SOLVING COMPLEX GEOMETRY KINEMATICS PROBLEM WITH FUNCTION APPROXIMATION BY ARTIFICIAL NEURAL NETWORK

by

Han Gao

A thesis submitted to The Johns Hopkins University in conformity
with the requirements for the degree of Master of Science in Engineering

Baltimore, Maryland
December, 2022

Abstract

Neural networks have been widely deployed to solve classification and regression problems, and, in recent years, there has been much interest in using neural networks for solving complicated problems. In this paper, a hybrid manipulator kinematics is studied, and an artificial deep neural network is used to solve the inverse kinematics problem. The noise resistance benefit of neural networks approach is explored.

Thesis Readers

Dr. Jin Seob Kim (Primary Advisor)
Senior Lecturer
Dept. of Mechanical Engineering and
Laboratory for Computational Sensing and Robotics
Johns Hopkins University

Dr. Axel Krieger
Assistant Professor
Department of Mechanical Engineering
Johns Hopkins University

Dedicated to my mother

and my late father,

a great engineer and human being,

who I always look up to and love.

Acknowledgements

Words cannot express my gratitude to my mom, who supports me and loves me unconditionally through the hardest and darkest time in our life.

I would like to express my deepest appreciation to Dr. Jin Seob Kim, who supervised and advised my thesis from beginning to end. I was introduced to the wonderful world of machining learning in his Statistical Learning for Engineers class. Through the class and the thesis, Dr. Kim has taught me everything that I know about machine learning and robot kinematics, and given so many valuable advice. His kind words lighten up the dark moments in this journey.

I'm extremely grateful to Dr. Axel Krieger, who took on the role of second reader under such urgent and difficult circumstance.

Many thanks to Co-op program director Mr. David Lee, Mechanical Engineering Academic Program Manager Mr. Mike Bernard, and Sr. Academic Program Coordinator Mr. John Soos. They helped me navigated through the process of the program and graduation. Mr. Lee and Mr. Bernard's supported me in making some of the hardest decision in this process.

I also want to thank my girlfriend, who are always here with me in the journey of graduate school. Her encouragement kept me going when I doubted myself.

Last but not least, special thanks go to my cousin, Ciel Sun, who inspired me to take the approach for this thesis.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Problem background

Robot kinematics is the study of motion in Robotics. Forward Kinematics (FK) and Inverse Kinematics (IK) are two fundamental problems in Robotics. The FK of a robot calculates the position and orientation of its end effector (the tool or point of interest on the end of the robot) in the workspace configuration, given the relative configurations (angle of rotary joints or length of prismatic joints) of adjacent links of the robot. Given a known and desired position and orientation in the workspace configuration, the task of IK is to find the sets of joint angles that produce it [1, 2].

Determining FK and IK of a robot is the first step for understanding its mechanism and studying its application. For majority of the industrial and medical robots, there are well established and well researched FK and IK. For example, extensive research have been done for IK of a 6R industrial robot arm [3], 7R serial robot arm [4], and 6 degree-of-freedom (DoF) parallel manipulators [5]. When a novel robotic mechanism is proposed for an application, understanding the kinematics is a crucial step. For example, in-depth research has been done for the new 4 DoF parallel medical robot proposed [6].

Kinematics, including both FK and IK, is not just studied in robotics. It is also applicable in biology and material science. In [7], IK of 6 DoF biopolymer segment is studied.

While for serial robot manipulators, closed-form FK is relatively simple to obtain, IK is known to be difficult to solve, and at times, impossible to find closed-form analytical solution, due to its non-linearity. Besides a closed-formed analytical approach, numerical methods with interactive processes is another approach. Numerical methods include Jacobian solutions, Newton method, Sequential Monte Carlo Method (SMCM) and so on [8]. Some of these approaches suffer from high complexity, difficulty to implement and high computational cost.

An artificial neural network (ANN), sometimes referred to as Neural Network(NN), is a computing modeling system that mimics the biology of animal brain and its neurons in it. ANN is widely used in classification problems. It has been deployed in cases from bank loan risk assessment to shopping preference prediction [9].

ANN can also be deployed to function approximation tasks as regression problems. ANN has been shown to be effective in nonlinear smooth function approximation [10]. In [11], ANN is successfully implemented to approximate IK of a two link planar robot arm. Compared to traditional numerical method, ANN is faster in real time performance, and resistance to noise in data.

In this thesis, a planer robot case with a complex FK is studied. The robot mechanism is consist of a link and a quadrilateral with two equal sides. This is a serial and parallel hybrid manipulator. The variation of such mechanism is used in flight simulators, surface treatment robots, and other robot manipulators. Figure 1-1 shows the basic structure of the robot manipulator.

The first link rotates around the origin, and the end of the first link is connected to the mid point of the bottom side of the quadrilateral. The point of interest, the end-effector, is located at the mid-point of the top side of the quadrilateral. The quadrilateral can rotate around the end of first link, the connection point between first link and quadrilateral.

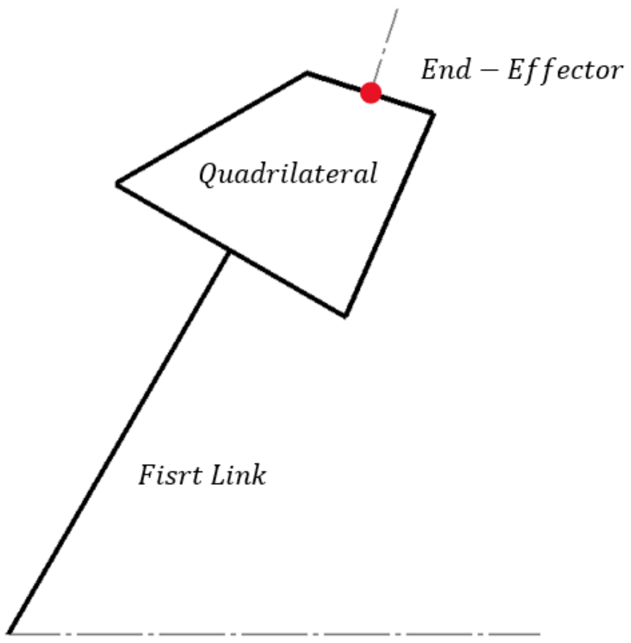


Figure 1-1. Robotic Mechanism Studied in This Thesis

For the thesis, we are interested in the end-effector location, position x, position y, and the facing angle α of the end-effector. Note that the facing angle α is also the normal angle of the top side of the quadrilateral.

Problem Formulation

Denote the length of the first link as L_1, the angle between the fist link and horizontal ground as θ_1. The bottom length of the quadrilateral is $Down$, or D for short, and the length of the two sides is $Side$, or S. The length of the top is Top, or T. The angle between first link and bottom of quadrilateral is denoted as θ_2, and the angle between the side and bottom of the quadrilateral is denoted as θ_3. See Figure 1-2 for graphic illustration.

For the FK, with given θ_1, θ_2, θ_3, we need to find the end-effector position x, position y, and angle α. For IK, the goal is to find of a combination of θ_1, θ_2, θ_3, so that the end-effector is at the given position with a given facing angle. The inputs for the FK function are θ_1, θ_2, θ_3, and the outputs are x, y and α. The inputs for the IK function are x, y and α, and the outputs are θ_1, θ_2, θ_3.

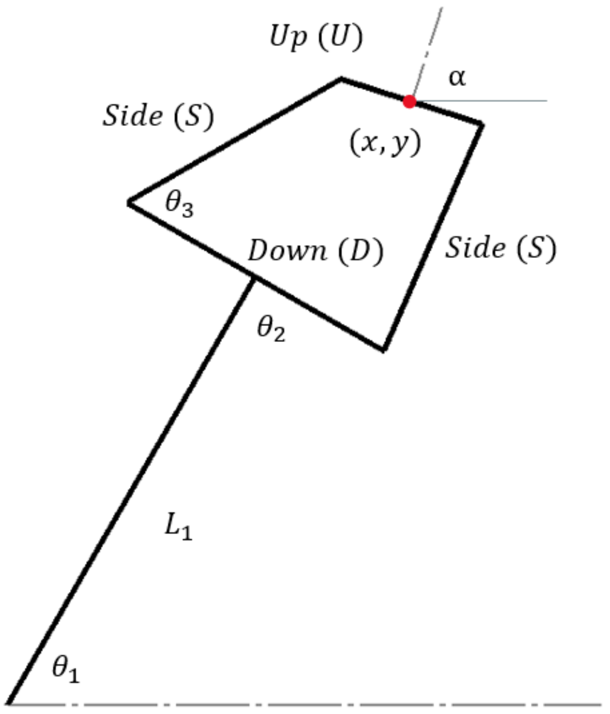


Figure 1-2. Problem Parameters and Variables

Figure 1-3 shows the relation between FK and IK, and calculation method for this robot manipulator.

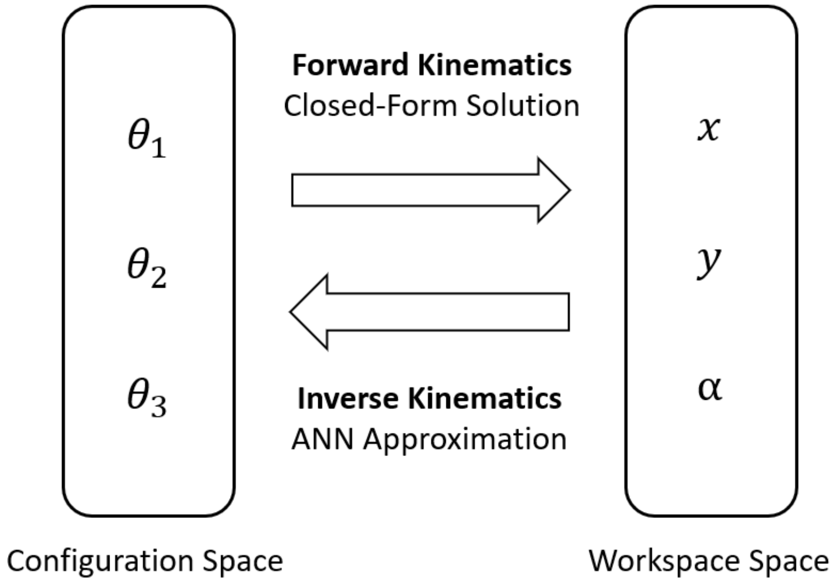


Figure 1-3. Relation between FK and IK

Chapter 2

Forward Kinematics

Method

To calculate the position and angle of the end effector, a construction line, or a virtual link is added by connection the end of first link and end effector (see Figure 2-1). Denote the length of the virtual link as L_2, and the angle between the virtual link and bottom of the quadrilateral as θ_4.

With L_2 and θ_4, position x and y can be calculated as

$$x = L_1 \cos(\theta_1) - L_2 \cos(\theta_1 + \theta_2 + \theta_4) \tag{2.1}$$

$$y = L_1 \sin(\theta_1) - L_2 \sin(\theta_1 + \theta_2 + \theta_4) \,. \tag{2.2}$$

Denote the angle between top and bottom of the quadrilateral as θ_5. The angle α of the end effector can be calculated as

$$\alpha = \theta_1 + \theta_2 + \theta_5 - \frac{\pi}{2}. \tag{2.3}$$

Once L_2, θ_4 and θ_5 are calculated, we will have complete FK.

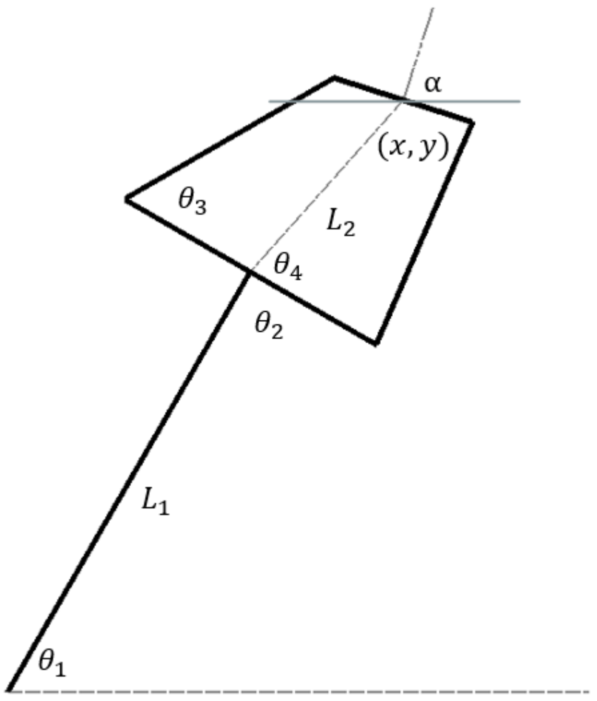


Figure 2-1. Construct the Virtual Link

Calculation of L_2, θ_4 and θ_5 in the quadrilateral

Finding L_2 and θ_4

To calculate L_2 and θ_4 within the quadrilateral, we make two construction lines l_1 and l_2, and denote angles a, b, c, d as shown in Figure 2-2.

Within the the triangle with three sides *Side*, l_1, and $\frac{1}{2} *$ Down, we use Law of Cosine with angle θ_3 to have the following expression.

$$l_1 = \sqrt{\frac{D^2}{4} - DS\cos(\theta_3) + S^2}. \tag{2.4}$$

Within the same triangle, Law of Sines is used to find angle a as

$$a = \frac{l_1}{S}\theta_3. \tag{2.5}$$

Within the the triangle with three sides *Side*, l_2, *Down*, we use Law of Cosine with angle θ_3 as

$$l_2 = \sqrt{D^2 - 2DS\cos(\theta_3) + S^2}. \tag{2.6}$$

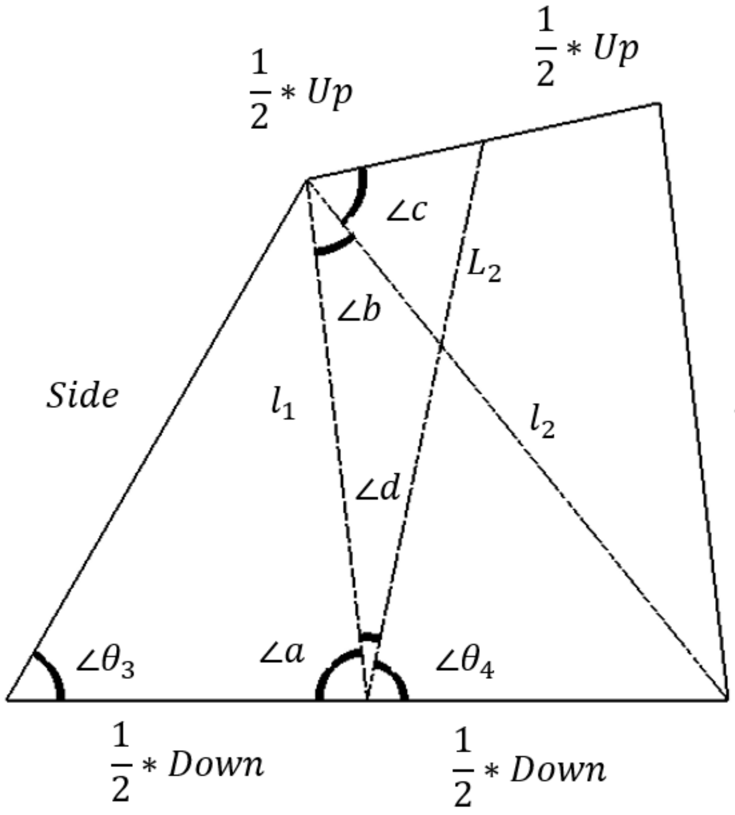


Figure 2-2. Internal Parameters and Variables of the Quadrilateral for Finding L_2, and θ_4

Within the the triangle with three sides Up, $Side$, l_2, Law of Cosine is applied to find angle c as

$$c = \cos^{-1}\left(\frac{l_2^2 - S^2 + U^2}{2l_2U}\right). \tag{2.7}$$

Similarly, we apply Law of Cosine within triangle with l_1, l_2, and $\frac{1}{2} *$ Down to find angle b as

$$b = \cos^{-1}\left(\frac{-\frac{D^2}{4} + l_1^2 + l_2^2}{2l_1l_2}\right). \tag{2.8}$$

Now L_2 can be found by applying Law of Cosine to the triangle with l_1, L_2, and $\frac{1}{2} *$ Up by using l_1, $\frac{1}{2} *$ Up and angle $(b + c)$, as

$$L_2 = \sqrt{-l_1U\cos(\mathrm{b}+\mathrm{c}) + l_1^2 + \frac{U^2}{4}}. \tag{2.9}$$

Within the same triangle, Law of Cosine is used to find angle d as

$$d = \frac{\mathrm{U}}{2L_2}(b+c). \tag{2.10}$$

In addition, one can find that

$$\theta_4 = \pi - a - d. \tag{2.11}$$

Now substitute l_1, angle b and angle c into equation (2.9),

$$L_2 = \sqrt{-l_1 U \cos\left(\cos^{-1}\left(\frac{-\frac{\mathrm{D}^2}{4} + l_1^2 + l_2^2}{2l_1 l_2}\right) + \cos^{-1}\left(\frac{l_2^2 - S^2 + U^2}{2l_2 U}\right)\right) + l_1^2 + \frac{U^2}{4}}$$

$$\theta_4 = -\frac{U\left(\cos^{-1}\left(\frac{-\frac{\mathrm{D}^2}{4}+l_1^2+l_2^2}{2l_1l_2}\right) + \cos^{-1}\left(\frac{l_2^2-S^2+U^2}{2l_2U}\right)\right)}{2L_2} - \frac{\theta_3 S}{l_1} + \pi \tag{2.12}$$

where,

$$l_1 = \sqrt{\frac{D^2}{4} - DS\cos(\theta_3) + S^2}$$
$$l_2 = \sqrt{D^2 - 2DS\cos(\theta_3) + S^2}$$

Finding θ_5

To calculate θ_5 within the quadrilateral, make two construction lines l_2 and p, and denote angles e as shown in Figure 2-3. Line l_2 is the diagonal line of the quadrilateral, and line p is parallel to the Down side.

Since line p is parallel to *Down* side, the two opposite angles of e are equal. l_2 is already calculated in the previous subsection. Angle e is calculated by applying Law of Cosine to the triangle with *Down*, *Side*, l_2 as

$$e = \cos^{-1}\left(\frac{\mathrm{D}^2 + l_2^2 - S^2}{2\mathrm{D}l_2}\right). \tag{2.13}$$

Angle of $e + \theta_5$ can be found by applying Law of Cosine the triangle with *Up*, *Side*, l_2, then θ_5 can be found by subtracting angle e from the sum as

$$e + \theta_5 = \cos^{-1}\left(\frac{l_2^2 - S^2 + U^2}{2l_2 U}\right) \tag{2.14}$$

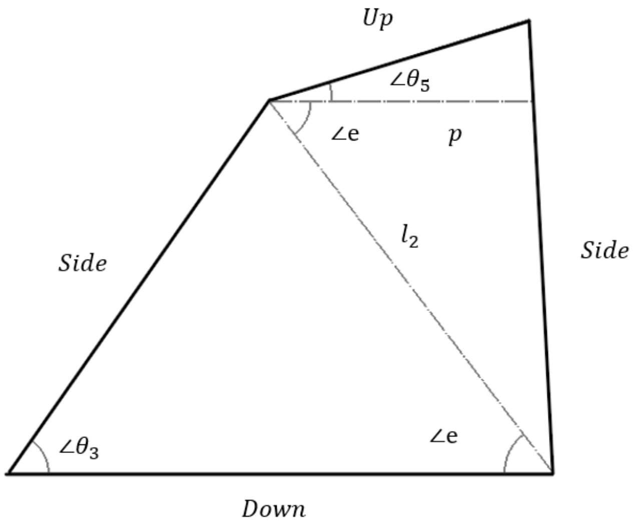


Figure 2-3. Internal Parameters and Variables of the Quadrilateral for Finding α

and

$$\begin{aligned}\theta_5 &= (e+\theta_5) - e \\ &= \cos^{-1}\left(\frac{l_2^2 - S^2 + U^2}{2l_2U}\right) - \cos^{-1}\left(\frac{\mathrm{D}^2 + l_2^2 - S^2}{2\mathrm{D}l_2}\right) \\ &= \cos^{-1}\left(\frac{\mathrm{D}^2 - 2\mathrm{D}S\cos(\theta_3) + U^2}{2U\sqrt{\mathrm{D}^2 - 2\mathrm{D}S\cos(\theta_3) + S^2}}\right) - \cos^{-1}\left(\frac{\mathrm{D} - S\cos(\theta_3)}{\sqrt{\mathrm{D}^2 - 2\mathrm{D}S\cos(\theta_3) + S^2}}\right).\end{aligned} \tag{2.15}$$

Complete Forward Kinematics

Substitute equations (2.12) and (2.15) into equations (2.1) - (2.3) to obtain the complete FK,

$$x = L_1\cos(\theta_1) - L_2\cos(\theta_1+\theta_2+\theta_4)$$

$$y = L_1\sin(\theta_1) - L_2\sin(\theta_1+\theta_2+\theta_4)$$

$$\alpha = -\cos^{-1}\left(\frac{\mathrm{D}^2 + l_2^2 - S^2}{2\mathrm{D}l_2}\right) + \theta_1 + \theta_2 + \cos^{-1}\left(\frac{l_2^2 - S^2 + U^2}{2l_2U}\right) - \frac{\pi}{2}$$

where,

$$L_2 = \sqrt{-l_1 U \cos\left(\cos^{-1}\left(\frac{-\frac{D^2}{4} + l_1^2 + l_2^2}{2l_1 l_2}\right) + \cos^{-1}\left(\frac{l_2^2 - S^2 + U^2}{2l_2 U}\right)\right) + l_1^2 + \frac{U^2}{4}}$$

$$\theta_4 = -\frac{U\left(\cos^{-1}\left(\frac{-\frac{D^2}{4}+l_1^2+l_2^2}{2l_1 l_2}\right) + \cos^{-1}\left(\frac{l_2^2-S^2+U^2}{2l_2 U}\right)\right)}{2L_2} - \frac{\theta_3 S}{l_1} + \pi$$

$$l_1 = \sqrt{\frac{D^2}{4} - DS\cos(\theta_3) + S^2}$$

$$l_2 = \sqrt{D^2 - 2DS\cos(\theta_3) + S^2}$$

FK Validation

The FK is validated by a computer aided design program (CAD), SolidWorks. Multiple combination of parameters and variables are substituted into the obtained FK equations. The CAD results agree very well with the output of FK equations consistently.

Synthetic Data Generation

Data Generation

After obtaining the FK of the robot, we can generate synthetic training data for IK using ANN.

For the purpose of this thesis, we will use the follow parameter for the robot mechanism.

$$L_1 = 50in$$

$$Up\,(U) = 15in$$

$$Side\,(S) = 25in$$

$$Down\,(D) = 30in$$

For the training data, we will sample linearly in configuration space, and the range of motion for θ_1, θ_2 and θ_3 are,

$$\begin{aligned} 0 \leq \theta_1 \leq \frac{\pi}{2} \\ 0 \leq \theta_2 \leq \frac{3\pi}{4} \\ \frac{\pi}{4} \leq \theta_3 \leq \frac{\pi}{2} \end{aligned}$$

The model is loosely based on a surface treatment robot arm, the range of θ_1 and θ_2 is determined by the joint limits. The range of θ_3 is determined by the geometry of the quadrilateral. When $\theta_3 < 0.674$ or $\theta_3 > 1.621$, it will result in invalid quadrilateral geometry.

Two sets of data are generated. The first set is generated without noise, and the second set is generated by adding random Gaussian noise with a mean of 0, and variance of 5 to position x, y, and angle α in degree. Noise with a mean of 0, and variance of 3 is added to θ_1, θ_2 and θ_3 in degree. 21,600 data points are generated for both data set without noise and data set with noise.

Figure 2-4 shows the generated data with no noise in Cartesian workspace (position and orientation of end effector). Figure 2-5 shows the data generated with Gaussian noise in workspace.

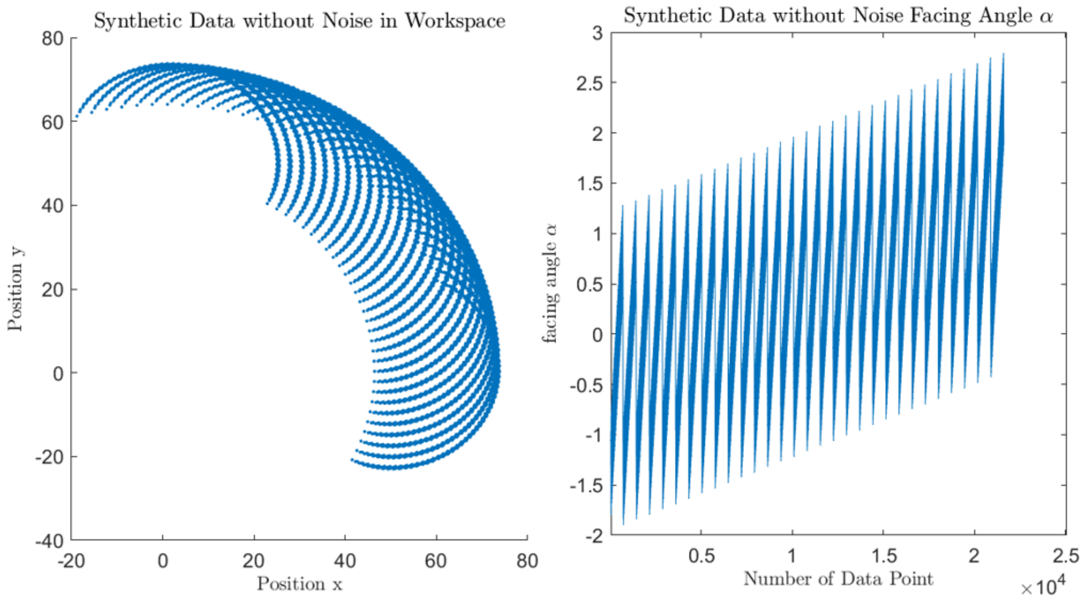


Figure 2-4. Synthetic Data Generated without Noise in Workspace

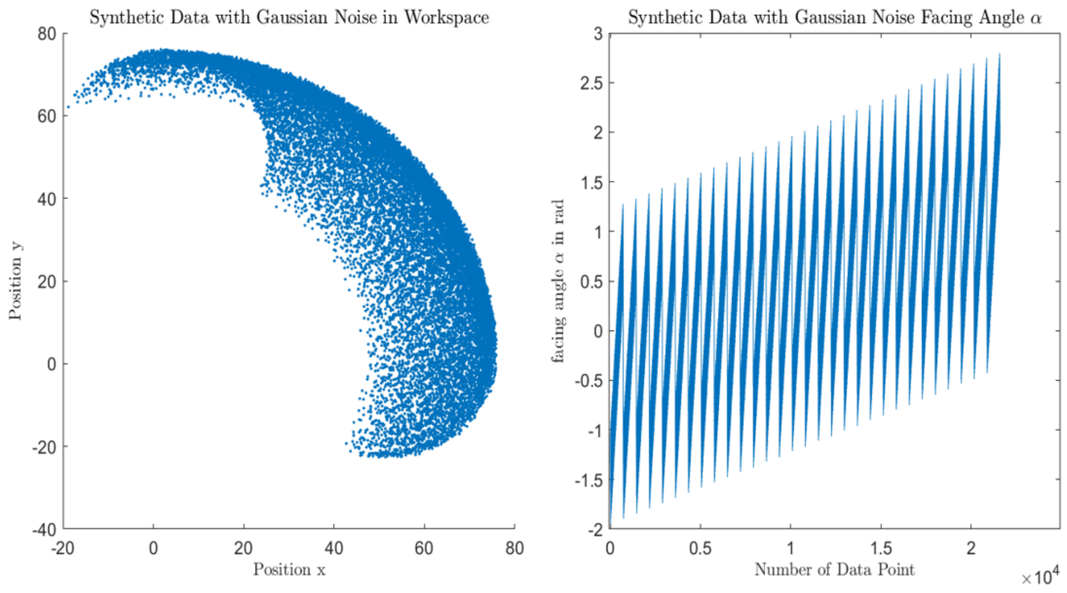


Figure 2-5. Synthetic Data Generated with Gaussian Noise in Workspace

Training, Validation and Test Split

The data set without noise is split into 80% training data and 20% testing data. With the 80% training data, cross validation and gridsearch are conducted to find the fitted hyerparameters for the ANN model. Test data will only be used for evaluation of model after the ANN model is finalized.

Chapter 3

Artificial Neural Network Based Approach for Inverse Kinematics

Introduction

Traditionally, there are two different approaches for IK problems: closed-form analytical approaches and numerical approaches. Closed-form solutions are fast and efficient at calculating the joint angles that produce a desired end-effector configuration. Numerical solutions depend on an interactive procedure, which often requires high computational power, to solve sets of equations [1]. Finding a closed-form solution for IK is known to be a challenging task. Often, it is difficult, if not impossible, to find a closed-form solution. Numerical methods can be time consuming in real time applications. Numerical methods also potentially have the issue of multiple local minima. Therefore, ANN is proposed to solve the IK. Despite the fact that the training process is time consuming and computationally costly, once trained, the ANN can quickly find a set of solution in real time. In this thesis, the ANN approach is also shown to be robust with noise present in data.

An ANN consists of input layer, output layer, and hidden layers. Each layer consists of neurons, and each layer is connected with the two adjacent layers via weights and activation functions. Activation functions mimic the behavior of a neuron transmitting signals in brain. The value of a neuron in the layer passes, together with

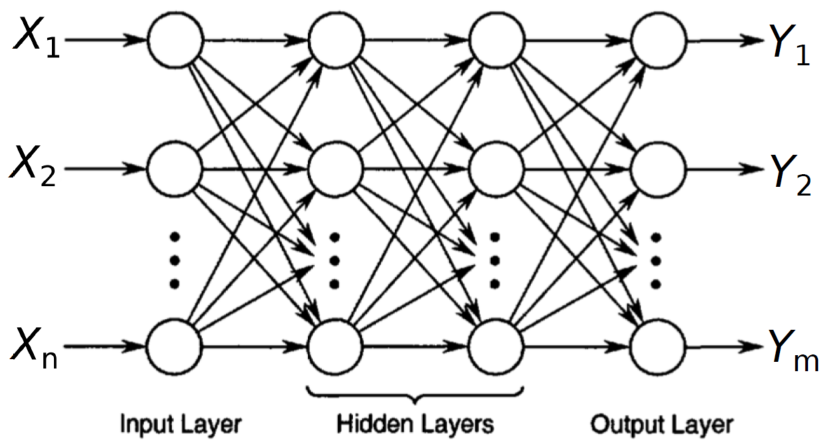


Figure 3-1. A Schematic of an Artificial Neural Network, Modified from [12]

weights, through the activation function, and becomes the input of the next layer. Figure 3-1 shows a basic structure of an ANN.

For the purpose of approximating the IK function, Mean Square Error (MSE) is used as a loss function. The goal of this ANN is minimize MSE of the three output, θ_1, θ_2 and θ_3 combined.

Data Pre-processing

Scaling

Because the scale of inputs determines the effective scaling of the weights in the first hidden layer, it impacts the loss convergence and the quality of training results [13]. Without scaling, the large input value may result in small weights for the first hidden layer, resulting in ineffectiveness for the training process. It has been shown that scaling can speed up the training of neural network, while improving the stability of the convergence. In practice, it is almost always preferred to normalize or standardize both inputs and outputs [14].

There are two general scaling approaches, *Normalization* and *Standardization*. Since the training data are generated by uniformly and linearly sampling in configuration space, we will use *Normalization* to process the data. The input and output data will be scaled to between 0 and 1 separately, because inputs and output have different unit of measure and numerical value range.

To normalize the data, we use *MinMaxScaler* function in *sklearn* library. With *MinMaxScaler* function, output data from an ANN can be scaled back to verify the results.

Training and Testing Split

For the training and validation purpose, the data set without noise is split into 80% training and 20% testing with random selection. In the experiment for data set with noise, the test data without noise is used to evaluate the model.

While searching for hyperparameters for the ANN, 10-fold repeated cross validation is implemented in grid search. Training data is split into 10-fold for cross validation, while test data is left for evaluation after the ANN model is finalized.

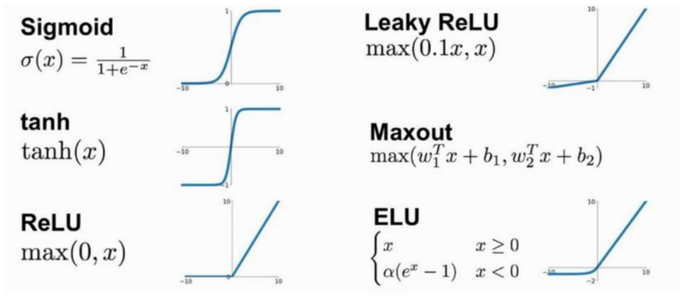
ANN Structure

Number of Hidden Layers

It is preferred to have too many layers to begin with than to few layers. The inflexibility caused by too few layers may not able to capture the non-linearity in complex systems [13]. Overfitting caused by too many hidden layers can be corrected by regularization. With fewer input and output neurons, fewer hidden layers are required. It has also been shown that even a shallow net ANN is fairly effective in approximating a nonlinear function. After comparing 2, 3, and 4 hidden layers, we conclude that 3 hidden layers is the most effective and efficient.

Choice of Activation Function

The choice of activation functions is crucial in the effectiveness of the function approximation of an ANN. There is a variety of activation functions available. Figure 3-2 shows some common activation functions. Since our task is a multi-outputs regression problem, the output layer uses linear function as the activation function. For the hidden layers, a grid search over 5 different activation function are conducted. For evaluation, we use negative mean squared score. A higher score is an indication of better results.



Sigmoid
$\sigma(x) = \frac{1}{1+e^{-x}}$
tanh
$\tanh(x)$
ReLU
$\max(0, x)$
Leaky ReLU
$\max(0.1x, x)$
Maxout
$\max(w_1^T x + b_1, w_2^T x + b_2)$
ELU
$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$

Figure 3-2. Some Common Activation Functions for ANN [15]

	relu	softmax	tanh	sigmoid	linear	choice
layer 1	-0.07954	-0.04626	-0.07765	-0.08316	-0.07856	softmax
SD	0.0319	0.0034	0.0209	0.0331	0.0238	
layer 2	-0.08147	-0.05504	-0.05844	-0.04911	-0.07348	sigmoid
SD	0.0368	0.0196	0.0207	0.0193	0.0310	
layer 3	-0.06075	-0.05314	-0.06693	-0.05685	-0.06869	softmax
SD	0.0207	0.0139	0.0282	0.0212	0.0468	

Table 3-I. Negative Mean Squared Score and Standard Deviation (SD) of Gridsearch for Activation Function

Table 3-I shows the average negative mean squared score and its standard deviation in the gridsearch with 10-fold cross validation for different activation function for each layer. A higher average negative mean squared score and a lower standard deviation indicate a better result. Softmax function for first hidden layer, sigmoid function for second hidden layer and softmax function for the third hidden layer achieve the highest score with low standard deviation.

Choice of Optimizer

There are a wide variety of optimizers available in Keras library to optimize the ANN. to name a few: Stochastic Gradient Descent (SGD), adaptive moment estimation (Adam), Nesterov-accelerated Adaptive Moment Estimation (Nadam), and so on. A grid search is conducted over 5 different optimizers. Negative mean squared score is used for evaluation. A higher score is an indication of better results.

Optimizer	SGD	RMSprop	Adagrad	Adam	Nadam	choice
Score	-0.12343	-0.08635	-0.12323	-0.05283	-0.05332	Adam
SD	0.0421	0.0485	0.0420	0.0261	0.0233	

Table 3-II. Negative Mean Squared Score and Standard Deviation (SD) of Gridsearch for Optimizer

Table 3-II shows the average negative mean squared score and its standard deviation in the gridsearch with 10-fold cross validation for different ANN optimizer. A higher average negative mean squared score and a lower standard deviation indicate a better result. Adam optimizer achieves the highest score with low standard deviation.

Regularization

Due to the high non-linearity nature of this IK problem, hidden layers have up to 300 neurons. To prevent over-fitting, we will use dropout regularization. Dropout regularization is proposed in [16] in 2014. It addresses the over-fitting issue by temporally ignoring randomly selected neurons and removing their contribution downstream.

Since the first hidden layer has the most neurons, 300. Dropout regularization is added to it. A gridsearch is conducted for dropout rate from 0 to 0.4. Negative mean squared score is used for evaluation. A higher score is an indication of better results.

Table 3-III shows the average negative mean squared score and its standard deviation in the gridsearch with 10-fold cross validation for different dropout rate

dropout rate	0.0	0.1	0.2	0.3	0.4	choice
score	-0.0689	-0.0516	-0.0601	-0.0545	-0.0587	0.1 dropout rate
SD	0.0419	0.0219	0.0286	0.0267	0.0214	

Table 3-III. Negative Mean Squared Score and Standard Deviation (SD) of Gridsearch for Dropout Rate

for the first hidden layer. A higher average negative mean squared score and a lower standard deviation indicate a better result. A dropout rate of 0.1 achieves the highest score with low standard deviation.

Running Iterations

In theory, the more epochs or iterations in the model, the better for convergence of the model. However, after a certain number of epochs, the increment in convergence becomes marginal, and time cost becomes high. To find a balance between good convergence and cost in time, different maximum epochs number are compared in Figure 3-3. From the comparison, the decrease in MSE is marginal after 750 epochs; therefore, 750 will be the training maximum epochs for the ANN.

Final Structure for ANN Implemented

After the grid search and comparing different configurations, we decided to use a three-hidden-layer structure. The first hidden layer consists of 300 neurons and uses softmax function for activation. The second hidden layer consists of 150 neurons with sigmoid function for activation. The third hidden layer consists of 70 neurons and uses softmax function for activation. A dropout rate of 0.1 is applied between first and second hidden layer. The optimizer is selected as ‘Adam’, and the maximum number of epochs is 750.

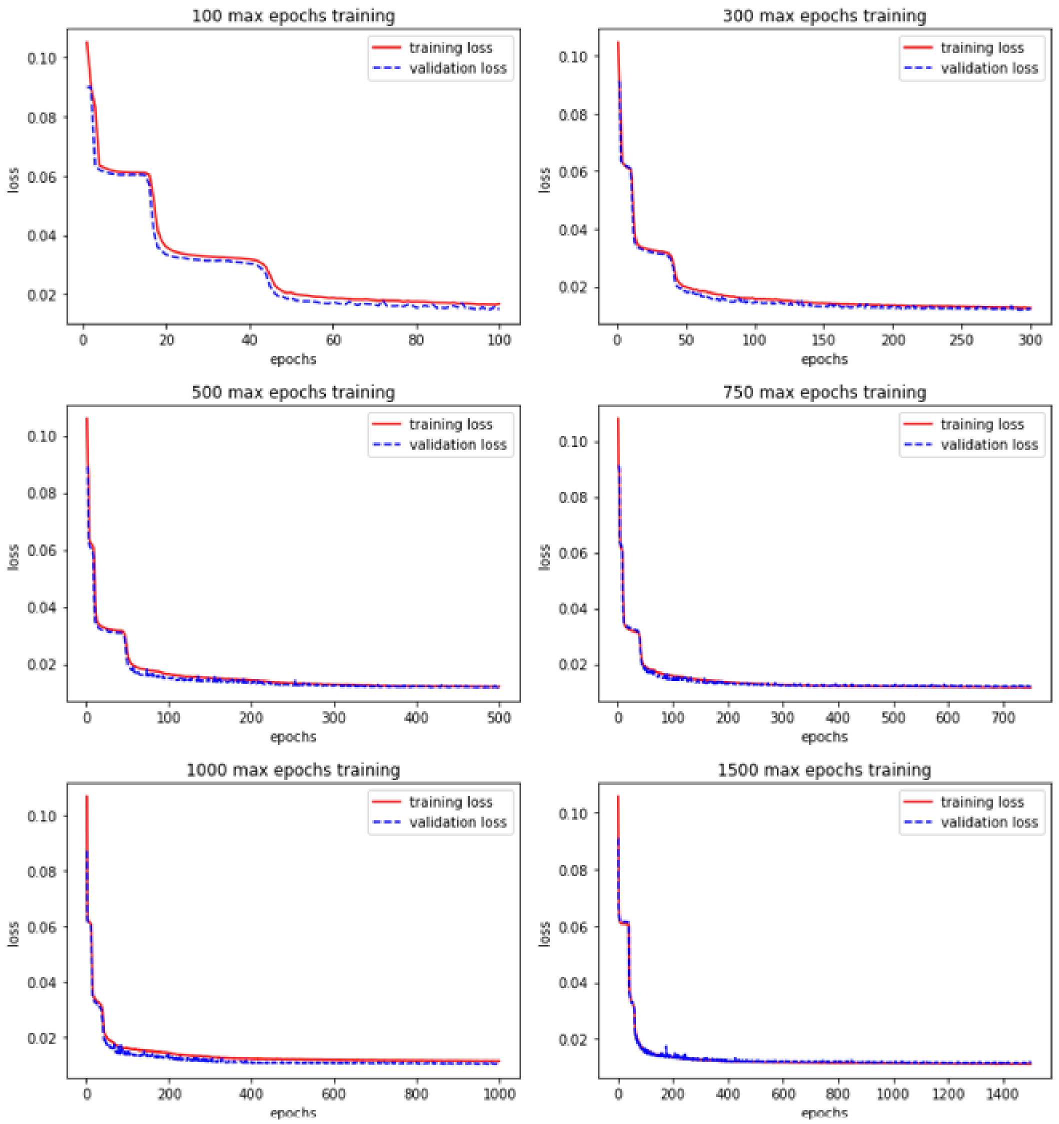


Figure 3-3. Comparing Different Max Epochs Number

Chapter 4

Result and Evaluation

Evaluation Methods

Two different methods are used in this thesis to evaluate the performance of the ANN on the task of Inverse Kinematics. The first method is to evaluate the MSE of the training and testing data. The second method is to generate a set a synthetic data where we first define a desired trajectory curve for the end effector in the workspace, then use this desired curve as the input of trained ANN to generate a set of solutions. Finally we map this set of solutions back to workspace with forward kinematics, to compare with the desired trajectory curve.

Training and Testing MSE

As we can see from Figure 4-1, the training loss MSE quickly converged to a value below 0.020, and remains steady after 300 epochs in the case where there is no noise in training. Importantly, the average test MSE loss of last 50 epochs is 0.01133, and the standard deviation is 0.00012.

For the case with noise present in the training data, the training loss still successfully converges to below, while the test loss is in a larger range from 0.014 to 0.010. The average test MSE loss of last 50 epochs is 0.01181, and the standard deviation is 0.00048. The noise in training data increases the average test MSE in last 50 epochs by 4.2%, but the standard deviation increases from 0.00012 to 0.00048. How much does this impact the result in practice is studied in the next evaluation method, in which the ANN output joint angles are mapped back to the end effector position and orientation in the workspace.

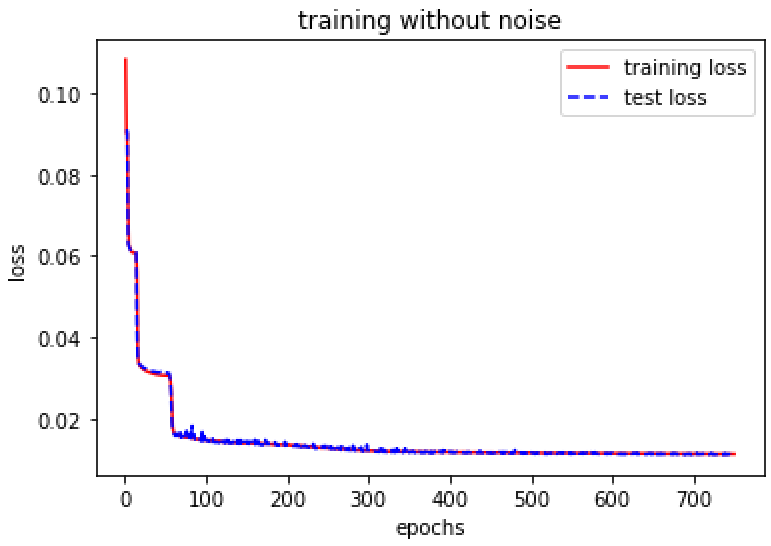


Figure 4-1. MSE Loss vs. Epoch Number, Data with No Noise

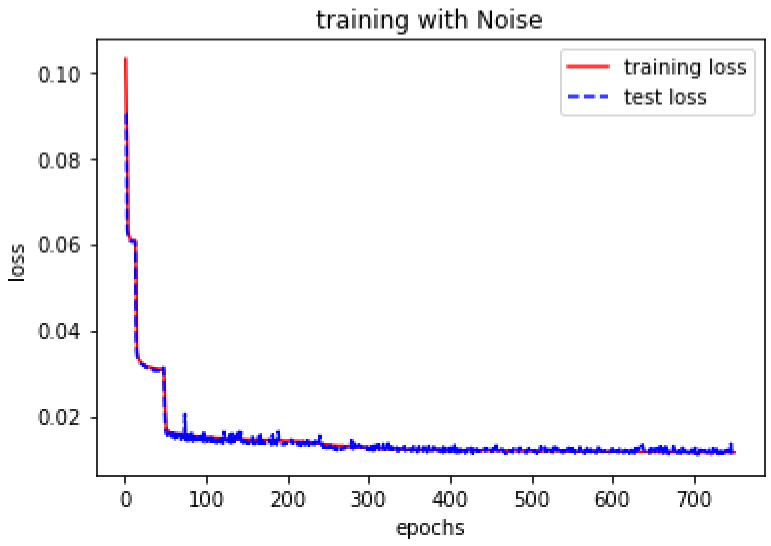


Figure 4-2. MSE Loss vs. Epoch Number, Data with Noise

Curve Matching for the End Effector

Training without Noise

In this evaluation, a set of data is generated by linearly sampling in the configuration space for three joint angles θ_1, θ_2, θ_3 as shown in Figure 4-3.

The sampled joint angles θ_1, θ_2, θ_3 are fed into the closed-form FK to generate a

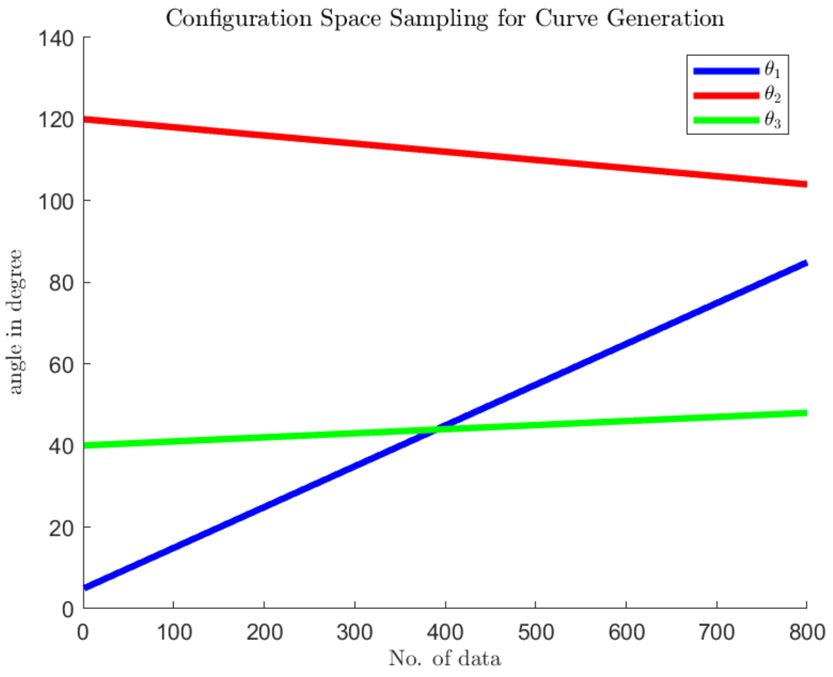


Figure 4-3. θ_1, θ_2, θ_3 Sampling for Generated Curve Matching

curve in the workspace, denoted as the target curve, and a set of position x, y and facing angle α as inputs for the trained ANN. The trained ANN produces a set of predicted joint angles $\hat{\theta_1}$, $\hat{\theta_2}$, $\hat{\theta_3}$. Then, a curve generated by the set of predicted angles is compared with the target curve.

Figure 4-4 shows the comparison between the target curve and the ANN generated curve, and Table 4-I shows the mean squared error (MSE) and mean absolute error (MAE) of the ANN (trained without noise) generated workspace result compared to target workspace points. The joint angles generated by ANN is shown in Figure 4-5.

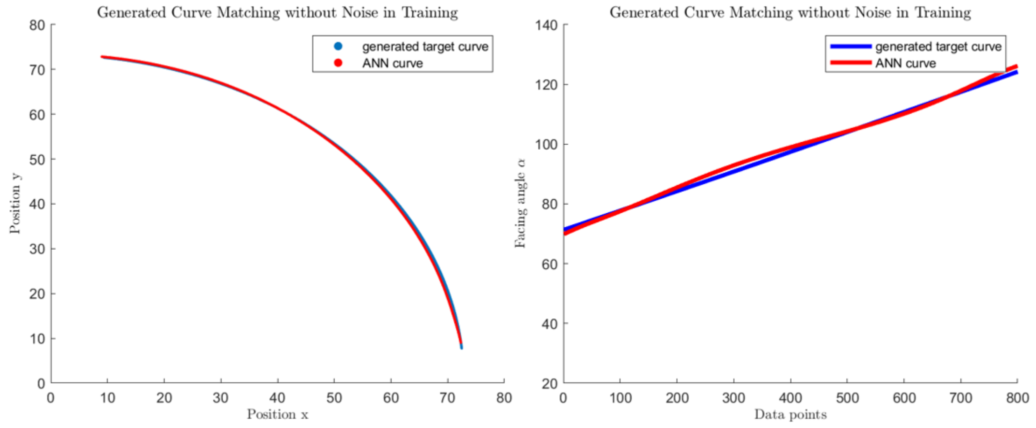


Figure 4-4. Curve Matching with Training Data without Noise)

	position x [in]	position y [in]	facing angle α [°]
MSE	0.8408	1.2939	1.4111
MAE	0.8408	0.9263	0.9819
target value range	10 - 75	5 - 75	60 - 120

Table 4-I. MSE and MAE of ANN Approximated Curve (No Noise in Training)

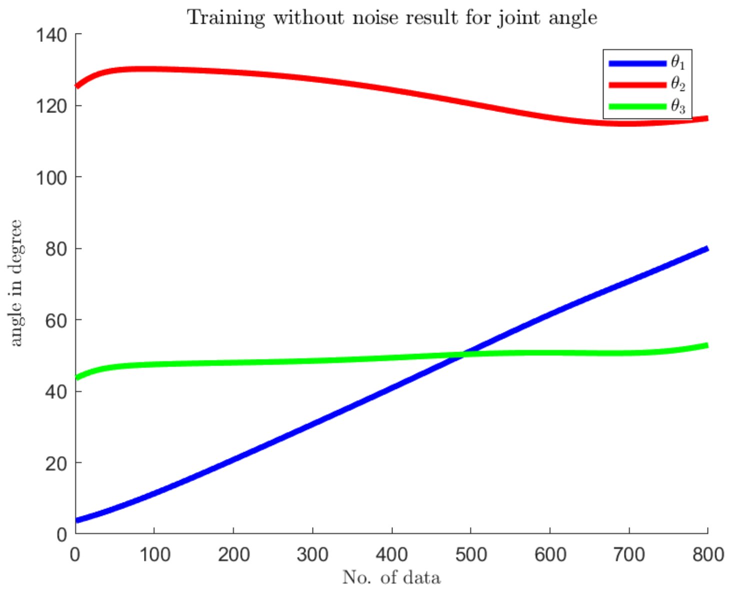


Figure 4-5. ANN Generated Joint Angels (No Noise in Training)

Training with Noise

Figure 4-6 shows the performance of the ANN model when there is noise present in the training data. Table 4-II shows the mean squared error (MSE) and mean absolute error (MAE) of the ANN (trained with noisy data) generated workspace result compared to target workspace points. The average MAE of facing angle increases from 0.819 to 3.14, but position x and y error are unimpaired.

Even though MAE of facing angle is increased by 2 folds from training with no noise to training with noise, the MAE is still only 2.5% to 5% to the facing angle value.

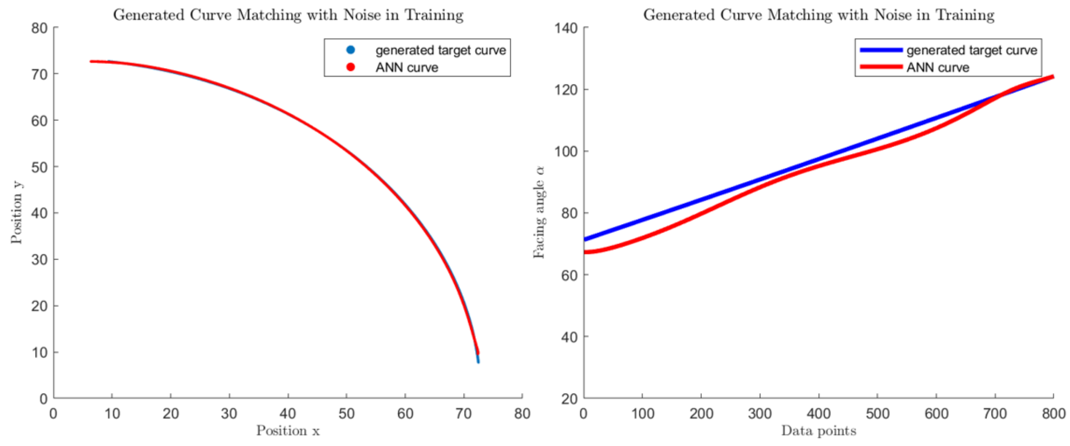


Figure 4-6. Curve Matching with Noisy Training Data

	position x [in]	position y [in]	facing angle α [°]
MSE	0.8453	0.6346	12.6941
MAE	0.5264	0.5435	3.1400
target value range	10 - 75	5 - 75	60 - 120

Table 4-II. MSE and MAE of ANN Approximated Curve (with Noise in Training)

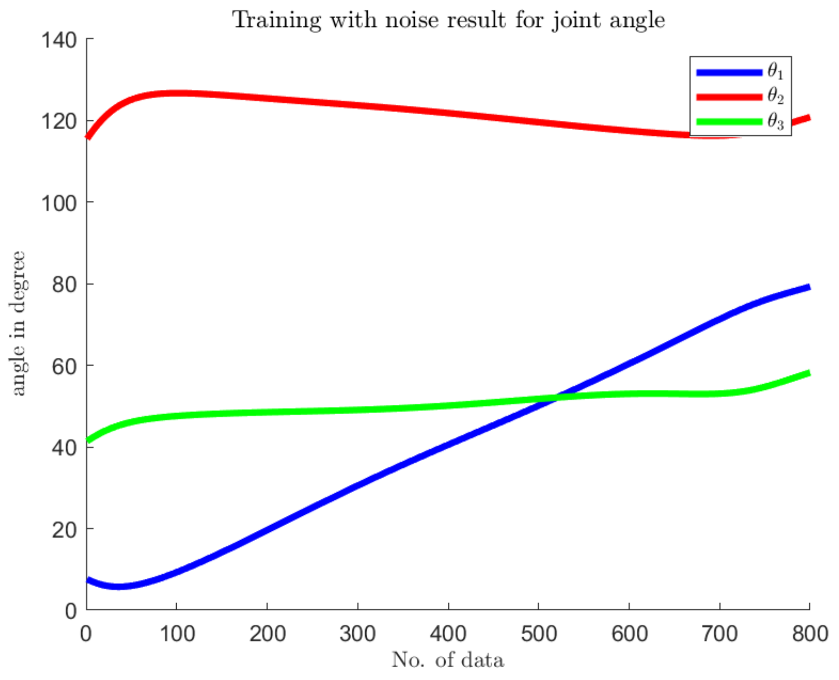


Figure 4-7. ANN Generated Joint Angels (with Noise in Training)

Conclusion

As shown in both MSE results and curve matching results, the ANN is effective in approximating the IK of the robot of interest. For desired curve matching, the average positional errors, in both training with or without noise, are under 1 for a value range from 5 to 75 inches, and angle error is within 1.5° in training with no noise, and under 3.5 in training with noise. With increased training data size, network complexity and training iterations, the accuracy will further improve. This ANN approach is suitable for real-world practical applications.

Chapter 5

Conclusion

Summary

In this thesis, we analyzed the kinematics of a serial and parallel hybrid robot manipulator. Closed-form solution for forward kinematics (FK) is derived analytically, and was validated with a computer aided design (CAD) program. With closed-form FK, two sets of 21600 data points in the workspace were generated, one set with noise and the other set without noise. After conducting cross validation and grid search, we concluded that a three-hidden-layer ANN with a mix of softmax and sigmoid activation functions is effective and efficient to approximate the inverse kinematics (IK) of the manipulator. The ANN model is evaluated with testing data. After evaluation, we concluded that the ANN approach is effective in performing the IK tasks. During the evaluation, the ANN approach has also demonstrated robustness with noise present in training data.

Limitation and Future Work

There are two main disadvantages in using ANN to approximate Inverse Kinematics solutions.

The first and biggest disadvantage is lack of comprehensive sets of solution. In more complex cases with high dimension, except singularity or out of workspace, there are more than one solution of joint configuration to a position and orientation of end effector. For example, there are 14 solutions to a 6-degree-of-freedom 6R robot arm [3]. The ANN method proposed in this thesis only finds one solution, and we cannot compare the solution with other possible solutions. However, there is a related work [17] which proposed an adaptive niching genetic algorithm approach that generates multiple solutions in certain robots.

The second potential disadvantage is failure to inform when the desired end-effector position and orientation is outside of the workspace, because ANN will produce an output with any given input, even if the input will result in invalid geometry. However, this issue can be resolved by setting the workspace boundaries beforehand. With known FK, it is relatively simple to find the workspace boundaries, both analytically or numerically.

References

1. Murray, R. M., Li, Z. & Sastry, S. S. *A Mathematical Introduction to Robotic Manipulation* (CRC Press, 1994).
2. Yoshikawa, T. *Foundations of Robotics, Analysis and Control* Fourth Edition, 124 (MIT Press, New York, 1990).
3. Raghavan, M. & Roth, B. Inverse Kinematics of the General 6R Manipulator and Related Linkages. *Journal of Mechanical Design* **115,** 502–508 (Sept. 1993).
4. Primrose, E. On the input-output equation of the general 7R-mechanism. *Mechanism and Machine Theory* **21,** 509–510 (1986).
5. Reboulet, C. & Berthomieu, T. Dynamic models of a six degree of freedom parallel manipulators, 1153–1157 vol.2 (1991).
6. Kim, J. S., Levi, D., Monfaredi, R., Cleary, K. & Iordachita, I. *A new 4-DOF parallel robot for MRI-guided percutaneous interventions: Kinematic analysis* 2017.
7. Kim, J. S. & Chirikjian, G. S. Inverse kinematic solutions of 6-D.O.F. biopolymer segments. *Robotica* **34,** 1734–1753 (2016).
8. Aristidou, A. & Lasenby, J. Inverse Kinematics: a review of existing techniques and introduction of a new fast iterative solver (Sept. 2009).
9. Hinton, G. E. How Neural Networks Learn from Experience. *Scientific American* **267,** 144–151 (1992).
10. Ferrari, S. & Stengel, R. F. Smooth Function Approximation Using Neural Networks. *IEEE Transactions on Neural Networks* **16,** 24–38 (2005).
11. Tejomurtula, S. & Kak, S. Inverse kinematics in robotics using neural networks. *Information Sciences* **116,** 147–164 (1999).
12. Chong, E. K. P. & Zak, S. H. *An Introduction to Optimization* Fourth Edition, 124 (John Wiley & Sons, Inc., New York, 1996).
13. Hastie, T., Tibshirani, R. & Friedman, J. *The Elements of Statistical Learning* (Springer, Stanford, California, 2008).
14. Bishop, C. M. *Neural Networks for Pattern Recognition* First edition (Clarendon Press, 1996).
15. Jayawardana, R. & Bandaranayake, T. Analysis of Optimizing Neural Networks and Artificial Intelligent Models for Guidance, Control, and Navigation Systems. *International Research Journal of Modernization in Engineering Technology and Science* **03,** 420 (03 Mar. 2021).

16. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research* **15,** 1929–1958 (2014).

17. Tabandeh, S., Melek, W. W. & Clark, C. M. An adaptive niching genetic algorithm approach for generating multiple solutions of serial manipulator inverse kinematics with applications to modular robots. *Robotica* **28,** 493–507 (July 2010).