Leveraging CVAE for Joint Configuration Estimation of Multifingered Grippers from Point Cloud Data

Julien Mérand[1], Boris Meden[1] and Mathieu Grossard[1]
[1]Université Paris-Saclay, CEA, List

***Abstract*— This paper presents an efficient approach for determining the joint configuration of a multifingered gripper solely from the point cloud data of its poly-articulated chain, as generated by visual sensors, simulations or even generative neural networks. Well-known inverse kinematics (IK) techniques can provide mathematically exact solutions (when they exist) for joint configuration determination based solely on the fingertip pose, but often require post-hoc decision-making by considering the positions of all intermediate phalanges in the gripper's fingers, or rely on algorithms to numerically approximate solutions for more complex kinematics. In contrast, our method leverages machine learning to implicitly overcome these challenges. This is achieved through a Conditional Variational Auto-Encoder (CVAE), which takes point cloud data of key structural elements as input and reconstructs the corresponding joint configurations. We validate our approach on the MultiDex grasping dataset using the Allegro Hand, operating within 0.05 milliseconds and achieving accuracy comparable to state-of-the-art methods. This highlights the effectiveness of our pipeline for joint configuration estimation within the broader context of AI-driven techniques for grasp planning.**

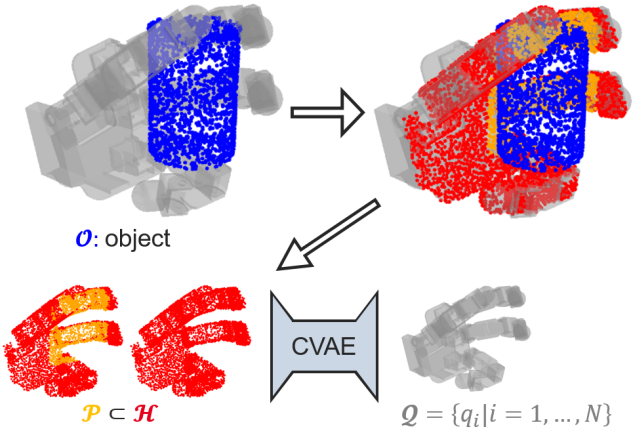

Fig. 1: Given a gripper point cloud $\mathcal{H}$ or a set of contact points $\mathcal{P} \subset \mathcal{H}$, our method reconstructs the joint configuration $\mathcal{Q}$. We evaluate this approach in a grasping context using the MultiDex dataset [13].

I. Introduction

Determining joint configurations for multi-fingered robotic grippers is a critical challenge from a control perspective, as precise joint control is essential for accurately positioning fingertips or phalanges at the desired contact points on the object. Indeed, several well-known approaches for generating valid grasps rely on analytical metrics, such as force- or form-closure criteria [1]–[3]. These methods heavily depend on the knowledge of the reachable contact points to analyze grasp properties [4]. From a geometric perspective, the selected grasp configuration must be kinematically reachable by the fingers and collision-free with respect to the environment. This often requires extensive simulation trials to test the accessibility of contact point, implicitly computing the Inverse Kinematics (IK) of each robotic finger.

Although mathematically solvable with an analytical foundation in some cases, the IK problem is not always straightforward, complicating reliability and efficiency in dynamic environments. IK problems can yield multiple solutions — or even an infinite number of solutions in the case of redundant systems — where the poses of intermediate phalanges significantly influence the determination of the appropriate solution. This complexity can also hinder the numerical approximation of solutions. For example, [5] tackles this challenge by iteratively solving a system of algebraic equations to address the IK problem of an underactuated finger mechanism with coupling.

To mitigate this issues, early works explored the use of feed-forward neural networks with a limited number of joints [6], [7], genetic algorithms [8], and Neuro-Fuzzy Inference Systems [9] to solve the IK problem. However, these methods often resulted in large errors and typically returned only a single solution for a given input, which limited their applicability in more complex scenarios. More recent approaches have leveraged generative networks [10]–[12] to overcome these limitations, providing the complete solution space and enabling the selection of the most appropriate solution afterwards.

Our method builds on this progress by leveraging deep learning techniques to directly infer joint configurations and implicitly select the correct configuration from the set of admissible solutions. This approach aims to streamline the grasping process, enhancing efficiency and adaptability in line with recent state-of-the-art methods that focus on determining the multifingered gripper pose from a 3D Point Cloud (PC) representation of the scene using AI-based paradigms [14], [15]. The prevalent approach in the field is to employ numerical optimization processes to derive the optimal joint configuration from this resulting Cartesian fingertip pose [16]–[18]. While an abundance of models address the initial part of the problem [13], [19]–[24] where data is collected in the form of PC (obtained either through visual sensors or simulations), we specifically focus on an AI-driven approach for determining the joint configuration (Figure 1). Our aim is to replace the optimization step with an AI-driven approach, aligning with current trends and propos-

ing a method compatible with state-of-the-art practices. By considering intermediate phalanges within the point cloud data, our model recognizes patterns corresponding to specific joint configurations. This eliminates the need for additional decision-making criteria, simplifying the process and enhancing reliability and efficiency in complex manipulation tasks.

The paper is organized as follows. In Section II, we introduce the principle of our method, which leverages a Conditional Variational Auto-Encoder (CVAE) [25] to reconstruct the gripper configuration from its PC representation. We also discuss the relevance of several gripper-related datasets (to account for several practical application cases) and hyperparameters for training, as well as details about the model architecture and its implementation. Section III provides qualitative and quantitative evaluations of our model's performance, using the recent MultiDex grasp dataset [13] as the core material for evaluation. The influence of the dataset on accurately predicting and reconstructing gripper configurations from new and unseen PCs is also examined. Finally, Section IV concludes the paper by summarizing the contributions of our method and offering perspectives for future work.

II. Method

We propose to train a CVAE [25] to reconstruct a gripper configuration from a point cloud. By training the CVAE on a dataset of PCs paired with corresponding gripper configurations, we aim to develop a model that can accurately predict and reconstruct gripper configurations from new, unseen PCs.

A. Dataset Generation

Our data generation process is robot-centric and requires only the gripper's URDF to access both its kinematic model and CAD files. The procedure involves the following sequential steps:

1) Generating joint configurations: A total of M joint configurations are sampled from the range of motion for each joint, based on a uniform distribution on $[0, 1]$ to guarantee training stability.
2) Filtering collision-free samples: Given the complex workspace of a multifingered gripper, certain configurations can result in self-collisions, where phalanges overlap or intersect. These configurations are not viable for practical applications. To address this, each generated joint configuration is used as input to the analytical Forward Kinematics model to determine the Cartesian pose of each link. We then implement a collision avoidance function, $f_{\text{self\_collision}}$, inspired by [26], to detect and filter out configurations where any two links are in collision:

$$f_{\text{self\_collision}} = \sum_{i=1}^{L} \sum_{j=1}^{L} \mu_{i,j} \cdot \max\Big(\delta_{i,j} - \|\mathbf{k}_i - \mathbf{k}_j\|_2, 0\Big), \tag{1}$$

where:

- $\mathbf{k}_i$ and $\mathbf{k}_j$ are predefined keypoints representing the center of each link's bounding box, given a gripper configuration.
- L is the total number of links.
- $\delta_{i,j}$ is a distance threshold that defines the minimum allowable distance between links.
- $\mu_{i,j}$ is a weighting factor, set to 1 here.

If $f_{\text{self\_collision}}$ is non-zero, it indicates that at least two links are in collision, and the configuration is deemed invalid.

3) Point Cloud creation: Finally, each link's mesh surface is evenly sampled at varying densities to create a gripper PC.

To demonstrate the model's capability to address a wide range of applications, three distinct datasets are covered following the previous procedure.

- **Fully Dense PC:** This dataset consists of PCs with 512 points equally spaced around the entire gripper. It is designed to benefit methods that map a gripper point cloud to an object, such as in [23].
- **Cluster PC:** This dataset uses a small yet representative set of points from each link, aligning with the definition of IK. It ensures that each link is represented by at least one cluster of points, capturing essential spatial information for an accurate determination of the joint configuration. For a given link, a cluster is defined as the set of all points that lie within a sphere of radius R. The sphere is centered on the inner surface of the link. This dataset is designed for object-oriented methods, focusing on the regions of the gripper that make contact with the held object through a set of contact points (denoted as $\mathcal{P}$ in Figure 1).
- **Handprint PC:** This dataset focuses on relevant parts of the gripper by filtering surface points using a dot product criterion. It retains only the points that belong to the inner side of the fingers and the palm. Specifically, the dot product is computed between the outward normal of the palm and each point's normal when the hand is in the nominal configuration, with all joints set to 0 radians. This dataset offers a balanced representation, explicitly capturing the gripper's features, especially around the joints, and is closer to real-world measurements, making it a practical compromise between the other two datasets.

All configurations within these datasets were considered within the same reference frame.

To ensure that the entire workspace is covered to achieve maximum generality, we assessed the diversity of all generated joint configurations by calculating the standard deviation of the joint angles' distribution (refer to Section III-A). This measurement helps verify that the configurations are varied and comprehensive, effectively covering the full range of possible finger motions.

B. Model Architecture

Given a complete multifingered robotic gripper point cloud, $\mathcal{H} \in \mathbb{R}^{n\times 3}$, we define a subset of points $\mathcal{S} = \{p \in \mathcal{H}\}$. The overall objective of the model is to retrieve the N joints parameters of the gripper $\mathcal{Q} = \{q_i, i \in [1, N]\}$.

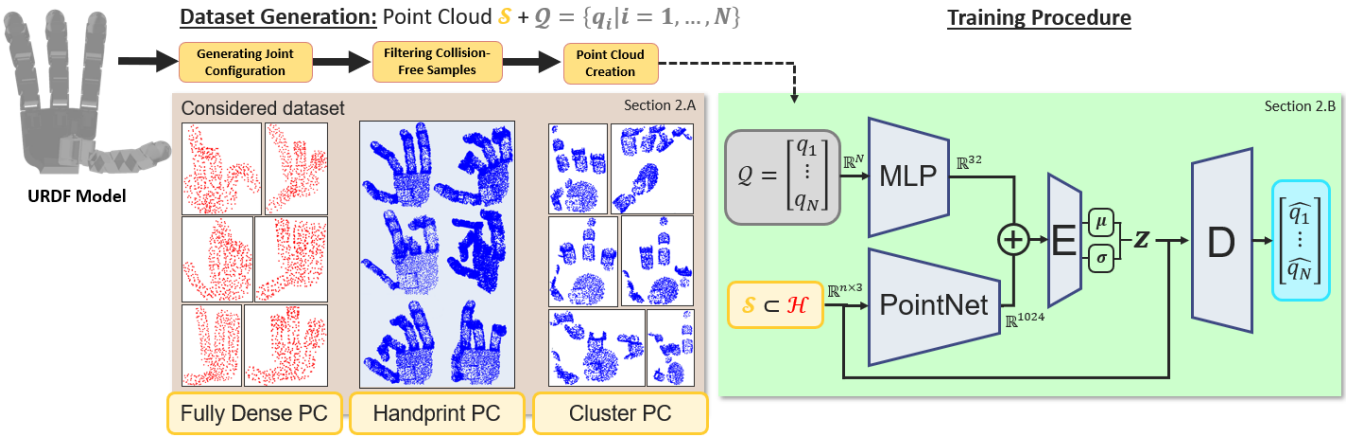

Fig. 2: Overview of our method: The left panel illustrates the dataset generation process. Given a gripper URDF model, three datasets of PCs (Fully Dense PC, Handprint PC, Cluster PC) are created as described in Section II-A, each associated with its joint configuration ($\mathcal{Q}$). The right panel illustrates the model architecture, where a CVAE generates a joint configuration ($\mathcal{Q}$) from a subset ($\mathcal{S}$) of the gripper PC ($\mathcal{H}$). The architecture includes an encoding step that processes the PC using a PointNet, and the joint configuration with a fully connected MLP. These encoded representations are then concatenated and further processed by a latent encoder (E). The decoding step (D) involves processing the encoded PC with a sample from the latent space z (characterized by mean μ and variance σ) to reconstruct the joint configuration $\hat{\mathcal{Q}}$ using a fully connected MLP.

The CVAE [25] is trained to reconstruct $\mathcal{Q}$ knowing $\mathcal{S}$. It consists of three main components: an encoder, a decoder and a prior distribution. The encoder (recognition network) aims to map the input $\mathcal{S}$, conditioned on $\mathcal{Q}$, to a latent distribution $q_\phi(z|\mathcal{S}, \mathcal{Q})$. We employ a PointNet encoder [27] to extract features from the points, which are then concatenated with joint value features obtained using a fully connected Multi-Layer Perceptron (MLP). The decoder (generation network), denoted as $p_\theta(\mathcal{Q}|\mathcal{S}, z)$, focuses on reconstructing the output $\mathcal{Q}$ based on the input $\mathcal{S}$ and the latent variable z. Concretely, the process is reversed compared to the encoder. We first extract the gripper point features using another PointNet, then concatenate the latent variable z with these per-point features, to generate the joint angles using a fully connected MLP. Finally, the prior distribution $p_\theta(z|x)$ serves as a regularizer, ensuring that the latent space follows this prior distribution. We chose to make the prior distribution independent of the input variable $p_\theta(z|x) = p_\theta(z) \sim \mathcal{N}(0, 1)$, to enhance flexibility in the learning process.

We train the generative model by maximizing the log-likelihood of $p_{\theta,\phi}(y|x)$, where θ and ϕ are the learnable parameters of the encoder and decoder, respectively. To achieve this, we maximize the evidence lower bound (ELBO) between the estimated posterior distribution and the true posterior distribution [28]. This is equivalent to solving a minimization problem with the following loss function:

$$\mathcal{L}_{\text{CVAE}} = \mathcal{L}_{\text{recon}} + \beta D_{\text{KL}}(q_\phi(z|\mathcal{S}, \mathcal{Q}) \| p_\theta(z)). \tag{2}$$

The first component $\mathcal{L}_{\text{recon}}$ is the reconstruction loss, representing the log-likelihood of $p_\theta(\mathcal{Q}|\mathcal{S}, z)$:

$$\mathcal{L}_{\text{recon}} = -\mathbb{E}_{q_\phi(z|\mathcal{S},\mathcal{Q})}[\log p_\theta(\mathcal{Q}|\mathcal{S}, z)]. \tag{3}$$

We approximate this term by a standard RMSE between the ground truth $\mathcal{Q}$ and the prediction $\hat{\mathcal{Q}}$. In other words, it computes the pairwise angular distance:

$$\mathcal{L}_{\text{recon}} = \mathbb{E}\left[\sqrt{\frac{\sum_{i=1}^{N}(q_i - \hat{q}_i)^2}{N}}\right]. \tag{4}$$

The second component, the Kullback-Leibler Divergence (D_{KL}), is applied to measure how much the learned distribution $q_\phi(z|\mathcal{S}, \mathcal{Q})$ diverges from the prior distribution $p_\theta(z)$.

These two terms are weighted by the hyperparameter β that can be either set to a fixed value or take the form of a monotonic or cyclical schedule in order to mitigate the known KL vanishing problem [29].

C. Implementation details

The parameter β (see Equation 2) was initially set to 0.0001 for the first 50 epochs, then followed a Sigmoid curve ranging from 0.0001 to 1.0 between epochs 50 and 100. After epoch 100, β remained constant at 1.0 until the end of training, as we saw no improvements using a cyclical schedule. The evolution of β over 250 epochs is illustrated Figure 3.

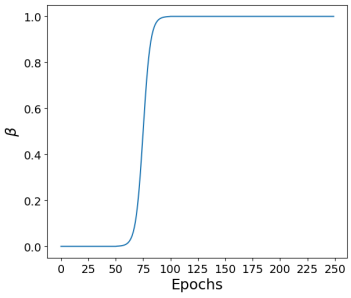

Fig. 3: Evolution of β over 250 epochs.

We defined the dataset size as $M = 30,000$ joint configurations, as it ensures comprehensive coverage of the Allegro's workspace and to facilitate comparison with our evaluation dataset. R is set to be 50% of each link's radius. The training process involved 4000 epochs with a batch size of 500, utilizing a single Nvidia GeForce RTX 4090 GPU. Training on the Fully Dense PC dataset required a total of 2 GPU hours. This dataset was split into 80% for training and 20% for testing. We used the Adam optimizer with a learning rate of 0.0001 for optimization. We conducted our experiment on the Allegro Hand V4 [30], which features $N = 16$ independent joints distributed across four fingers—three generic fingers and one thumb—in an anthropomorphic design.

III. EVALUATION

Our evaluation of the model's performance leverages the extensive MultiDex dataset [13]. We start with a qualitative analysis comparing this dataset to our training data. Following this, we conduct a quantitative evaluation to assess the model's accuracy and efficiency using various robotics-based metrics.

A. Dataset: Qualitative analysis

We evaluated the accuracy of our method using the extensive grasp dataset MultiDex [13], created by Li et al. This dataset includes 5 multifingered grippers (EZGripper, Barrett, Robotiq-3F, Allegro, and ShadowHand), 58 household objects from YCB [31] and ContactDB [32], and 436,000 diverse grasping poses. We considered the grasping poses of the Allegro Hand, resulting in 37,774 hand PC, compared to the 30,000 PC in our training dataset. The PC are sampled following the same procedure as for our three datasets (Fully Dense PC, Handprint PC, Cluster PC).

Upon examining these configurations, we noticed frequent instances of finger collisions or overlaps. Applying our collision criteria (Equation 1), only 64% (24,077 out of 37,774) of the configurations would have been retained. However, to demonstrate our model's ability to generalize, we included these configurations in the validation set, meaning that at least 36% of the MultiDex dataset [13] is not included in our training dataset.

Our proposed dataset is uniformly distributed, reflecting a comprehensive coverage of the gripper's workspace (Figure 5), while the distribution in the MultiDex dataset [13] is not uniform, likely due to its focus on grasp-related poses. Our dataset exhibited a standard deviation of 0.287, compared to 0.250 for the MultiDex dataset [13] (Figure 4). This aligns closely with the expected standard deviation of a uniform distribution on $[0, 1]$, which is $\frac{1}{\sqrt{12}} \approx 0.2887$.

Figure 5 illustrates the workspace of the generic finger and the thumb from a distal perspective, comparing our dataset Fully Dense PC (blue dots) with the MultiDex dataset [13] (red dots). The workspaces of the generic fingers of the Allegro Hand, along with the thumb, which exhibits a distinct kinematic diagram, are analyzed. This visualization highlights that the MultiDex dataset [13] does not adequately represent certain regions of the entire workspace. It is worth noting that areas close to the root axis of the thumb are not as well covered in our dataset. This discrepancy is due to our collision avoidance criteria, which do not consider unrealistic configurations from a practical point of view.

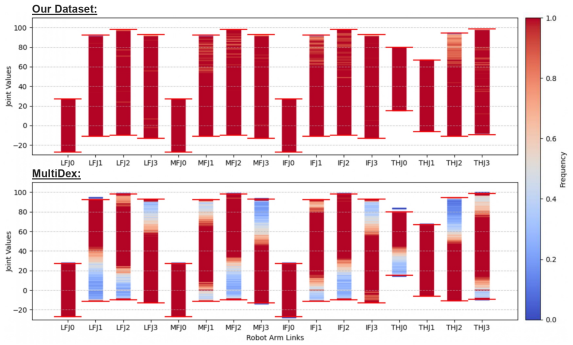

Fig. 4: Joint Distribution of the Allegro Hand for both datasets. Top: Fully Dense PC, Bottom: MultiDex dataset [13].

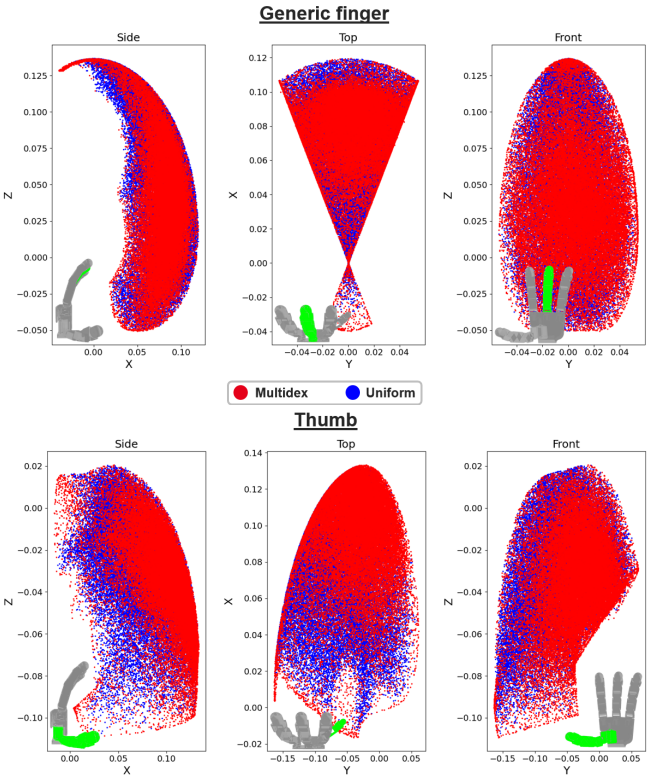

Fig. 5: Workspace analysis of the generic and thumb fingers.

B. Quantitative Evaluation

To evaluate the performance of our model, we considered the following quantitative metrics:

- **Mean Joint Error (rad)**: This metric represents the mean error across all joints. An error of 0.01 radians indicates that, on average, each joint in a predicted configuration deviates by 0.01 radians from the ground truth. This metric provides a direct measure of the precision in joint angle predictions.

Metrics		Fully Dense PC	Cluster PC	Handprint PC	Generic finger	Thumb
Mean Joint error	(rad ↓)	0.075 ± 0.042 4.49%	0.064 ± 0.029 3.90%	0.063 ± 0.035 3.84%	0.017 ± 0.012 1.04%	0.018 ± 0.009 1.10%
Mean of Lowest Joint error per batch	(rad ↓)	0.022 ± 0.003 1.43%	0.029 ± 0.003 1.94%	0.022 ± 0.003 1.43%	0.002 ± 0.0005 0.13%	0.002 ± 0.0004 0.12%
Mean Cartesian error	(mm ↓)	3.889 ± 2.421 2.89%	3.967 ± 1.633 3.07%	3.821 ± 2.023 2.85%	0.219 ± 0.178 0.64%	0.363 ± 0.380 1.09%
Mean of Lowest Cartesian error per batch	(mm ↓)	1.331 ± 0.176 1.03%	1.911 ± 0.227 1.54%	1.475 ± 0.223 1.13%	0.031 ± 0.009 0.09%	0.024 ± 0.006 0.07%
Time	(ms ↓)	0.042	0.050	0.048	0.044	0.043

TABLE I: Performance evaluation on MultiDex [13] dataset. We evaluate our model's ability to estimate gripper configurations using PCs from the MultiDex dataset [13]. The evaluation considers both Joint error and Cartesian error metrics.

- **Mean Cartesian error (mm)** : This metric is the mean of the L2 norm between ground truth and predicted keypoints, using the same keypoints as in Equation 1. It focuses on positional accuracy, as rotational errors are implicitly considered in the finger kinematic diagram. This metric complements joint error by capturing the cumulative effect of errors along the poly-articulated chain. For example, a small error at the finger's base can lead to a larger displacement error at the fingertip.
- **Inference time (ms)**: We measured the inference time on a single Nvidia GeForce RTX 4090 GPU, using a batch size of 100, and computed the mean inference time across all batches. This metric is crucial for evaluating the perspective of embedding our model in a robotic time-constrained controller.

Table I presents the results achieved on the MultiDex dataset [13]. The error rate is relative to the maximum achievable error. For joint error, this corresponds to the full range of each joint. For Cartesian error, it corresponds to the maximal distance between keypoints, determined by evaluating the Euclidean distance between the extreme positions of each keypoint, considering the full range of motion for each joint.

The resulting processing time for inferring the joint configuration is less than 0.05 ms, using the same resources as those employed for training (Section II-C). This efficiency is fully compatible with the real-time constraints of robot controllers, which typically operate within a few milliseconds. Therefore, this approach is well-suited for integration into various gripper control strategies.

Comparing our datasets —Fully Dense PC, Handprint PC, and Cluster PC— showed varying levels of accuracy and efficiency. The Handprint PC dataset, for instance, exhibited improved accuracy compared to the Cluster PC dataset, likely due to better feature representation since more points describe the gripper configuration. However, the timing differences were minimal, suggesting that our model's efficiency is robust across different data representations.

The results for the generic finger and the thumb are of comparable magnitude, despite the thumb having a more complex kinematic diagram that allows for distinct behavior, such as opposing the three generic fingers. This demonstrates that our model is independent of any specific kinematic structure, and the method can thus be easily generalized to any multifingered gripper.

The table includes the mean of the minimum error per batch, both Joint and Cartesian, to provide insight into the model's potential accuracy. Given that a CVAE inherently maps one input to multiple possible outputs, inferring with multiple latent samples and selecting the best result (i.e., the one with the lowest error) could improve performance. However, this approach would multiply the inference time, making it less suitable for real-time applications.

The achieved accuracy is comparable (all things considered) to other methods that leverage CVAE to compute the IK model of a robot, as demonstrated in [12].

It is important to note that the PointNet encoder, designed to be invariant to geometric transformations, does not fully achieve this invariance in practice. Introducing rotations and translations to the dataset's PCs would require a different encoder architecture. This new architecture must understand and distinguish both the relationships between points defining the joint configuration and the overall position of the point cloud, representing the gripper's 6D pose.

IV. Conclusion

This paper proposes a CVAE-based pipeline for joint configuration estimation of complex poly-articulated chains using only their Point Cloud representation. The architecture consists of encoding and decoding steps, incorporating PointNet and MLP networks. The results, considering the Allegro Hand and the grasp patterns included in the MultiDex dataset [13], demonstrate that the method meets the precision and inference time requirements for robotic grasping. Its performance is on par with state-of-the-art methods, making it a viable option to be included as part of existing AI-driven grasp planning and control strategies. Additionally, a key advantage of this method is its robot-centric approach. Generating training data is quite straightforward, requiring only the gripper's URDF to access both its kinematic model and CAD files. This simplicity significantly enhances the model's adaptability and efficiency.

To expand the applicability of our pipeline to hand pose estimation and improve its robustness across various grasping

scenarios, we plan to estimate the gripper's pose, as an element of $SE(3)$, relative to a desired reference frame. From the perspective of grasping tasks, this step is crucial for establishing a mapping between the object pose and the gripper's pose. Therefore, future work will focus on extending the proposed architecture to estimate the gripper's pose while simultaneously optimizing the model's hyperparameters.

ACKNOWLEDGMENTS

This project has received funding from the European Union's Horizon Europe research and innovation program under grant agreement nº 101135708 (JARVIS Project).

REFERENCES

[1] A. Rodriguez, M. T. Mason, and S. Ferry, "From caging to grasping," The International Journal of Robotics Research, vol. 31, no. 7, pp. 886–900, 2012.

[2] D. Prattichizzo, M. Malvezzi, M. Gabiccini, and A. Bicchi, "On the manipulability ellipsoids of underactuated robotic hands with compliance," Robotics and Autonomous Systems, vol. 60, no. 3, pp. 337–346, 2012.

[3] C. Rosales, R. Suárez, M. Gabiccini, and A. Bicchi, "On the synthesis of feasible and prehensile robotic grasps," in 2012 IEEE international conference on robotics and automation. IEEE, 2012, pp. 550–556.

[4] D. Prattichizzo and J. C. Trinkle, "Grasping," Springer handbook of robotics, pp. 955–988, 2016.

[5] J. M. Escorcia-Hernández, M. Grossard, F. Gosselin, and C. Dubois, "An iterative method for solving the inverse kinematic problem of three-joints robotic fingers with distal coupling." in 2023 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM). IEEE, 2023, pp. 623–628.

[6] R. Köker, C. Öz, T. Çakar, and H. Ekiz, "A study of neural network based inverse kinematics solution for a three-joint robot," Robotics and autonomous systems, vol. 49, no. 3-4, pp. 227–234, 2004.

[7] A.-V. Duka, "Neural network based inverse kinematics solution for trajectory tracking of a robotic arm," Procedia Technology, vol. 12, pp. 20–27, 2014.

[8] R. KöKer, "A genetic algorithm approach to a neural-network-based inverse kinematics solution of robotic manipulators based on error minimization," Information Sciences, vol. 222, pp. 528–543, 2013.

[9] J. Demby's, Y. Gao, and G. N. DeSouza, "A study on solving the inverse kinematics of serial robots using artificial neural network and fuzzy neural network," in 2019 IEEE international conference on fuzzy systems (FUZZ-IEEE). IEEE, 2019, pp. 1–6.

[10] H. Ren and P. Ben-Tzvi, "Learning inverse kinematics and dynamics of a robotic manipulator using generative adversarial networks," Robotics and Autonomous Systems, vol. 124, p. 103386, 2020.

[11] B. Ames, J. Morgan, and G. Konidaris, "Ikflow: Generating diverse inverse kinematics solutions," IEEE Robotics and Automation Letters, vol. 7, no. 3, pp. 7177–7184, 2022.

[12] C. Gaebert and U. Thomas, "Generating dual-arm inverse kinematics solutions using latent variable models," in 2024 IEEE-RAS 23rd International Conference on Humanoid Robots (Humanoids). IEEE, 2024, pp. 373–380.

[13] P. Li, T. Liu, Y. Li, Y. Geng, Y. Zhu, Y. Yang, and S. Huang, "Gendexgrasp: Generalizable dexterous grasping," in 2023 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2023, pp. 8068–8074.

[14] R. Newbury, M. Gu, L. Chumbley, A. Mousavian, C. Eppner, J. Leitner, J. Bohg, A. Morales, T. Asfour, D. Kragic et al., "Deep learning approaches to grasp synthesis: A review," IEEE Transactions on Robotics, vol. 39, no. 5, pp. 3994–4015, 2023.

[15] J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-driven grasp synthesis—a survey," IEEE Transactions on robotics, vol. 30, no. 2, pp. 289–309, 2013.

[16] Y. Zhou and K. Hauser, "6dof grasp planning by optimizing a deep learning scoring function," in Robotics: Science and systems (RSS) workshop on revisiting contact-turning a problem into a solution, vol. 2, 2017, p. 6.

[17] A. Wu, M. Guo, and C. K. Liu, "Learning diverse and physically feasible dexterous grasps with generative model and bilevel optimization," arXiv preprint arXiv:2207.00195, 2022.

[18] 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 European Conference on Computer Vision. Springer, 2022, pp. 201–221.

[19] 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," IEEE Robotics and Automation Letters, vol. 5, no. 2, pp. 2286–2293, 2020.

[20] Z. Xu, C. Gao, Z. Liu, G. Yang, C. Tie, H. Zheng, H. Zhou, W. Peng, D. Wang, T. Hu et al., "Manifoundation model for general-purpose robotic manipulation of contact synthesis with arbitrary objects and robots," in 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2024, pp. 10 905–10 912.

[21] J. Varley, J. Weisz, J. Weiss, and P. Allen, "Generating multi-fingered robotic grasps via deep learning," in 2015 IEEE/RSJ international conference on intelligent robots and systems (IROS). IEEE, 2015, pp. 4415–4420.

[22] 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," in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 3891–3902.

[23] Z. Wei, Z. Xu, J. Guo, Y. Hou, C. Gao, Z. Cai, J. Luo, and L. Shao, "D (r, o) grasp: A unified representation of robot and object interaction for cross-embodiment dexterous grasping," arXiv e-prints, pp. arXiv–2410, 2024.

[24] J. Lu, H. Kang, H. Li, B. Liu, Y. Yang, Q. Huang, and G. Hua, "Ugg: Unified generative grasping," in European Conference on Computer Vision. Springer, 2024, pp. 414–433.

[25] K. Sohn, H. Lee, and X. Yan, "Learning structured output representation using deep conditional generative models," Advances in neural information processing systems, vol. 28, 2015.

[26] T. Zhu, R. Wu, X. Lin, and Y. Sun, "Toward human-like grasp: Dexterous grasping via semantic representation of object-hand," in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 15 741–15 751.

[27] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 652–660.

[28] D. P. Kingma, M. Welling et al., "Auto-encoding variational bayes," 2013.

[29] H. Fu, C. Li, X. Liu, J. Gao, A. Celikyilmaz, and L. Carin, "Cyclical annealing schedule: A simple approach to mitigating kl vanishing," arXiv preprint arXiv:1903.10145, 2019.

[30] "Allegro Hand V4." [Online]. Available: https://www.allegrohand.com/v4

[31] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar, "The ycb object and model set: Towards common benchmarks for manipulation research," in 2015 international conference on advanced robotics (ICAR). IEEE, 2015, pp. 510–517.

[32] S. Brahmbhatt, C. Ham, C. C. Kemp, and J. Hays, "Contactdb: Analyzing and predicting grasp contact via thermal imaging," in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 8709–8719.