Generative Graphical Inverse Kinematics Plus (GGIK+)

Calvin Stahoviak, Gabriel Urbaitis

***Abstract*—Inverse kinematics is critical in almost all robotics tasks. Although it has been extensively researched, the increasing complexity of robot kinematic structures requires equally complex solutions. Numerical methods currently dominate this space for most manipulators, however, learned methods have slowly become comparable in speed and accuracy. This paper presents a modified version of the Generative Graphical Inverse Kinematics (GGIK) framework, called GGIK+[1], which addresses its limitations in handling joint structures with non-coplanar rotation axes. GGIK+ introduces attention-based message passing, directional edge vectors, and expanded node feature representations. These additions are intended to help the model better prioritize important joints and capture spatial layout more effectively. However, even with the added features, GGIK+ did not outperform the base GGIK model. Notably, improvements were found in the base GGIK architecture when trained on non-coplanar data.**

***Index Terms*—Inverse kinematics, robotics, graph neural networks, self-attention, distance geometry**

I. Introduction

Control over the position and orientation of an end-effector is one of, if not the most critical capability for a robot. In addition, knowledge of the position and orientation of the end-effector can be necessary as well. A multitude of tasks, including manipulation [1], [2], [3], grasping [4], transport [5], human-robot interaction [6], surgical robotics [7], [8], etc. would be impossible to perform without this capability. The latter, knowledge of the end-effector pose, can be readily solved for using *forward kinematics*. Forward kinematics is the process of calculating the end-effectors pose given the joint angles and a description of the robot's kinematic structure. This can be framed as a problem of translating the robot from joint angle space to Cartesian space (or work space) [9]. Less trivial is the inverse, calculating the necessary joint angles for a desired end-effector pose. This is known as the *inverse kinematics* (IK) problem and can be framed as a translation from Cartesian space to joint space [9]. IK is a famously nuanced problem with many common struggles regardless of the approach taken to solve it. Manipulators with just 6 degrees-of-freedom (DOF) can have at most 16 valid solutions, manipulators with more than 6-DOF have an infinite solution set. Singularities also occur in configurations where the manipulator looses a DOF, which happens whenever rotation axes become aligned. Numerical methods may also not converge to a solution if they are improperly seeded and fall into a local minimum that isn't representative of the solution. Finally IK solvers are sensitive to the allotted time for calculation, more time can yield a more accurate result but is not always available in real-time systems.

[1][Online]. Available: https://github.com/cjstahoviak/generative-graphik

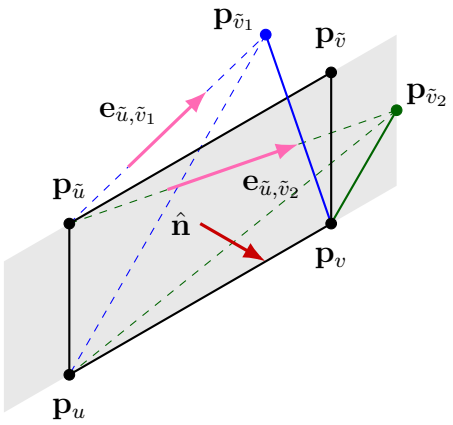

Fig. 1: A diagram that illustrates the issue of *chirality*, non-uniqueness, and the proposed solution (in pink). For non-coplanar pairs of nodes, $(\mathbf{p}_u, \mathbf{p}_{\tilde{u}}, \mathbf{p}_v, \mathbf{p}_{\tilde{v}_1})$ and $(\mathbf{p}_u, \mathbf{p}_{\tilde{u}}, \mathbf{p}_v, \mathbf{p}_{\tilde{v}_2})$, the set of edge distances between these nodes is identical and thus their distance-graphs are identical. This is addressed by not only formulating a distance-graph using distances but also including the $\mathbf{e}_{\tilde{u},\tilde{v}_1}$ and $\mathbf{e}_{\tilde{u},\tilde{v}_2}$ direction vectors for all edges.

Currently, FK and IK have established methods for finding a solution. For FK, the position and orientation of the end-effector of a manipulator with n joints can be calculated as:

$$ {}^0\mathbf{T}_n = {}^0\,\mathbf{T}_1 \cdot {}^1\,\mathbf{T}_2 \cdot {}^2\,\mathbf{T}_3 \cdot \ldots \cdot {}^{n-1}\,\mathbf{T}_n \tag{1} $$

where ${}^0\mathbf{T}_n$ is the homogeneous transformation matrix that describes the end-effector frame relative to the base frame and ${}^{i-1}\mathbf{T}_i$ is the homogeneous transformation matrix that describes the transformation from frame $i-1$ to frame i, in this case, from joint $i-1$ to joint i. In this form, transformation matrices are commonly represented using *Denavit-Hartenberg* (DH) parameters [16]. The final solution produces a transformation matrix, ${}^0\mathbf{T}_n$, in the following form:

$$ {}^0\mathbf{T}_n(\boldsymbol{\theta}) = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2} $$

where $\boldsymbol{\theta}$ is the set of joint angles, $\mathbf{R} = [r_{ij}]$ is a rotation matrix that describes the orientation, and vector $\mathbf{p} = [p_x, p_y, p_z]^T$

Author(s)	Runtime	Manipulator Type	Base Architecture
Daya (2010) [10]	–	2R, Planar	MLP MoE
Duka (2014) [11]	–	3R, Planar	MLP
Bensadoun et al. (2022) [12]	0.0070	4 DOF, Spatial	MLP Hypernetwork
Lu et al. (2022) [13]	0.0218	6 DOF, Spatial	MLP
Stephan et al. (2023) [14]	–	Multi-DOF, Spatial	MLP
Limoyo et al. (2025) [15]	< 0.001	Generalizable, Spatial	EGNN+CVAE

TABLE I: A comparative summary of IK solutions in related literature. Accuracy is not listed a statistic for these papers since almost every paper had their own metric for calculating accuracy. The "generalizable" term in the Manipulator Type expresses that a single model works on all robots. "Multi-DOF" expresses that the model can work on any DOF robot, but model needs to be trained for each one. The most impressive results are found to be from *Generative Graph Inverse Kinematics*, Limoyo et al. (2025).

describes the position of the end-effector relative to the base frame.

IK is solved by either the *numerical* method or the *analytical* method. Analytical methods calculate a complete set of exact solutions using an algebraic formula. They benefit particularly in having fast, constant runtime, the ability find the whole solution set, and are deterministic algorithms. However, they come with some burdensome limitations. The manipulator must satisfy certain conditions such as *Pieper's three adjacent axes intersect* for a 6-DOF wrist [17], inability to solve for $>$ 6-DOF manipulators where the solution set is infinite, and no handling for joint limits or collision avoidance. These limitations can exclude most commercial manipulators from being applicable and warrant using a numerical method instead. In contrast, numerical methods use a "guess" configuration of the manipulator, and iteratively approach the desired pose. Because of this, numerical methods can only ever find a single solution at a time. However, there is no guarantee of convergence, and may get trapped in a local minimum. The iterative nature of numerical methods causes computation time to be significantly slower, especially where high accuracy is requested. The current state-of-the-art (SOTA) in numerical IK solving is *TRAC-IK* which combines *sequential quadratic programming* and *randomized gradient descent optimization* techniques [18].

The relationship between the joint angles and end-effector can be highly non-linear, especially as the DOF of the manipulator arm increases. For many years now, machine learning models have shown to be able to quickly and accurately estimate complex non-linear relationships given a large enough dataset [19]. As discussed in section II, prior work has attempted to use machine learning to solve IK problems with varying success. These attempts often have major weaknesses, such as high inaccuracies, the inability to account for multiple or infinite solutions, and a lack of generalization (models only work on specific manipulators). This study explores a model that has overcome each of those obstacles.

This study investigates *generative graphical inverse kinematics* (GGIK), a robust architecture designed to solve the IK problem and address its limitations [15]. GGIK strictly concerns itself with solving on kinematic structures of certain requirements. The kinematic structure must consist of *revolute joints*, which are joints that allow one DOF rotation around a fixed axis. Within the kinematic structure, all consecutive joints must also have *coplanar rotation axis*, that is to say the rotation axis of all consecutive joint pairs must either be parallel or intersect at some point in space. This is a limitation of distance-graphs which is addressed in section III.

A modified version of the base architecture, generative graphical inverse kinematics plus (GGIK+), is proposed. Three major changes are made: the addition of direction vectors to combat the limitation of distance-graphs, updates to how joint features are represented to incorporate that change, and the addition of an attention mechanism to message passing to attend to those new features. These changes are intended to enhance the data generation process, assist the model on integrating the changes made to distance-graph formulation, and to improve predictions when the structures are non-coplanar. The overall goal is to make GGIK more accurate and more flexible in situations where kinematic structures are more complex.

The remainder of the report is organized in the following order. In section II the current SOTA concerned with solving IK using machine learning is covered. Prior work that established the advanced machine learning techniques used is briefly covered. Next, section III covers the modifications made to the base GGIK architecture. In section IV and section V the results of GGIK+ are discussed. Finally, section VI covers the specific limitations that the modifications have introduced into the overall architecture.

II. Related Works

Analytical and numerical methods for solving IK have been long established, so the review of the SOTA is restricted to learned methods for IK solving. Learned methods for IK solving can involve many different types of architectures, capabilities, and use-cases (generalizable or robot-model-specific). A complete comparison of this review is summarized in Table I.

Duka et al. analyze the performance of a deep neural network (DNN) on a 2D planar 3R manipulator with arbitrary joint limits [11]. A dataset is created by generating uniformly random joint angles within constraints and then computing the position and orientation of the end-effector. The network itself is a feed-forward neural network consisting of 3 inputs,

100 neurons in the hidden layer and 3 neurons in the output layer. They find a best validation RMSE of 0.0054387 when comparing the predicted and desired output.

Daya et al. present a mix of experts (MoE) MLP architecture that is strictly valid for 2R planar manipulators and is capable of identifying multiple solutions [10]. This work attempts to find a solution in each quadrant of 2D space.

Bensadoun et al. present a solution, IKNet, which is a hypernetwork that estimates the parameters of several DNNs. Instead of predicting joint angles, each MLP predicts parameters for a Gaussian Mixture Model [12]. This architecture is tested on several different arm types, ranging from 4 DOF of 7 DOF. Accuracy is measured as the percentage of points that are up to 2 centimeters from the end-effector. For example, IKNet finds an accuracy of $99.5\% \pm 1.3\%$ and a runtime of 0.0070 seconds on the DIGIT 4 DOF arm.

Stephan et al. introduce an unsupervised training method for inverse kinematics on redundant robots using MLPs with loss computed in Cartesian space through a differentiable forward kinematics model [14]. Training data is generated by randomly sampling joint angles and computing end-effector poses; joint-space labels are not required. Their loss includes pose error, joint limit regularization, and an optional loss associated with obstacle avoidance. On a 7 DOF robot, the model is able to achieve a position error of 3.59 cm and rotation error of $4.68°$, with improved performance over supervised MLP baselines for higher-DOF arms.

Lu et al. propose a neural network-based IK solver for general six-axis robots that combines multiple MLPs with a classification model to select between them [13]. Training data is generated by sampling random joint angles, computing end-effector poses, and segmenting the joint space into 192 subspaces. A five-output MLP is used for the first five joints, and a single-output MLP predicts the final joint, followed by Newton–Raphson-based numerical error minimization. On a test set of 4800 randomly sampled poses, the classification system achieves 99.5% accuracy, and the authors report convergence thresholds of 0.001 mm position error and 0.001 rad orientation error for final predictions.

Limoyo et al. introduced the *Generative Graphical Inverse Kinematics* (GGIK) framework, which has been recognized as a leading approach in the field and is the subject of improvement in this report [15]. They frame the IK problem as a distance geometry problem in which desired poses and solutions are represented as distance-graphs [20]. GGIK can be considered a conditional variational autoencoder (CVAE). It uses several modules of equivariant graph neural networks (EGNNs) and linear variational autoencoders to learn a latent space of IK problems and a diverse set of IK solutions [21]. GGIK is also trained on a set of randomly generated kinematic structures of varying link lengths and axis of rotation. This approach allows a single model to generalize across various robot geometries and degrees of freedom. Since a CVAE uses Gaussian mixture model distributions, GGIK is also able to produce unlimited solutions for a single desired pose. In evaluations, GGIK achieved a mean position error of under 6mm and a mean orientation error of under 0.4 degrees on several commercial manipulators, including the Kuka IIWA and Franka Emika Panda. Additionally, GGIK has the capability to generate 1,000 IK solutions in approximately 25 milliseconds, making it well-suited for applications that require real-time and diverse sampling.

III. Methods

The presented architecture, GGIK+, maintains almost all the core functionality and structure of the base GGIK architecture. The improvements discussed in this section include changes to distance-graph theory, feature representation and the addition of an attention mechanism to handle those features.

A. Distance-Graph Improvement

A major limitation of the base GGIK architecture is that it is not "truly" generalizable to all robots. The base GGIK architecture creates graphs using the distance-graph method which is limited to kinematic structures whose consecutive joints are always coplanar [20]. Although almost all commercial manipulators fall within this restriction, it does exclude more unconventional manipulators and complex robots such as humanoid and quadruped robots.

Traditionally a distance-graph is defined as $G = (V, E, d)$ where V is a set of vertices, $E \subseteq V \times V$ is a set of edges connecting all pairs of vertices, and $d : E \rightarrow \mathbb{R}^+$ is a function that assigns a positive weight to each edge equal to the Euclidean distance between both vertices. This theory is then applied to robot kinematic structures in which each joint is represented as a pair of points [20]. This definition is limited because it only allows for unique distance-graphs to be created for kinematic structures whose joints are coplanar. In the case of pairs of joints that are not coplanar, a single distance-graph can represent two different kinematic structures. A visualization of this issue of chirality and non-uniqueness is shown in Figure 1.

To overcome this, a modification to the distance-graph formulation is proposed. The distance-graph edges not only include the Euclidean distance between its vertices but also a direction vector that points from one to the next. This solution is also visualized in Figure 1. The addition of edge direction ensures that even for pairs of joints that are non-coplanar, only a single distance graph can be representative of a single structure.

B. Dataset Generation

To train the model, datasets were prepared by randomly generating kinematic structures of 5, 6, 7, or 8 DOF. A kinematic structure is defined by randomizing link lengths and axis of rotation. For each sample, the joint angles were randomized within their valid range to configure the robot in a random pose. These angles were then passed through a forward kinematics function to calculate the position and orientation of each joint, as well as the end-effector.

Using this data, two graphs were created per sample. The input, or partial graph $\tilde{G}$, contained the known, fixed-structure

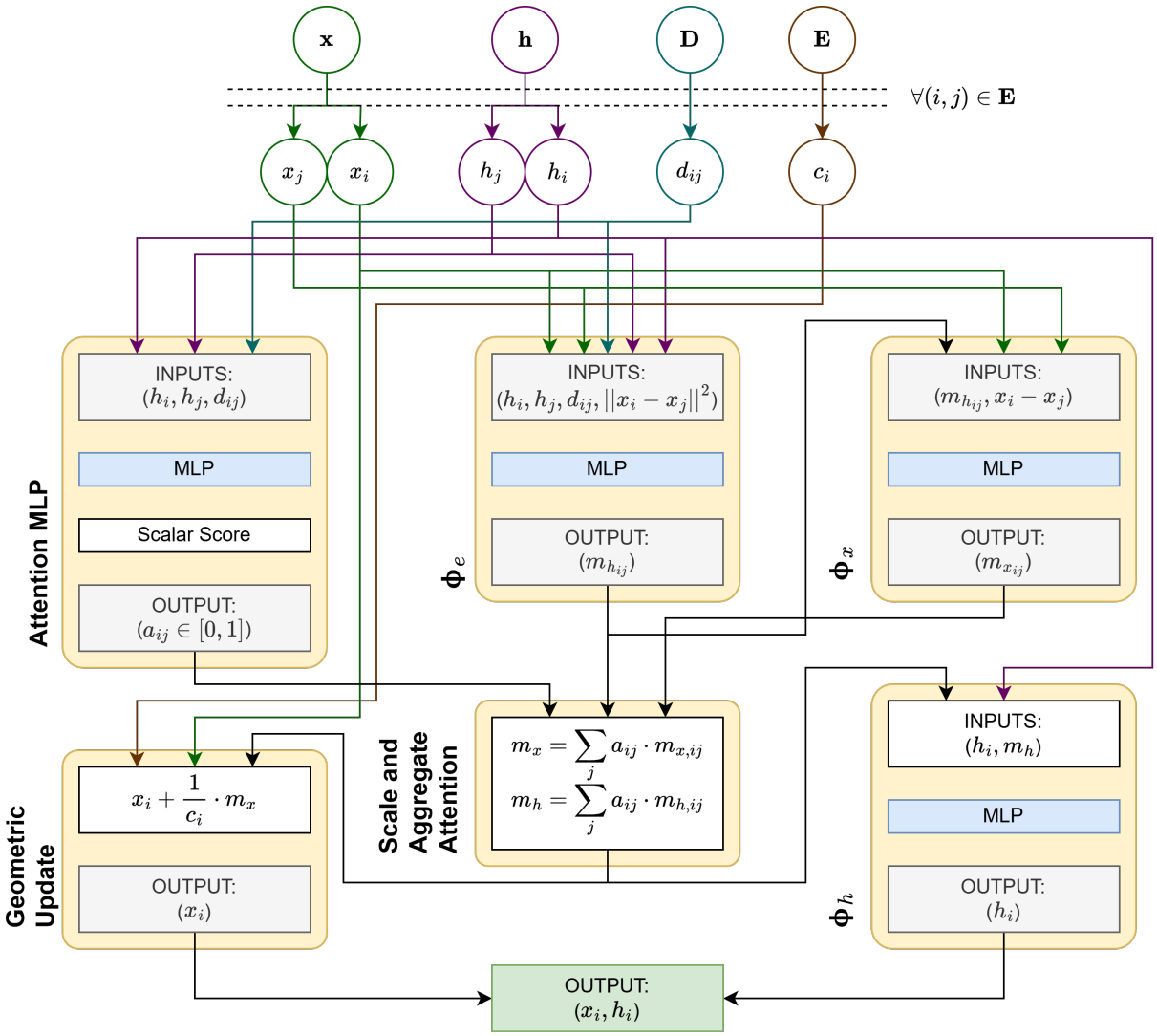

Fig. 2: The message passing algorithm for a single E(n) equivariant graph attention (EGAT) layer. In this diagram $\mathbf{x}$ is the set of node positions, $\mathbf{h}$ is the set of node type, $\mathbf{D}$ is the set of edge attributes, and $\mathbf{E}$ is the set of edge indices. The process is repeated for each pair of nodes, or each edge in the set $\mathbf{E}$ until completion.

of the robot and the pose end-effector relative to the base. The output, or complete graph G, contains all of the information in $\tilde{G}$ as well as all joint positions and edge relationships between each joint. This pairing of $\tilde{G}$ and G was used to train the generative model to infer full joint configurations from a desired end-effector pose.

A total of 5120 samples were generated by varying link lengths and rotation axis, for each of which a single random IK problem was posed. A dataset was created with coplanar joint layouts to test the baseline, and a non-coplanar joint dataset using the modification to the distance-graph was also used to better test the impact of the additional information from the direction vectors.

C. Model Overview

This project builds on the Generative Graphical Inverse Kinematics (GGIK) framework, which uses an Equivariant Graph Neural Network (EGNN) to estimate joint angles based on the structure of a robot and the target position of its end-effector. In GGIK, each node in the graph is associated with a robot joint, and each edge represents a connection between joints, with features based on spatial relationships. The base GGIK architecture uses identical encoders and decoders and treats all neighbors equally when passing messages between nodes.

In GGIK+, both the encoder and decoder were replaced with a modified graph neural network that adds attention-based message weighting and directional edge vectors. These modifications were designed to help the model better focus on relevant joints and distinguish spatial arrangements, especially for non-coplanar kinematic structures. The same architecture was used for both encoding and decoding to keep the learned representations consistent across the pipeline.

D. Attention and Directional Features

GGIK+ introduces two key additions: attention and direction vectors. Attention helps the network focus on the most relevant joint relationships [22]. For each edge between two nodes, a small neural network computes an attention score based on the features of both nodes and their edge attributes. This score is normalized using softmax across all neighbors and used to scale the messages passed along each edge.

Direction vectors, defined as the difference in position between the connected nodes ($x_j - x_i$), provide information about spatial orientation that scalar distance alone cannot capture. These vectors are passed into the network alongside the messages to help the model distinguish joint configurations that may be close in distance but differ in direction. These features work together to give the model better awareness of the robot's structure.

E. Training Setup

GGIK and GGIK+ can be considered Conditional Variational Autoencoder (CVAE) frameworks. This approach allows the model to learn a distribution over possible joint configurations conditioned on a goal and partial structure, making it suitable for tasks with multiple or infinite valid solutions. The loss function included both a reconstruction term (mean squared error between predicted and target joint positions) and a Kullback–Leibler (KL) divergence term to regularize the latent space. A β scaling schedule was used to gradually introduce the KL term during training, helping the model prioritize accurate reconstruction in early epochs and later learn a smoother latent distribution. The evidence lower bound (ELBO) loss function, which includes a KL divergence term, is defined as follows:

$$\begin{aligned}\mathcal{L} = \mathbb{E}_{q_\phi(Z|G)}\left[\log p_\gamma(G|\tilde{G}, Z)\right] \\ -\beta \mathrm{KL}\left(q_\phi(Z|G) \parallel p_\gamma(Z|\tilde{G})\right)\end{aligned} \tag{3}$$

where Z is the latent distribution space, β is a KL divergence scaling factor that increases throughout training, $q_\phi(Z|G)$, $q_\gamma(G|\tilde{G}, Z)$, and $q_\gamma(Z|\tilde{G})$ are the distributions that the training phase seeks to optimize.

F. Implementation Notes

To support attention and direction-aware message passing, changes were made to both the encoder and decoder networks. A learnable attention module was added to calculate attention weights for each edge. These scores were used to scale the spatial and hidden messages before aggregation. Directional vectors were passed into the spatial message function to improve spatial awareness.

The neural networks used to compute spatial and hidden messages were expanded to handle larger inputs, including the direction vector. The output size of the hidden message network was also increased from 3 to 64 dimensions to allow for richer representations. These updates were applied consistently to both the encoder and decoder so that the same message-passing logic was used in both parts of the model.

G. Message Passing and Attention

Message passing in GGIK+ follows the EGNN framework, but adds attention weighting and directional features (see Figure 2). For each edge from node i to node j, the process starts with the function ϕ_e, which creates a hidden message $m_{h_{ij}}$ using the node features, the squared distance between x_i and x_j, and the edge attributes, including the direction vector.

Then, the function ϕ_x takes this hidden message and the direction vector $d_{ij} = x_j - x_i$ to produce a spatial message $m_{x_{ij}}$. At the same time, a small attention network computes an attention weight a_{ij} using h_i, h_j, and the edge features. This value is normalized with softmax across all neighbors of node i.

The hidden and spatial messages are scaled by a_{ij}, and the model sums these weighted messages over all incoming edges to produce the aggregated messages m_h and m_x. After aggregation, the function ϕ_h updates each node's feature h_i using the original feature and m_h, and the node's position x_i is updated using m_x, optionally divided by c_i, the number of neighboring nodes.

As shown in Figure 2, the message-passing process is repeated for all edges in the graph at each layer, allowing GGIK+ to gradually refine node positions and features based on attention and directional context.

IV. Experimentation

The base GGIK architecture is compared to GGIK+ over a series of experiments that evaluate its position accuracy, rotation accuracy, and training behaviors. For context, GGIK was originally published with results from a model trained on 4096000 randomly generated kinematic structures and trained over 300 epochs. Limitations in computing power restricted the training of the models to 5120 randomly generated kinematic structures over 50 epochs.

Two different datasets were prepared, a coplanar and a non-coplanar dataset. Also, two different architectures, base GGIK and GGIK+. The original thesis primarily planned to compare GGIK on coplanar data, as presented in the original publication, against GGIK+ with non-coplanar data. However, both GGIK and GGIK+ can be trained on either dataset, so a total of four models were prepared for a more in depth analysis. This includes the base GGIK architecture on each dataset, referred to as "egnn coplanar" and "egnn non-coplanar", and the GGIK+ architecture on each dataset, referred to as "egat coplanar" and "egat non-coplanar" in figures. The addition of these two other models, egnn non-coplanar and egat coplanar, which were expected to perform poorly, introduced surprising results. Training was conducted on a single *GeForce RTX 4070 Ti Super*. Each model required about 6 hours to train.

A. Loss Curves

During training, the loss function was tracked at each epoch to generate the loss curves. The loss function, standard to CVAE models, consists of a reconstruction term and KL divergence term as described by Equation 3. Both the reconstruction

TABLE II: Minimum Loss Values for All Models

	Training			Validation		
	Pose Loss	KL Loss	Total Loss	Pose Loss	KL Loss	Total Loss
EGNN (coplanar)	1.4955	14.2663	15.7617	1.6027	14.1936	15.7963
EGNN (non-coplanar)	**0.9705**	12.9278	13.8983	**1.0156**	12.9542	13.9698
EGAT (coplanar)	2.4903	25.8403	28.3307	2.4741	27.3254	29.7995
EGAT (non-coplanar)	2.0616	**7.2253**	**9.2869**	2.0264	**7.1660**	**9.1924**

TABLE III: Pose-estimation error statistics for each model.

	Position Error [mm]					Rotation Error [°]				
Model	Mean	Min	Max	Q_1	Q_3	Mean	Min	Max	Q_1	Q_3
EGAT (coplanar)	682.0	20.8	2411.9	438.1	866.2	98.7	0.9	208.9	60.4	138.2
EGAT (non-coplanar)	686.6	30.3	2385.4	447.4	879.5	102.2	1.1	248.9	64.8	141.5
EGNN (coplanar)	486.1	10.9	1582.1	324.1	625.3	53.5	**0.5**	**180.3**	26.2	69.5
EGNN (non-coplanar)	**414.7**	**6.0**	**1499.7**	**270.1**	**539.2**	**51.0**	**0.5**	180.6	**23.4**	**66.7**

term, the KL divergence term and the total loss curves are shown in Figure 3 for two of the four models trained, the base GGIK architecture on coplanar data and GGIK+ on non-coplanar data.

The pose loss of the base GGIK model reaches a lower score than the GGIK+ model. This is further reinforced by the fact that the accuracy of this model is also higher in practice. However, the KL divergence loss and total loss of GGIK+ on non-coplanar data reaches a lower value than all other models in terms of validation loss and training loss. Overall, the base GGIK architecture performs much better on pose loss and the GGIK+ architecture performs much better on KL divergence and total loss.

B. Accuracy

Once training is complete, models are tested on five commercially available manipulator arms. These include some 6-DOF manipulators: Universal Robots UR10 and Schunk LWA4P. As well as a few 7-DOF manipulators: KUKA IIWA, Schunk LWA4D, and Panda. For each of these robot manipulators, 100 random IK problems are generated within the manipulators workspace. For each problem, a position error in millimeters and rotation error in degrees is recorded. These errors are then averaged for each manipulator as seen in Table III. The training results and experimental results averaged over the 500 sample problems are presented and discussed below.

The errors in position and rotation for all 500 sample problems are summarized in the following figures. Figure 4 presents the position error results for all four models categorized by the manipulator the model was tested on. There are a few surprising results here. The average error of the base GGIK models was much lower than the modified GGIK+ models across every robot. Critically, the modified GGIK+ architecture with non-coplanar data performed 34.19% worse in terms of position error than the base GGIK model and dataset. Also, base GGIK improved its performance with the non-coplanar dataset compared to the coplanar dataset, despite the lack of enhancements in GGIK.

V. Discussion

The changes to the model were made to help it focus on important joint relationships and better handle complex layouts. Adding attention to the message-passing steps in both the encoder and decoder was intended to let the model learn which connections mattered most for predicting the full configuration. This allowed it to give more weight to certain joints based on their features and position, rather than treating all neighbors equally. Direction vectors were added to the edge features to give the model a clearer sense of how joints were arranged in space, not just how far apart they were. This was meant to help the model tell the difference between similar distances that had different shapes, especially in non-coplanar setups. Finally, node features were updated to include more detailed input about each joint, which was expected to improve how joint roles were represented throughout the graph. Together, these changes aimed to improve the model's ability to pass useful information through the network and to handle cases where multiple solutions were possible.

The most surprising result of the loss curves was observing the GGIK+ architecture perform better for KL divergence and total loss, while simultaneously having worse accuracy in practice. This indicates that although the model was optimizing better than the base GGIK architecture, it was most likely optimizing for the wrong representation. A lower KL divergence indicates that the distributions $q_\phi(Z|G)$ and $p_\gamma(Z|\tilde{G})$ were much closer in latent space, however in the process of doing this the overall accuracy was compromised.

Despite the failures of the modifications to the EGNN layer, a single improvement on the GGIK architecture was made. The results indicate that the base GGIK architecture performs 15.85% better in terms of position, and 4.78% better in terms of rotation on average when using non-coplanar data than when using a dataset that is limited to coplanar only kinematic structures. Allowing the generation of non-coplanar kinematic structures without modifying how the distance-graph is created, introduced the issue of chirality. The initial hypothesis was that this issue would cause problems generalizing and that a major change to the GGIK architecture and distance-graph

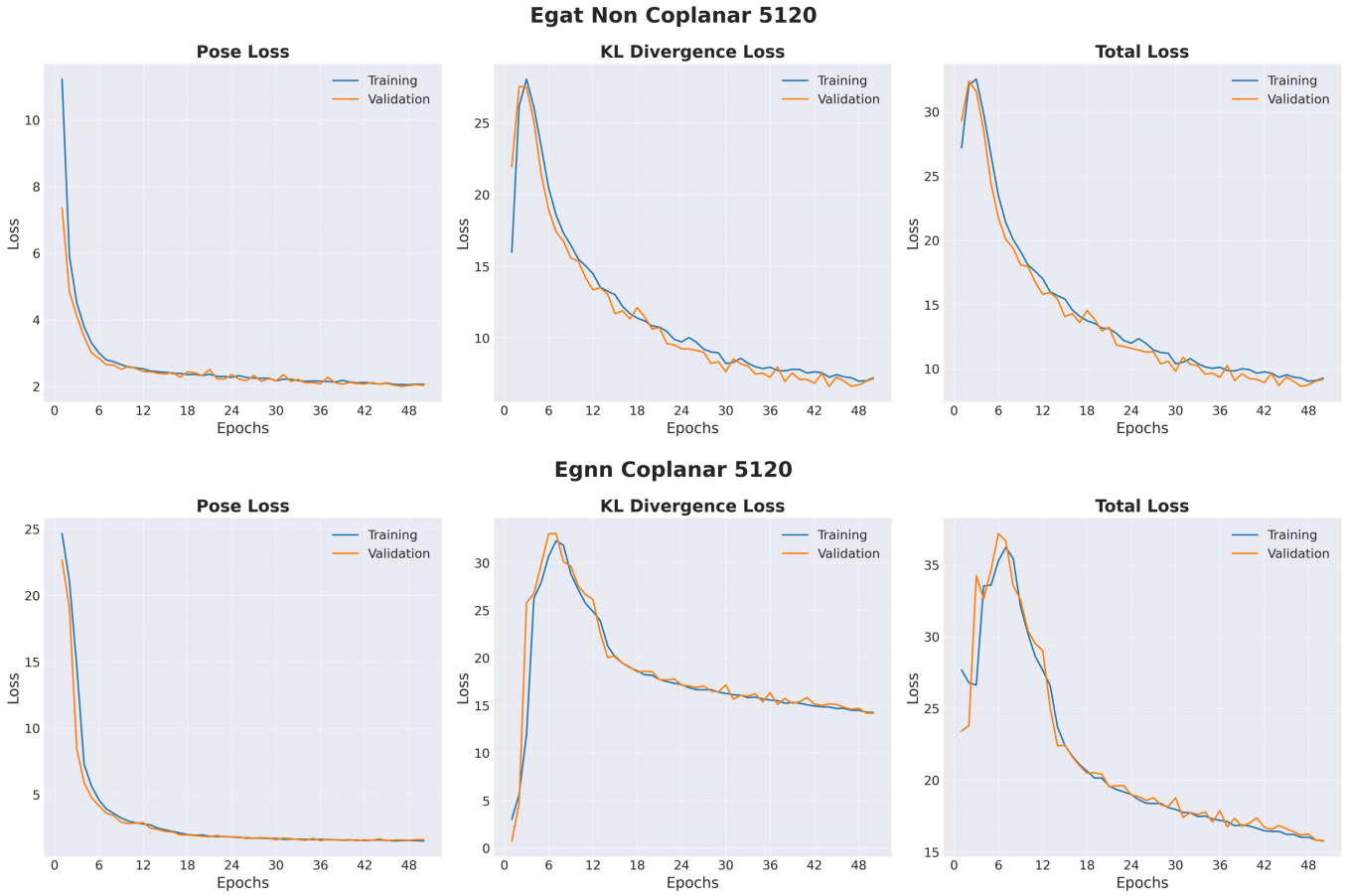

Fig. 3: Training and validation loss curves for GGIK using the coplanar dataset and GGIK+ using the non-coplanar. Both models are trained using 5120 instances for 50 epochs. Loss curves are divided by terms of the loss function: reconstruction loss, KL divergence loss, and total loss.

formulation would be necessary to train on non-coplanar data. However, results indicate that this issue of chirality does not harm the model and actually improves its performance. The issue of chirality in distance-graphs may not be as large of a concern as expected. In theory, this issue means that the model would be unable to differentiate between kinematic structures whose joints rotation axis are mirrors of each other. However, in practice the total configuration space of possible randomly generated kinematic structures is so vast, that the possibility of a pair of kinematic structures that meet this requirement is almost certainly negligible.

VI. Limitations

Although GGIK+ introduced new features intended to improve performance and generalization, its final performance was worse than the base GGIK model. This suggests that the added directional vectors and attention mechanism introduced new challenges that affected training or generalization. Several possible explanations are discussed below.

A. Model Complexity and Training Difficulty

GGIK+ includes more layers and larger hidden features than the baseline. For example, the size of the hidden messages was increased from 3 to 64 dimensions to support directional information and more expressive message passing. While this change aimed to improve the model's ability to capture spatial context, it also made training more difficult. Larger models with more parameters are generally harder to train and can be more sensitive to learning rates and initialization. In GGIK+, no consistent improvement in accuracy was observed, suggesting the added complexity did not result in better convergence.

B. Diversity and Ambiguity Resolution

One of the original goals of the GGIK framework is to handle ambiguity in inverse kinematics by learning a conditional distribution over valid joint configurations. Unlike deterministic models, GGIK is designed to output multiple plausible configurations that satisfy the same end-effector constraint. The modifications were intended to improve this behavior by helping the model understand spatial layout via directional edge vectors and focus on relevant joints via attention.

Fig. 4: Box-and-whiskers plot comparing the accuracy of four separate models. Each model is tested on 500 randomly sampled IK problems evenly distributed over five commercial manipulators. The position and rotation error in the end-effector between the actual solution and predicted solution is plotted.

However, the diversity and distribution quality of the sampled outputs were not measured. Without additional metrics to assess the variety or coverage of the predicted joint layouts, it is unclear whether GGIK+ led to more meaningful interpretations of latent samples. While the model likely produced valid results, it is unclear whether it improved ambiguity resolution or simply favored one type of solution. Future work could analyze output diversity using latent space traversal or by comparing multiple sampled outputs per input.

C. Handling of Non-Coplanar Configurations

A portion of the dataset was generated using non-coplanar joint layouts to test whether the model could generalize to more complex kinematic structures. To support this, the model was given directional edge vectors as part of the message passing inputs. These vectors were expected to help differentiate between joints that are close in distance but oriented differently in space.

However, non-coplanar inputs increase geometric complexity and variability in the edge attributes. This may have made training more difficult, especially without normalization or edge-specific regularization. While directional vectors were included to support non-coplanar cases, whether they improved results specifically for these examples was not evaluated.

D. Gaps in Evaluation

GGIK and GGIK+ were evaluated using ELBO, which combines reconstruction loss and KL divergence loss. This is useful but does not capture all aspects of performance, such as diversity of sampled outputs, ability to generalize, or accuracy on non-coplanar cases. Because inverse kinematics allows multiple correct solutions, a more complete evaluation would involve sampling multiple outputs and checking whether they all satisfy the forward kinematics constraint and represent varied but valid configurations.

The extent to which the attention mechanism consistently focused on the most important joints or whether directional vectors improved message relevance was also not evaluated. These components were trained end-to-end, but their contributions to final performance remain unclear without further

analysis or ablation studies. In addition, the latent space was not closely examined outside of KL divergence.

VII. Conclusion

GGIK+ is presented, a modification to the GGIK architecture specifically designed to overcome the requirement that the rotation axis of consecutive joints must be coplanar, and simultaneously improve accuracy. The benefits of integrating attention into the EGNN layers used in GGIK was explored. The features introduced in the GGIK+ model—attention, directional vectors, and deeper message functions—were designed to improve the ability to resolve ambiguity and handle spatial complexity. The results indicate that our modifications had mixed effects on the overall architecture. However, the modifications to the original dataset, including making it support randomized non-coplanar kinematic structures, improved the performance of the base GGIK architecture.

In the future, other modifications could be made to the GGIK architecture with the goal of improving accuracy even more. Ideally, a learned approach to IK should be able to at least match numerical methods if it wants to see viability in practice. Currently GGIK offers 4-6mm error on most manipulators while numerical methods offer < 0.2mm error.

EGNN layers have a critical role in performance, however other layers types such as graph attention networks (GAT), and graph convolution networks (GCN) may still bring value. A hybrid approach to layer types is not uncommon and careful concatenation of these types could be beneficial. Especially since the input to GGIK (distance-graphs) has many different kinds of features, specialized layers can process a subset of those features that they work best on.

GGIK has already been shown to be extremely responsive to curated datasets, a biased data generation algorithm which prioritizes stable or "upright" configurations that could be beneficial in practice. For example, humanoid and quadruped robots almost always need to maintain a configuration that is balanced and prevents itself from tipping over. Biased datasets could be generated by incorporating the center of mass and manipulability of robots as well.

Finally, Ho et al. show that cascaded diffusion models are able to progressively increase the resolution of an image by seeding cascaded models with the output of the previous model instead of random noise [23]. As GGIK follows a CVAE framework which also utilizes distributions, cascading GGIK models could be successful in pushing GGIK to high precisions that are comparable to numerical methods or increasing accuracy on high DOF robots. Although this would compromise inference time, it could be beneficial to applications that are not under a time constraint.

References

[1] A. A. Hayat, R. O. M. Sadanand, and S. K. Saha, "Robot manipulation through inverse kinematics," in *Proceedings of the 2015 Conference on Advances In Robotics*, ser. AIR '15. New York, NY, USA: Association for Computing Machinery, 2015. [Online]. Available: https://doi.org/10.1145/2783449.2783497

[2] C. Welman, *Inverse kinematics and geometric constraints for articulated figure manipulation*. Simon Fraser University, 1993.

[3] M. H. Sayour, S. E. Kozhaya, and S. S. Saab, "Autonomous robotic manipulation: Real-time, deep-learning approach for grasping of unknown objects," *Journal of Robotics*, vol. 2022, p. 2585656, June 2022, academic Editor: Weitian Wang. [Online]. Available: https://doi.org/10.1155/2022/2585656

[4] S. Qiu and M. R. Kermani, "Inverse kinematics of high dimensional robotic arm-hand systems for precision grasping," *J. Intell. Robot. Syst.*, vol. 101, no. 70, 2021. [Online]. Available: https://doi.org/10.1007/s10846-021-01349-7

[5] Q. Jiang and V. Kumar, "The inverse kinematics of cooperative transport with multiple aerial robots," *IEEE Transactions on Robotics*, vol. 29, no. 1, pp. 136–145, 2013.

[6] J. Beaudoin, T. Laliberté, and C. Gosselin, "Inverse kinematics strategies for physical human-robot interaction using low-impedance passive link shells," *Robotica*, vol. 40, no. 12, p. 4555–4569, 2022.

[7] J. Colan, A. Davila, K. Fozilov, and Y. Hasegawa, "A concurrent framework for constrained inverse kinematics of minimally invasive surgical robots," *Sensors*, vol. 23, no. 6, p. 3328, 2023.

[8] L. Bai, J. Yang, X. Chen, P. Jiang, F. Liu, F. Zheng, and Y. Sun, "Solving the time-varying inverse kinematics problem for the Da Vinci surgical robot," *Applied Sciences*, vol. 9, no. 3, p. 546, 2019. [Online]. Available: https://doi.org/10.3390/app9030546

[9] J. J. Craig, *Introduction to Robotics: Mechanics and Control*, 3rd ed. Upper Saddle River, NJ: Pearson Prentice Hall, 2005.

[10] B. Daya, S. Khawandi, and M. Akoum, "Applying neural network architecture for inverse kinematics problem in robotics," vol. 03, no. 3, pp. 230–239. [Online]. Available: http://www.scirp.org/journal/doi.aspx?DOI=10.4236/jsea.2010.33028

[11] A.-V. Duka, "Neural network based inverse kinematics solution for trajectory tracking of a robotic arm," vol. 12, pp. 20–27. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2212017313006361

[12] R. Bensadoun, S. Gur, N. Blau, T. Shenkar, and L. Wolf, "Neural inverse kinematics." [Online]. Available: http://arxiv.org/abs/2205.10837

[13] J. Lu, T. Zou, and X. Jiang, "A neural network based approach to inverse kinematics problem for general six-axis robots," *Sensors*, vol. 22, no. 22, 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/22/8909

[14] B. Stephan, I. Dontsov, S. Müller, and H.-M. Gross, "On learning of inverse kinematics for highly redundant robots with neural networks," in *2023 21st International Conference on Advanced Robotics (ICAR)*, 2023, pp. 402–408.

[15] O. Limoyo, F. Marić, M. Giamou, P. Alexson, I. Petrović, and J. Kelly, "Generative graphical inverse kinematics," *IEEE Transactions on Robotics*, vol. 41, pp. 1002–1018, 2025. [Online]. Available: https://arxiv.org/abs/2209.08812

[16] J. Denavit and R. S. Hartenberg, "A kinematic notation for lower-pair mechanisms based on matrices," *J. Appl. Mech.*, vol. 22, no. 2, pp. 215–221, Jun. 1955.

[17] D. L. Pieper, "The kinematics of manipulators under computer control," Ph.D. dissertation, Stanford University, Stanford, CA, 1968.

[18] P. Beeson and B. Ames, "Trac-ik: An open-source library for improved solving of generic inverse kinematics," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2015, pp. 928–935.

[19] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA: MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org

[20] F. Marić, M. Giamou, A. W. Hall, S. Khoubyarian, I. Petrović, and J. Kelly, "Riemannian optimization for distance-geometric inverse kinematics," vol. 38, no. 3, pp. 1703–1722. [Online]. Available: http://arxiv.org/abs/2108.13720

[21] V. G. Satorras, E. Hoogeboom, and M. Welling, "E(n) equivariant graph neural networks." [Online]. Available: http://arxiv.org/abs/2102.09844

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[23] J. Ho, C. Saharia, W. Chan, D. J. Fleet, M. Norouzi, and T. Salimans, "Cascaded diffusion models for high fidelity image generation," in *Journal of Machine Learning Research*, vol. 23, no. 47. JMLR, 2022, pp. 1–33.