnet: Variational grasp generation for object manipulation. In *International Conference on Computer Vision (ICCV)*, 2019.

[4] A. ten Pas, M. Gualtieri, K. Saenko, and R. Platt. Grasp pose detection in point clouds. *The International Journal of Robotics Research*, 36(13-14), 2017.

[5] M. Gualtieri, A. ten Pas, K. Saenko, and R. Platt. High precision grasp pose detection in dense clutter. In *International Conference on Intelligent Robots and Systems (IROS)*, 2016.

[6] Z. Jiang, Y. Zhu, M. Svetlik, K. Fang, and Y. Zhu. Synergies between affordance and geometry: 6-dof grasp detection via implicit representations. *arXiv preprint arXiv:2104.01542*, 2021.

[7] S. Wang, Z. Zhou, and Z. Kan. When transformer meets robotic grasping: Exploits context for efficient grasp detection. *Robotics and Automation Letters*, 7(3), 2022.

[8] C. Wu, J. Chen, Q. Cao, J. Zhang, Y. Tai, L. Sun, and K. Jia. Grasp Proposal Networks: An End-to-End Solution for Visual Learning of Robotic Grasps. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[9] Y. Xu, W. Wan, J. Zhang, H. Liu, Z. Shan, H. Shen, R. Wang, H. Geng, Y. Weng, J. Chen, et al. Unidexgrasp: Universal robotic dexterous grasping via learning diverse proposal generation and goal-conditioned policy. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2023.

[10] W. Wan, H. Geng, Y. Liu, Z. Shan, Y. Yang, L. Yi, and H. Wang. UniDexGrasp++: Improving Dexterous Grasping Policy Learning via Geometry-aware Curriculum and Iterative Generalist-Specialist Learning. *arXiv preprint arXiv:2304.00464*, 2023.

[11] D. Turpin, L. Wang, E. Heiden, Y.-C. Chen, M. Macklin, S. Tsogkas, S. Dickinson, and A. Garg. Grasp'd: Differentiable contact-rich grasp synthesis for multi-fingered hands. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part VI*, 2022.

[12] P. Li, T. Liu, Y. Li, Y. Geng, Y. Zhu, Y. Yang, and S. Huang. Gendexgrasp: Generalizable dexterous grasping. *arXiv preprint arXiv:2210.00722*, 2022.

[13] L. Shao, F. Ferreira, M. Jorda, V. Nambiar, J. Luo, E. Solowjow, J. A. Ojea, O. Khatib, and J. Bohg. Unigrasp: Learning a unified model to grasp with multifingered robotic hands. *Robotics and Automation Letters*, 5(2), 2020.

[14] Z. Xu, B. Qi, S. Agrawal, and S. Song. Adagrasp: Learning an adaptive gripper-aware grasping policy. In *International Conference on Robotics and Automation (ICRA)*, 2021.

[15] M. Sundermeyer, A. Mousavian, R. Triebel, and D. Fox. Contact-graspnet: Efficient 6-dof grasp generation in cluttered scenes. In *International Conference on Robotics and Automation (ICRA)*, 2021.

[16] S. Brahmbhatt, A. Handa, J. Hays, and D. Fox. ContactGrasp: Functional Multi-finger Grasp Synthesis from Contact. In *International Conference on Intelligent Robots and Systems (IROS)*, 2019.

[17] T. Liu, Z. Liu, Z. Jiao, Y. Zhu, and S.-C. Zhu. Synthesizing diverse and physically stable grasps with arbitrary hand structures using differentiable force closure estimator. *Robotics and Automation Letters*, 7(1), 2021.

[18] D. Turpin, T. Zhong, S. Zhang, G. Zhu, E. Heiden, M. Macklin, S. Tsogkas, S. Dickinson, and A. Garg. Fast-Grasp'D: Dexterous Multi-finger Grasp Generation Through Differentiable Simulation. In *ICRA*, 2023.

[19] K. Sohn, H. Lee, and X. Yan. Learning Structured Output Representation using Deep Conditional Generative Models. In *Advances in Neural Information Processing Systems*, 2015.

[20] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 2017.

[21] K. Li, N. Baron, X. Zhang, and N. Rojas. Efficientgrasp: A unified data-efficient learning to grasp method for multi-fingered robot hands. *Robotics and Automation Letters*, 7(4), 2022.

[22] C. Pan, B. Okorn, H. Zhang, B. Eisner, and D. Held. TAX-Pose: Task-Specific Cross-Pose Estimation for Robot Manipulation. In *Conference on Robot Learning (CoRL)*, 2022.

[23] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini. The Graph Neural Network Model. *Transactions on Neural Networks*, 2009.

[24] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun. Graph neural networks: A review of methods and applications. *AI open*, 1, 2020.

[25] I. Huang, Y. Narang, R. Bajcsy, F. Ramos, T. Hermans, and D. Fox. DefGraspNets: Grasp Planning on 3D Fields with Graph Neural Nets. *arXiv preprint arXiv:2303.16138*, 2023.

[26] A. Garcia-Garcia, B. S. Zapata-Impata, S. Orts-Escolano, P. Gil, and J. Garcia-Rodriguez. Tactilegcn: A graph convolutional network for predicting grasp stability with tactile sensors. In *International Joint Conference on Neural Networks (IJCNN)*, 2019.

[27] X. Lou, Y. Yang, and C. Choi. Learning object relations with graph neural networks for target-driven grasping in dense clutter. In *International Conference on Robotics and Automation (ICRA)*, 2022.

[28] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[29] X. Yan, J. Hsu, M. Khansari, Y. Bai, A. Pathak, A. Gupta, J. Davidson, and H. Lee. Learning 6-DOF Grasping Interaction via Deep Geometry-Aware 3D Representations. In *International Conference on Robotics and Automation (ICRA)*, 2018.

[30] M. Van der Merwe, Q. Lu, B. Sundaralingam, M. Matak, and T. Hermans. Learning continuous 3d reconstructions for geometrically aware grasping. In *International Conference on Robotics and Automation (ICRA)*, 2020.

[31] J. Bohg and D. Kragic. Learning grasping points with shape context. *Robotics and Autonomous Systems*, 58(4), 2010.

[32] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li. On the continuity of rotation representations in neural networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[33] B. Calli, A. Singh, J. Bruce, A. Walsman, K. Konolige, S. Srinivasa, P. Abbeel, and A. M. Dollar. Yale-CMU-Berkeley dataset for robotic manipulation research. *The International Journal of Robotics Research*, 36(3), 2017.

[34] S. Brahmbhatt, C. Ham, C. C. Kemp, and J. Hays. ContactDB: Analyzing and Predicting Grasp Contact via Thermal Imaging. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[35] J. J. Moré and D. C. Sorensen. Computing a Trust Region Step. *SIAM Journal on Scientific and Statistical Computing*, 4(3), 1983.

[36] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State. Isaac Gym: High Performance GPU-Based Physics Simulation For Robot Learning, 2021.