A Study on Solving the Inverse Kinematics of Serial Robots using Artificial Neural Network and Fuzzy Neural Network

Jacket Demby's*, Yixiang Gao** and G. N. DeSouza***
Vision-Guided and Intelligent Robotics Lab - ViGIR Lab
Department of Electrical Engineering and Computer Science
University of Missouri-Columbia
Columbia, United States
udembys@mail.missouri.edu*, yg5d6@mail.missouri.edu** and DeSouzaG@missouri.edu***

***Abstract*—One of the most important problems in robotics is the computation of the inverse kinematics (IK). This apparently simple task is necessary to determine how to move each joint in order to reach a desired end-effector pose in Cartesian space. However, the associated forward kinematics can be a highly non-linear, non-bijective, and multidimensional function for which it may be difficult or even impossible to find closed-form solutions for its inverse – especially as the number of Degrees of Freedom (DoF) increases. Several approaches have been taken using non-linear approximators to solve IK problems. In this paper, we present a study on solving the inverse kinematics of multiple robotic arms using Artificial Neural Networks (ANN) and Adaptive Neuro-Fuzzy Inference Systems (ANFIS). For this study, we experimented with 4, 5, 6 and 7 DoF serial robots, with combinations of prismatic and revolute joints. Unlike other task-oriented solvers, our goal was not to predict poses based on specific trajectories (linear or otherwise), but instead to learn the entire robot workspaces. This goal better addresses real-world uses of robotic IK, where any end-effector pose should be reachable from any current pose. From the experiments conducted, we conclude that both ANN and ANFIS converged to some degree to the underlying inverse kinematics function, however approximation errors and the time and effort required to achieve those results may not justify their use vis-a-vis other methods in the literature.**

***Index Terms*—Inverse kinematics, Artificial Neural Network, Fuzzy Neural Network, Adaptive Neuro-Fuzzy Inference System**

I. Introduction

Serial robotic manipulators are made of alternating links and joints, where joints are either prismatic (linear) or revolute. Each joint is carefully guided by an actuator (e.g. an electric or hydraulic motor equipped with positional encoders). Generally, the kinematics of such robotic arms can be obtained by a chain structure, where each link is regarded fixed with respect to the next moving link in the chain. Every link in the chain moves according to the specific DoF of the associated joint: i.e. either translational (prismatic) or rotational (revolute). Two important functions arise from this setup and are required for programming robotic arms: the forward and the inverse kinematics. On one hand, the forward kinematics maps the pose (position and orientation) of the end-effector given the joint values, while the inverse kinematics returns the required joint configuration (joint values) needed to achieve a desired pose of the end-effector. Considering the two vectors: $\overrightarrow{D}$ describing the pose (position and orientation of the end-effector) and $\overrightarrow{Q}$ describing the joint configuration of a robotic arm; the forward and inverse kinematics can be expressed as two multivariate, non-linear and not necessarily bijective functions f_{FK} and f_{IK} respectively such that: $\overrightarrow{D} = f_{FK}(\overrightarrow{Q})$ and $\overrightarrow{Q} = f_{IK}(\overrightarrow{D})$, where ideally $f_{IK}(.) = f_{FK}(.)^{-1}$. The forward kinematics (f_{FK}) is relatively simple to compute using the chain structure mentioned above and what is known as the Denavit-Hartenberg (D-H) representation. Through the D-H, we can express the end-effector pose with respect to the base as a succession of translational and rotational transformations incurred by the frames attached to two consecutive links after activating the joint attached to those same links. However, the solution for the inverse kinematics (f_{IK}) is not only particular to a specific robot, but it can be difficult or even impossible to derive for some robots.

The IK solvers in the literature can be divided into four categories: analytical, numerical, hybrid, and data-driven (i.e. learning-based methods) [1]. Analytical methods provide optimum solution, but as the number of DoF of the robot increases, the more complex become the inverse kinematics equations, while for some sequences of revolute and prismatic joints, deriving analytical solutions is again a daunting or even impossible task. Numerical approaches generally use the robot Jacobian, which may not be invertible near singular and redundant configurations. So, some hybrid methods, such as in [2], attempt to tackle these situations by simultaneously considering multiple competitive paths to the solution. Finally, various learning-based methods have been developed over the years to handle the IK problem of generic robotic manipulators and are summarized in Section II [2]–[10]. These methods focus on finding an approximate mapping function from the end-effector pose to the joint configuration, usually in a constrained workspace where the robot operation is required. While approximate, these mappings still need to provide high accuracy and repeatability of the estimated solutions. In this sense, neural networks (NN) are known to be good universal approximators for other problems and hence they should provide good results [5], [7], [9], [10] for unconstrained IK.

The purpose of this paper is to provide an analysis of two NN-based techniques for solving the IK problem: Artificial Neural Networks (ANN) and Adaptive Neuro-Fuzzy Inference Systems (ANFIS). For that purpose, three case studies, applied to four robots with different number of DoF (4, 5, 6 and 7), were considered and their performances were analyzed in terms of precision and accuracy of the predicted joint configuration and the reconstructed end-effector Cartesian poses (i.e. position and orientation). In the first case study, denoted as *All-Joints-ANN*, a single ANN had to learn and predict all joints outputs given the desired pose inputs. In the second case study, denoted as *Individual-Joints-ANN*, a set of NNs were used to learn each joint output given the desired inputs. For the third case study, an ANFIS neural network was trained to approximate the inverse kinematics, also of each joint separately: the *Individual-Joints-ANFIS*.

The rest of the paper is organized as follows: Section II, Background and Related Work, provides a quick survey on learning-based techniques for solving the IK problem, and a discussion on the NN architectures and learning algorithms used in this paper. Section III presents the methods and procedures used for obtaining the datasets (training vs. testing, structured vs. random) and the metrics used for validation. Then, in Section IV, we summarize the experiments performed and provide an analysis of their results. Finally, Section V concludes with insights on the best technique to be used, the reasons for the limitations found, and future work.

II. Background and Related Work

As mentioned above, various IK solvers in the literature employed a data-driven, i.e. a learning-based approach [3]–[9], [11], [12]. Among those, systems that learned the IK functions using some form of Neural Networks (NN) and Support Vector Machines (SVM) were reportedly the most effective ones [4]–[9], [11], [12]. However, most of the scientific literature only targeted the learning of task-oriented trajectories (e.g. linear, planar, circular, elliptic, etc.) instead of learning the entire robot workspace – which leads to a much more complex problem due to the randomness (pattern-free) and vastness of the search space. A recent study by El-Sherbiny et al. [3], for example, applied three soft-computing methods – neural network, fuzzy neural network, and genetic algorithm – to solve the inverse kinematics of a 5 DoF robotic arm. However, the study was limited to a specific spring-shape trajectory, hence not providing deeper insights on the performance of these methods in more generic paths within the robot workspace. Cavalcanti and Santana [4] used a Self-Organizing Map (SOM) with different sizes (18x18, 36x36 and 72x72) to learn the inverse kinematics of Aura robotic arm with only 2 DoF and confined to two well-defined trajectories: linear and planar. Finally, and on a different front, Chen and Lau [6] investigated the use of Support Vector Machines (SVM) to solve the inverse kinematics of a 7 DoF Mitsubishi PA-10 manipulator. But as in the other studies, the results were restricted to a particular robot in a limited workspace.

An initial attempt, presented in [13], has been made with Deep Learning to learn more general, task-independent IK. In that case, a convolutional neural network (CNN) was used to scale up a robot learning to many skills, resulting in the robot being able to perform a more unrestricted pick and place manipulation and without the need for camera calibration and analytical IK solvers. Although the robot was able to grasp objects with different shapes, sometimes only after multiple

TABLE I: *D-H parameters of the 4 robotic arms used in this paper*

i	θ	d	a	α
1	θ_1	400	250	0
2	θ_2	0	150	180
3	0	d_3	0	0
4	θ_4	150	0	0

(a) *4 DoF robotic arm (SCARA)*

i	θ	d	a	α
1	θ_1	380	0	0
2	θ_2	0	280	-90
3	θ_3	0	280	0
4	θ_4	0	80	-90
5	θ_5	0	0	90

(b) *5 DoF robotic arm in [3]*

i	θ	d	a	α
1	θ_1	0	0	-90
2	θ_2	140	0	90
3	0	d_3	0	0
4	θ_4	0	0	-90
5	θ_5	0	0	90
6	θ_6	8.5	0	0

(c) *6 DoF robotic arm (Stanford)*

i	θ	d	a	α
1	θ_1	-275.5	0	90
2	θ_2	0	0	90
3	θ_3	-410	0	90
4	θ_4	-9.8	0	90
5	θ_5	-311.1	0	90
6	θ_5	0	0	90
7	θ_5	-263.8	0	180

(d) *7 DoF robotic arm (Kinova)*

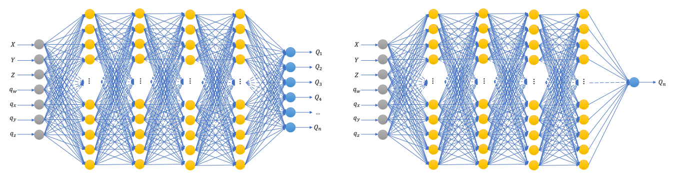
attempts, a large CNN had to be used. Therefore, the training required: 1) a large amount of data with different object types; 2) use of an appropriate CNN architecture; and 3) high-performance computing infrastructure to run the CNN model; all of which limited its practical use. In that sense, a more practical system was proposed by Farzan and DeSouza in [2], who developed a hybrid method consisting of soft-computing and numerical techniques that was guaranteed to converge to an accurate solution, if one existed (i.e. assuming the target joint configuration was non-singular), for any type of robotic arm, and with any number and combination of revolute and prismatic joints.

In summary, learning-based approaches seem to constitute viable and efficient solvers, with one particular example being through the integration of fuzzy logic and neural networks, as for example, in ANFIS. In fact, ANFIS have been successfully used in the approximation of various inverse kinematics ([3], [10], [14], [15]), though limited to 2 DoF, 3 DoF, 4 DoF or at most 5 DoF robotic manipulators and in some cases for position estimation only (i.e. no orientation). Among their claimed advantages, ANN and ANFIS are simpler to implement and to integrate into a robot control system, and do not require heavy computational costs during deployment.

In this paper, we present a study on ANN- and ANFIS-based IK (position and orientation) solvers using four robots with various DoF (Table I). Our goal is to evaluate the potential of these methods under three case studies: *All-Joints-ANN*, *Individual-Joints-ANN*, and *Individual-Joints-ANFIS*. Also, we conducted the studies using task-independent datasets: i.e. a more complete coverage of the robot workspace for the price of a highly pattern-free and much vast search space.

A. Artificial Neural Networks

An ANN with back-propagation learning algorithm allows for the approximation of input-output mappings of multivariate, non-linear functions. At the end of the training, the input vector $\overrightarrow{x}$ and the output vector $\overrightarrow{y}$ represent such mapping, that is: $\overrightarrow{y} = f(\overrightarrow{x})$. The core of the learning begins at the output neurons where the instantaneous error $\overrightarrow{e}$ between desired and predicted outputs is continually calculated and back-propagated through the networks to modify its parameters (weights and biases) according to specific choices for momentum (β) and



(a) All-Joints-ANN (b) Individual-Joints-ANN

Fig. 1: *ANN for the first two case studies, both with 4 hidden layers, 7 inputs neurons ($[X, Y, Z, q_w, q_x, q_y, q_z]$), and outputs $Q_i = d_i$ for prismatic or $Q_i = \theta_i$ for revolute joint, where $i = 1, n$ for $n = 4, 5, 6$ and 7 DoF manipulators: a) All-Joints-ANN, with 6 neurons in the output layer; and b) Individual-Joints-ANN, with only 1 neuron in the output layer (n of these ANNs were required for estimating each of the Q_i's). ANNs were implemented with Keras using TensorFlow backend.*

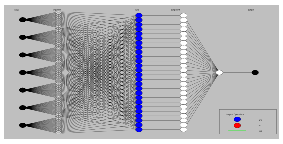


Fig. 2: *Matlab implementation of the Takagi-Sugeno Fuzzy Inference System (ANFIS) used in the Individual-Joints-ANFIS study case with 7 inputs and 1 output. This structure was repeated n-times, one for each DoF of the robots tested.*

learning rate (α).

For this study, the input vector $\overrightarrow{x}$ always represents the full pose of the robot, that is $\overrightarrow{D} = [X, Y, Z, q_w, q_x, q_y, q_z]^T$, where $[X, Y, Z]^T$is the position and $[q_w, q_x, q_y, q_z]^T$ is the orientation of the end-effector expressed in quaternion form. However, the output $\overrightarrow{y}$ can be either all or one of the individual joint configurations. That is, in the *All-Joints-ANN* case, $\overrightarrow{y}$ is the entire vector $\overrightarrow{Q} = [Q_1, Q_2 ... Q_n]^T$ consisting of $Q_i = d_i$ for the prismatic joins or $Q_i = \theta_i$ for the revolute ones. On the other hand, in the *Individual-Joints-ANN* case, a set of NNs are trained and their outputs $\overrightarrow{y_i}$ are either $\overrightarrow{y_i} = [Q_i] = [d_i]$ or $\overrightarrow{y_i} = [Q_i] = [\theta_i]$. In either cases, $i = 1...n$, where n is the number of DoF of the robot, and the goal is to find the IK, or $\overrightarrow{Q} = f_{IK}(\overrightarrow{D})$. The NN used in the *All-Joints-ANN* case is presented in Figure 1a), while the NN for the *Individual-Joints-ANN* case is presented in Figure 1b). The reason to separate into two case studies was to evaluate the effectiveness of an ANN in accurately learning a large robotic workspace on a per-joint vs. all-joint basis. For having a smaller space to learn, we expected the *Individual-Joints-ANN* to perform better than the *All-Joints-ANN.*

B. Fuzzy Neural Network

The Adaptive Neuro-Fuzzy Inference Systems (ANFIS), which are basically five-layer-neural networks, were developed by integrating fuzzy logic into the ANN forward-pass. ANFIS have been found to provide good prediction accuracy in some restricted IK learning problems, as well as faster convergence [3], [14], [15]. The ANFIS structure employed in our study relied on the Takagi-Sugeno Fuzzy Inference System (FIS), depicted in Figure 2. In this case, the input elements were from the pose vector $\overrightarrow{D} = [X, Y, Z, q_w, q_x, q_y, q_z]^T$ and the network output was one of the n joint variables $Q_i = d_i$ or $Q_i = \theta_i$.

The first layer is made of adaptive nodes which do not require incoming weights, but instead directly map the node input to the node output using a node function. This layer's outputs are called the degrees of membership of each input element and it relies on Gaussian fuzzy membership functions applied to the crisp elements of the input pose vector $\overrightarrow{D}$. The outputs from the second layer are referred to as the firing strengths obtained by taking the product of the degrees of membership for each $l-th$ input element. At the third layer, the outputs are called the normalized firing strengths. They are the ratio between the $l-th$ firing strength and the summation of all firing strengths. Finally, the fourth and fifth layers provide the defuzzified predicted outputs and the aggregated defuzzified outputs. The learning algorithm adopted was the hybrid algorithm (a combination of least squares and back-propagation algorithms) [16].

The reason to investigate the ANFIS in a per-joint basis was to compare it with the ANN also on a per-joint basis – i.e. by keeping the search space smaller than for the all-joint case, we contrast the ANN and ANFIS within a per-joint basis, and then against the all-joint case.

III. Methodology

Since an inverse kinematics is specific to the robotic arm, there are no publicly available datasets to be used for learning. Therefore, dataset generation is usually an important step for learning IK. In this study, we used the following procedure to create two datasets for each robot. We start by selecting a robot and generating a number of $\overrightarrow{Q} \rightleftharpoons \overrightarrow{D}$ matching points (mappings) to form: 1) a structured dataset, where joint values were progressively selected within their ranges; and 2) a random dataset, where a dense set of points were randomly chosen within the robots' workspaces. In both cases, the total FK transformation matrix, ${}^nT_0$, obtained from each robot's D-H representation was employed to create the $\overrightarrow{Q} \rightleftharpoons \overrightarrow{D}$ mappings.

The predicted values output by the neural networks were continuously compared to the desired values to adjust the network weights and biases. At the end, the performance of each neural network was evaluated using the Mean Squared Error (MSE) of those same predicted versus desired outputs, that is:

$$MSE(y_i) = \frac{1}{K} * \sum_{k=1}^{K} (e_k)^2 = \frac{1}{K} * \sum_{k=1}^{K} (\hat{y}_{ik} - y_{ik})^2 \qquad (1)$$

where $\hat{y}_{ik}$ and y_{ik} were the desired and predicted outputs i for the training samples $k = 1\, to\, K$.

In Section IV, besides reporting the MSE of the n predicted joint values y_i in the vector $\overrightarrow{Q}$ *(Joint MSE)*, we also reported the *Pose MSE,* which is the error between desired $\hat{\overrightarrow{D}}$ and reconstructed $\overrightarrow{D}$ poses (i.e. reconstructed from the predicted $\overrightarrow{Q}$ using the FK). Here, even though the training was actually performed using quaternions as inputs for the NNs, those same vectors are not very intuitive. So instead, we used the angles Roll, Pitch and Yaw (i.e $\overrightarrow{D} = [X, Y, Z, Ro, Pi, Ya]^T$) to express the errors in orientation of the end effectors.

The procedure above was repeated for all four robots in Table I a)-d). As mentioned earlier, the structured datasets used the ranges in Table II a)-d). The positions of the end-effectors for the structured datasets are depicted in Figure 3 a)-d), while the associated orientations, not shown, were made random for adjacent positions in order to better simulate task-independent situations. As for the random datasets, each robot in Table

TABLE II: *Joint ranges for the structured workspaces*

(a) *4 DoF robotic arm (SCARA)*

i	joint	range
1	θ_1	$[0^o, 360^o]$
2	θ_2	$[50^o, 100^o]$
3	d_3	$[100mm, 250mm]$
4	θ_4	$[30^o, 60^o]$

(b) *5 DoF robotic arm in [3]*

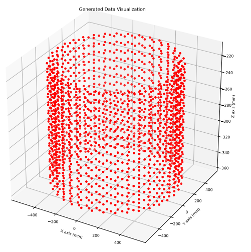
i	joint	range
1	θ_1	$[0^o, 360^o]$
2	θ_2	$[30^o, 60^o]$
3	θ_3	$[20^o, 70^o]$
4	θ_4	$[0^o, 50^o]$
5	θ_5	$[200^o, 240^o]$

(c) *6 DoF robotic arm (Stanford)*

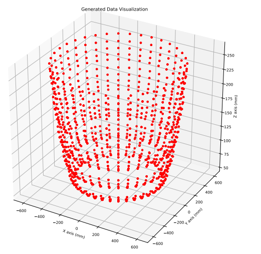
i	joint	range
1	θ_1	$[0^o, 360^o]$
2	θ_2	$[50^o, 150^o]$
3	d_3	$[100mm, 220mm]$
4	θ_4	$[50^o, 90^o]$
5	θ_5	$[200^o, 240^o]$
6	θ_6	$[0^o, 20^o]$

(d) *7 DoF robotic arm (Kinova)*

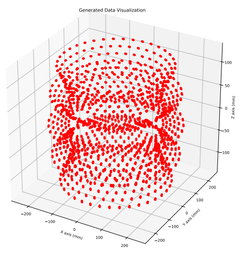
i	joint	range
1	θ_1	$[0^o, 360^o]$
2	θ_2	$[30^o, 60^o]$
3	θ_3	$[90^o, 140^o]$
4	θ_4	$[180^o, 200^o]$
5	θ_5	$[180^o, 200^o]$
6	θ_5	$[0^o, 20^o]$
7	θ_5	$[0^o, 20^o]$



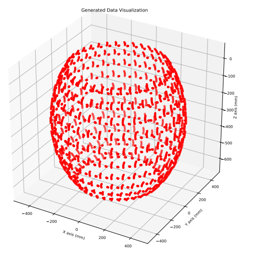
(a) *4860 data points – 4 DoF*



(b) *5400 data points – 5 DoF*



(c) *8640 data points – 6 DoF*



(d) *7200 data points – 7 DoF*

Fig. 3: *Pose of the end-effector for the structured workspaces.*

I a)-d) had their joint values randomly chosen. Due to page limitations, the end-effector positions in the random datasets are not presented here, but the quantitative results will be detailed in Section IV.

IV. Experimental Results

In this section, we present the experimental results. As mentioned earlier, our goal was twofold: 1) to evaluate the performance of ANNs using an all-joint versus per-joint basis – i.e. the *All-Joints* vs. an *Individual-Joints* approaches for the ANN; and 2) to evaluate the performance of *ANN* and *ANFIS* under the same per-joint basis – i.e. the *Individual-Joints* approaches. All three approaches were applied to all four robots using the datasets as described earlier. Both the predicted joint configurations (Joint-MSE) and the reconstructed end-effector cartesian poses (Pose-MSE) were computed. In the next two subsections, we analyze the results for the structured and the random datasets. All experiments were performed using 10% of the datasets set aside for testing.

Structured Datasets: Tables III, IV, V and VI summarize the results for goals #1 (columns 1 vs. 2) and #2 (columns 2 vs. 3), using the structured datasets for robots with 4, 5, 6, and 7 DoF, respectively.

TABLE III: *Results for 4 DoF robot using a structured dataset*

	All-Joints-ANN (min/max)	Individual-Joints-ANN (min/max)	Individual-Joints-ANFIS (min/max)
Joint-MSE in rad^2/m^2			
θ_1	4.175e-4 (0.000e+0/2.280e-2)	**1.054e-4** (0.000e+0/2.158e-3)	1.469e-2 (1.949e-8/1.612e-1)
θ_2	3.632e-3 (3.542e-11/4.058e-2)	5.910e-4 (1.450e-9/9.340e-3)	**2.707e-5** (6.064e-10/2.301e-4)
d_3	1.744e-6 (1.000e-14/2.270e-5)	2.136e-9 (6.400e-15/5.335e-8)	**3.043e-34** (0.000e+0/3.081e-33)
θ_4	2.342e-3 (2.111e-9/2.222e-2)	1.071e-4 (3.629e-11/1.779e-3)	**5.250e-5** (1.499e-12/8.284e-4)
Reconstructed Pose-MSE in rad^2/m^2			
X	6.318e-5 (4.464e-14/1.767e-3)	**7.683e-6** (3.807e-11/7.767e-5)	5.924e-4 (1.016e-9/7.493e-3)
Y	3.796e-5 (1.342e-9/7.781e-4)	**1.250e-5** (1.265e-11/1.764e-4)	1.200e-3 (1.681e-10/2.090e-2)
Z	1.744e-6 (1.000e-14/2.270e-5)	2.136e-9 (6.400e-15/5.335e-8)	**3.043e-34** (0.000e+0/3.081e-33)
Ro	1.152e-3 (1.166e-9/2.577e-2)	**6.152e-4** (8.962e-12/5.879e-3)	1.477e-2 (1.322e-10/1.573e-1)
Pi	2.879e-35 (1.125e-41/4.070e-34)	**4.543e-36** (1.859e-40/5.194e-35)	1.282e-34 (2.888e-41/1.417e-33)
Ya	1.991e-35 (3.749e-42/2.122e-34)	**3.842e-36** (9.033e-41/6.187e-35)	9.119e-35 (2.033e-41/9.062e-34)

TABLE IV: *Results for 5 DoF robot using the structured dataset*

	All-Joints-ANN (min/max)	Individual-Joints-ANN (min/max)	Individual-Joints-ANFIS (min/max)
Joint-MSE in rad^2/m^2			
θ_1	**1.306e-4** (0.000e+0/2.099e-3)	4.690e-4 (0.000e+0/9.104e-3)	3.332e-3 (1.251e-9/4.765e-1)
θ_2	1.033e-6 (2.639e-12/5.120e-5)	**2.030e-9** (2.079e-15/1.738e-8)	1.224e-6 (1.499e-12/5.756e-4)
θ_3	1.054e-4 (3.451e-10/2.594e-3)	3.009e-5 (9.605e-12/6.294e-4)	**5.879e-7** (2.804e-5/2.893e-12)
θ_4	1.381e-4 (0.000e+0/3.057e-3)	1.050e-4 (0.000e+0/1.340e-3)	**5.447e-7** (1.499e-12/9.371e-6)
θ_5	1.294e-4 (4.168e-13/6.564e-3)	2.738e-4 (1.041e-8/9.043e-3)	**1.276e-6** (7.346e-11/2.169e-5)
Reconstructed Pose-MSE in rad^2/m^2			
X	**2.156e-5** (1.059e-10/4.663e-4)	7.303e-5 (8.659e-14/2.321e-3)	1.667e-4 (7.641e-11/9.743e-3)
Y	**2.011e-5** (3.306e-11/4.062e-4)	6.595e-5 (7.190e-11/1.250e-3)	5.704e-4 (2.289e-11/8.013e-2)
Z	5.470e-6 (5.073e-14/1.158e-4)	2.129e-6 (1.132e-13/6.206e-5)	**3.873e-8** (5.269e-14/1.423e-6)
Ro	**1.314e-4** (1.966e-10/3.845e-3)	1.707e-4 (3.944e-11/4.596e-3)	9.198e-4 (2.601e-10/1.799e-1)
Pi	**1.362e-4** (2.796e-11/5.378e-3)	2.787e-4 (2.524e-9/8.654e-3)	5.238e-4 (1.617e-12/3.880e-2)
Ya	**1.032e-4** (1.208e-11/2.304e-3)	2.614e-4 (3.069e-10/5.809e-3)	1.847e-3 (5.777e-12/2.629e-1)

In general, i.e. for all cases studied using the structured datasets, the errors (MSE) in joint and reconstructed pose were not as expected. That is, even though this is a challenging problem – to learn the large search space of task-independent IK scenarios – our expectations were for errors in the order of millimeter for distances and sub-degree for angles. Yet, we observed errors in joint angles as large as 7 degrees in average, and a few cm's for reconstructed positions of the end effector.

Random Datasets: The results above led us to restrict the vast space on which the NNs had to learn the IK. That is, while still keeping the poses random, we limited the workspaces for the next datasets to approximately two quadrants (i.e. joint 1 = $[0 - 180]$) for all robots, at the same time that the other joints were roughly kept the same as in Table II – with the exception

TABLE V: *Results for 6 DoF robot using the structured dataset*

	All-Joints-ANN (min/max)	Individual-Joints-ANN (min/max)	Individual-Joints-ANFIS (min/max)
Joint-MSE in rad^2/m^2			
θ_1	6.474e-4 (0.000e+0/2.740e-1)	**3.908e-4** (0.000e+0/6.586e-4)	2.407e-3 (7.855e-10/1.142e-1)
θ_2	2.295e-4 (5.370e-10/2.758e-3)	6.499e-4 (2.795e-9/1.627e-2)	**4.103e-5** (1.349e-11/5.591e-4)
d_3	3.160e-6 (6.350e-12/5.738e-5)	1.472e-6 (1.369e-13/1.097e-4)	**2.750e-7** (0.000e+0/5.108e-6)
θ_4	3.290e-4 (1.602e-9/9.016e-3)	**1.424e-5** (4.543e-13/9.027e-3)	5.328e-4 (9.288e-10/5.512e-3)
θ_5	2.698e-4 (3.244e-11/3.276e-3)	3.494e-4 (5.178e-11/3.827e-2)	**2.649e-5** (5.931e-10/3.832e-4)
θ_6	**5.070e-4** (0.000e+0/1.624e-2)	5.466e-3 (0.000e+0/3.046e-2)	5.275e-4 (1.261e-9/1.163e-2)
Reconstructed Pose-MSE in rad^2/m^2			
X	9.991e-6 (1.466e-11/2.546e-3)	**6.480e-6** (1.071e-10/2.110e-4)	4.419e-5 (5.038e-11/3.662e-3)
Y	1.143e-5 (3.390e-12/4.488e-3)	**8.054e-6** (2.949e-14/3.338e-4)	4.031e-5 (6.604e-14/1.472e-3)
Z	3.947e-6 (2.855e-13/6.294e-5)	1.188e-5 (5.744e-13/2.791e-4)	**4.796e-7** (1.168e-14/6.569e-6)
Ro	**8.679e-4** (2.687e-11/3.239e-1)	4.904e-3 (1.939e-10/4.061e-2)	1.058e-3 (2.372e-10/2.435e-2)
Pi	**4.179e-4** (4.210e-9/8.106e-2)	4.875e-4 (2.139e-12/7.424e-3)	6.337e-4 (6.883e-10/3.257e-2)
Ya	**2.207e-4** (5.681e-9/4.914e-3)	2.666e-4 (1.003e-10/4.213e-3)	1.202e-3 (9.736e-11/6.775e-2)

TABLE VI: *Results for 7 DoF robot using the structured dataset*

	All-Joints-ANN (min/max)	Individual-Joints-ANN (min/max)	Individual-Joints-ANFIS (min/max)
Joint-MSE in rad^2/m^2			
θ_1	**1.928e-4** (0.000e+0/5.390e-3)	2.053e-4 (0.000e+0/7.417e-3)	9.160e-3 (3.479e-8/7.113e-1)
θ_2	1.188e-4 (2.944e-11/1.584e-3)	2.637e-5 (5.907e-11/3.967e-4)	**5.256e-7** (5.997e-12/7.537e-6)
θ_3	6.782e-3 (1.600e-7/4.759e-2)	5.430e-3 (2.091e-9/2.077e-2)	**3.544e-3** (3.866e-9/2.190e-2)
θ_4	8.136e-6 (2.840e-10/1.668e-4)	4.480e-8 (2.359e-14/5.524e-7)	**3.597e-9** (5.397e-11/7.346e-6)
θ_5	5.676e-6 (9.324e-12/7.803e-5)	1.061e-7 (2.359e-14/3.015e-6)	**2.698e-8** (3.244e-12/3.276e-5)
θ_6	7.440e-3 (2.384e-3/1.306e-2)	7.586e-3 (6.557e-3/8.690e-3)	**1.855e-6** (4.248e-13/1.515e-5)
θ_7	1.553e-2 (0.000e+0/3.046e-2)	7.346e-3 (3.508e-3/1.282e-2)	**3.545e-3** (1.034e-9/1.842e-2)
Reconstructed Pose-MSE in rad^2/m^2			
X	2.258e-4 (6.342e-10/1.726e-3)	**2.154e-4** (1.821e-9/1.221e-3)	2.787e-4 (1.567e-10/2.750e-2)
Y	**2.198e-4** (2.653e-10/1.301e-3)	2.384e-4 (9.914e-11/1.460e-3)	1.156e-3 (3.590e-11/9.004e-2)
Z	6.123e-5 (5.872e-11/5.402e-4)	5.672e-5 (6.077e-11/2.900e-4)	**2.771e-7** (1.110e-12/5.846e-6)
Ro	1.031e-2 (4.560e-8/5.077e-2)	**1.841e-3** (3.018e-9/1.428e-2)	2.986e-3 (2.035e-9/4.633e-1)
Pi	7.950e-3 (7.702e-5/2.489e-2)	8.015e-3 (3.222e-3/2.132e-2)	**5.925e-3** (1.077e-9/5.236e-1)
Ya	2.683e-4 (4.003e-10/2.025e-3)	**2.005e-4** (5.618e-9/1.356e-3)	8.970e-4 (7.625e-12/1.631e-1)

of some slightly larger ranges for the 4-DoF robot. As before, Tables VII, VIII, IX and X summarize the results for goals #1 (column 1 vs. 2) and #2 (column 2 vs. 3), using the random datasets for robots with 4, 5, 6, and 7 DoF, respectively.

Here, the results improved, but only after we increased the training set from a few thousands for the structured datasets to 15,000 in the random datasets. Also, since the search space in which the NNs needed to learn was made smaller, it was expected that the error would also decrease. Nevertheless, none of the performances for these approaches were comparable to

TABLE VII: Results for 4 DoF robot using a random dataset

	All-Joints-ANN (min/max)	Individual-Joints-ANN (min/max)	Individual-Joints-ANFIS (min/max)
Joint-MSE in rad^2/m^2			
θ_1	**6.742e-6** (7.719e-12/3.014e-4)	1.812e-5 (1.401e-11/4.044e-4)	1.560e-4 (7.164e-10/5.206e-3)
θ_2	3.651e-5 (8.648e-13/1.784e-3)	**2.854e-5** (5.401e-14/1.289e-3)	3.020e-4 (6.264e-10/5.108e-3)
d_3	1.414e-8 (5.166e-14/1.131e-6)	8.706e-11 (2.140e-17/2.154e-8)	**7.507e-15** (3.825e-22/2.545e-14)
θ_4	**2.175e-5** (6.889e-13/2.881e-3)	3.476e-5 (3.900e-13/4.256e-3)	3.241e-4 (1.251e-10/5.582e-3)
Reconstructed Pose-MSE in rad^2/m^2			
X	**1.323e-7** (1.822e-13/2.125e-6)	8.042e-7 (3.863e-13/2.666e-5)	7.216e-6 (8.741e-12/2.522e-4)
Y	**1.470e-7** (6.376e-14/6.275e-6)	1.117e-6 (6.875e-13/3.747e-5)	9.965e-6 (8.133e-16/4.766e-4)
Z	1.414e-8 (5.166e-14/1.131e-6)	8.706e-11 (2.140e-17/2.154e-8)	**7.507e-15** (3.825e-22/2.545e-14)
Ro	**1.103e-5** (4.184e-11/3.111e-3)	5.339e-5 (3.190e-13/6.525e-3)	6.695e-4 (1.709e-12/1.076e-2)
Pi	**1.284e-37** (3.620e-49/8.565e-36)	1.731e-37 (9.664e-49/4.950e-36)	2.049e-36 (4.829e-45/3.809e-35)
Ya	**1.273e-37** (7.178e-45/9.278e-36)	1.844e-37 (4.175e-46/5.701e-36)	2.147e-36 (9.253e-44/6.470e-35)

TABLE VIII: Results for 5 DoF robot using a random dataset

	All-Joints-ANN (min/max)	Individual-Joints-ANN (min/max)	Individual-Joints-ANFIS (min/max)
Joint-MSE in rad^2/m^2			
θ_1	**8.060e-3** (1.759e-10/3.208e-2)	8.070e-3 (3.059e-9/3.018e-2)	8.105e-3 (4.610e-9/3.201e-2)
θ_2	**8.079e-3** (4.792e-9/3.003e-2)	8.093e-3 (1.264e-9/2.780e-2)	8.090e-3 (1.577e-8/3.128e-2)
θ_3	5.812e-6 (1.860e-11/1.118e-3)	9.677e-7 (1.593e-14/2.510e-4)	**7.310e-9** (8.278e-15/8.100e-7)
θ_4	6.683e-6 (3.325e-11/7.886e-4)	2.017e-6 (4.370e-14/3.026e-4)	**3.005e-8** (2.456e-15/1.616e-6)
θ_5	1.965e-3 (3.309e-11/1.295e-2)	**1.965e-3** (5.219e-13/1.202e-2)	1.966e-3 (2.023e-10/1.192e-2)
Reconstructed Pose-MSE in rad^2/m^2			
X	4.818e-7 (1.645e-13/1.058e-4)	7.883e-7 (1.780e-15/2.043e-4)	**4.413e-7** (1.531e-13/2.137e-5)
Y	**7.239e-7** (2.797e-13/9.721e-5)	7.791e-7 (1.461e-13/3.675e-5)	9.402e-7 (3.454e-14/5.525e-5)
Z	4.674e-8 (2.014e-15/3.643e-6)	6.732e-9 (9.137e-18/1.749e-7)	**1.039e-10** (1.207e-17/8.057e-9)
Ro	6.816e-6 (6.972e-16/6.088e-4)	3.049e-6 (2.071e-12/8.866e-5)	**1.148e-6** (3.126e-14/2.746e-5)
Pi	7.397e-6 (5.144e-12/7.715e-4)	1.015e-5 (2.903e-15/5.765e-4)	**6.676e-6** (3.848e-13/1.916e-4)
Ya	**1.463e-5** (2.349e-12/3.842e-4)	3.328e-5 (6.891e-13/3.349e-3)	2.903e-5 (1.530e-11/1.123e-3)

hybrid solutions such as the one in [2] (sub-millimeters for positions and tenths of degrees for orientations).

V. Conclusion

This paper presented a study in which three different neural networks were evaluated for solving the IK problem. Different architectures of ANNs and an ANFIS were employed using four robotic arms with 4, 5, 6 and 7 DoF, and the errors measured in terms of predicted joint configurations and reconstructed poses, under a challenging task-independent scenario. The datasets were generated by varying all manipulator joints within certain ranges, in an effort to confine the learned workspace and hence improve the chances of better results.

Unfortunately, the main conclusion drawn from our experiments is that ANNs and ANFIS are apparently not the

TABLE IX: Results for 6 DoF robot using a random dataset

	All-Joints-ANN (min/max)	Individual-Joints-ANN (min/max)	Individual-Joints-ANFIS (min/max)
Joint-MSE in rad^2/m^2			
θ_1	**3.161e-5** (1.803e-11/8.168e-4)	5.147e-5 (2.763e-14/1.498e-3)	4.271e-5 (2.907e-11/1.567e-3)
θ_2	**1.479e-4** (5.265e-13/9.901e-3)	2.368e-4 (8.412e-12/9.688e-3)	2.653e-4 (6.411e-10/1.017e-2)
d_3	5.105e-7 (1.657e-14/1.793e-5)	**3.506e-7** (1.228e-13/1.296e-5)	5.194e-7 (1.474e-13/1.704e-5)
θ_4	**5.053e-5** (1.685e-12/1.359e-3)	7.562e-5 (2.799e-11/3.370e-3)	1.639e-4 (9.757e-12/2.632e-3)
θ_5	**3.309e-2** (1.126e-7/1.059e-1)	3.312e-2 (6.184e-7/1.029e-1)	3.316e-2 (1.153e-7/1.024e-1)
θ_6	3.291e-2 (1.065e-8/1.017e-1)	3.278e-2 (1.514e-4/9.708e-2)	**3.272e-2** (1.470e-6/1.014e-1)
Reconstructed Pose-MSE in rad^2/m^2			
X	**2.120e-7** (2.468e-13/4.434e-6)	4.873e-7 (4.377e-14/1.153e-5)	4.533e-7 (2.947e-12/1.167e-5)
Y	**2.194e-7** (2.723e-13/5.328e-6)	3.753e-7 (2.493e-12/8.731e-6)	4.059e-7 (2.852e-12/1.563e-5)
Z	**1.324e-7** (3.833e-14/3.620e-6)	6.091e-7 (8.738e-13/8.866e-6)	1.012e-6 (2.587e-12/2.219e-5)
Ro	**1.470e-5** (4.037e-12/3.684e-4)	8.152e-5 (3.904e-11/1.174e-3)	1.408e-4 (2.980e-10/1.732e-3)
Pi	**1.590e-5** (7.366e-12/6.308e-4)	5.060e-5 (1.906e-11/1.205e-3)	1.010e-4 (1.468e-11/1.484e-3)
Ya	**1.579e-5** (7.990e-12/4.203e-4)	4.943e-5 (1.393e-12/1.555e-3)	8.242e-5 (1.239e-11/1.951e-3)

TABLE X: Results for 7 DoF robot using a random dataset

	All-Joints-ANN (min/max)	Individual-Joints-ANN (min/max)	Individual-Joints-ANFIS (min/max)
Joint-MSE in rad^2/m^2			
θ_1	**3.239e-4** (4.544e-10/2.303e-2)	3.721e-4 (1.381e-9/2.438e-2)	3.643e-4 (5.613e-12/2.432e-2)
θ_2	**4.594e-5** (1.664e-10/1.076e-3)	9.508e-5 (1.109e-9/1.771e-3)	5.808e-5 (1.600e-11/1.387e-3)
θ_3	**6.027e-4** (1.157e-9/1.907e-2)	6.919e-4 (7.019e-13/2.211e-2)	6.909e-4 (5.479e-9/1.336e-2)
θ_4	2.041e-5 (2.000e-12/2.538e-3)	1.662e-5 (7.667e-11/2.380e-4)	**8.727e-6** (4.304e-11/1.839e-4)
θ_5	4.514e-3 (2.400e-12/2.605e-2)	**4.444e-3** (2.722e-8/2.536e-2)	4.491e-3 (6.254e-11/2.690e-2)
θ_6	3.870e-3 (2.069e-9/2.702e-2)	**3.807e-3** (1.178e-7/2.428e-2)	3.938e-3 (1.528e-8/2.731e-2)
θ_7	3.870e-3 (2.069e-9/2.702e-2)	**3.807e-3** (1.178e-7/2.428e-2)	3.938e-3 (1.528e-8/2.731e-2)
Reconstructed Pose-MSE in rad^2/m^2			
X	**1.440e-6** (5.553e-13/9.361e-5)	4.927e-6 (2.344e-11/6.523e-5)	2.235e-6 (2.408e-12/5.250e-5)
Y	**1.182e-6** (4.018e-12/6.045e-5)	3.615e-6 (1.194e-11/3.923e-5)	2.458e-6 (2.437e-12/6.889e-5)
Z	**5.610e-7** (4.001e-14/1.067e-5)	5.517e-6 (2.236e-12/9.786e-5)	1.156e-6 (2.482e-15/4.351e-5)
Ro	**6.522e-5** (6.228e-13/8.757e-3)	3.410e-4 (1.476e-9/9.604e-3)	1.170e-4 (4.332e-10/5.419e-3)
Pi	**2.370e-5** (2.724e-11/9.293e-4)	1.313e-4 (3.078e-12/1.574e-3)	3.333e-5 (1.424e-11/9.293e-4)
Ya	**1.292e-5** (1.197e-11/7.342e-4)	3.075e-5 (7.408e-11/4.651e-4)	2.290e-5 (5.497e-12/4.218e-4)

best options for solving task-independent IK problems. The vast combination of highly *pattern-free* input vectors makes it quite difficult for neural networks to capture any similarity in the inputs and consequently to learn the high-dimension mapping function between end-effector poses and required joint values. In fact, while further investigation should be carried out before we can categorically dismiss their use in unrestricted IK problems, it was clear from this investigation that the effort required by neural networks may prove any attempt unworthy, given the approximation errors obtained. In that sense, it took an extensive amount of time not only to create an effective training dataset, but also to perform the actual training of the NNs.

In the future, we plan to exploit other NN training algorithms, with different activation functions, different architectures including CNNs and larger training sets to make sure that the best results possible were obtained. But in either cases, it is the authors' opinion that NNs will ultimately require much more effort than a hybrid [2] or even a soft-computing method [3] for similar results, at best.

Acknowledgment

The work presented here started as a class project in Dr. James Keller's course and with the collaboration of Abdulrahman Eldarrat, Guangyu Sun, Jonathan Davis and Thunpisit Amnuaikiatloet, all of which the authors would like to acknowledge and express their gratitude.

References

[1] A. Aristidou, J. Lasenby, Y. Chrysanthou, and A. Shamir, "Inverse kinematics techniques in computer graphics: A survey," in *Computer Graphics Forum*, vol. 37, pp. 35–58, Wiley Online Library, 2018.

[2] S. Farzan and G. N. DeSouza, "From dh to inverse kinematics: A fast numerical solution for general robotic manipulators using parallel processing," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2507–2513, IEEE, 2013.

[3] A. El-Sherbiny, M. A. Elhosseini, and A. Y. Haikal, "A comparative study of soft computing methods to solve inverse kinematics problem," *Ain Shams Engineering Journal*, 2017.

[4] S. Cavalcanti and O. Santana, "Self-learning in the inverse kinematics of robotic arm," in *2017 Latin American Robotics Symposium (LARS) and 2017 Brazilian Symposium on Robotics (SBR)*, pp. 1–5, IEEE, 2017.

[5] A. Csiszar, J. Eilers, and A. Verl, "On solving the inverse kinematics problem using neural networks," in *2017 24th International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*, pp. 1–6, IEEE, 2017.

[6] J. Chen and H. Y. Lau, "Inverse kinematics learning for redundant robot manipulators with blending of support vector regression machines," in *2016 IEEE Workshop on Advanced Robotics and its Social Impacts (ARSO)*, pp. 267–272, IEEE, 2016.

[7] A. R. Almusawi, L. C. Dülger, and S. Kapucu, "A new artificial neural network approach in solving inverse kinematics of robotic arm (denso vp6242)," *Computational intelligence and neuroscience*, vol. 2016, 2016.

[8] H. S. Bidokhti and J. Enferadi, "Direct kinematics solution of 3-rrr robot by using two different artificial neural networks," in *2015 3rd RSI International Conference on Robotics and Mechatronics (ICROM)*, pp. 606–611, IEEE, 2015.

[9] P. Jha and B. Biswal, "A neural network approach for inverse kinematic of a scara manipulator," *IAES International Journal of Robotics and Automation*, vol. 3, no. 1, p. 52, 2014.

[10] S. N. Raptis and E. S. Tzafestas, "Robot inverse kinematics via neural and neurofuzzy networks: architectural and computational aspects for improved performance," *Journal of Information and Optimization Sciences*, vol. 28, no. 6, pp. 905–933, 2007.

[11] Z. Bingul, H. Ertunc, and C. Oysu, "Comparison of inverse kinematics solutions using neural network for 6r robot manipulator with offset," in *2005 ICSC congress on computational intelligence methods and applications*, pp. 5–pp, IEEE, 2005.

[12] A. Guez, "Solution to the inverse kinematics problem in robotics by neural network," in *Proc. Int. Conf. on Neural Networks, Skovde, 1988*, pp. II617–II624, 1988.

[13] C. Finn and S. Levine, "Deep visual foresight for planning robot motion," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2786–2793, IEEE, 2017.

[14] S. Alavandar and M. J. Nigam, "Neuro-fuzzy based approach for inverse kinematics solution of industrial robot manipulators," *International Journal of Computers Communications & Control*, vol. 3, pp. 224–234, Sept. 2008.

[15] J. Narayan and A. Singla, "Anfis based kinematic analysis of a 4-dofs scara robot," in *2017 4th International Conference on Signal Processing, Computing and Control (ISPCC)*, pp. 205–211, IEEE, 2017.

[16] J.-S. Jang, "Anfis: adaptive-network-based fuzzy inference system," *IEEE transactions on systems, man, and cybernetics*, vol. 23, no. 3, pp. 665–685, 1993.