Hindawi Publishing Corporation
Journal of Robotics
Volume 2010, Article ID 870923, 9 pages
doi:10.1155/2010/870923

Research Article

Solution for Ill-Posed Inverse Kinematics of Robot Arm by Network Inversion

Takehiko Ogawa and Hajime Kanada

Department of Electronics and Computer Systems, Takushoku University, 815-1 Tatemachi, Hachioji, Tokyo 193-0985, Japan

Correspondence should be addressed to Takehiko Ogawa, togawa@es.takushoku-u.ac.jp

Received 1 December 2009; Revised 9 April 2010; Accepted 7 July 2010

Academic Editor: Noriyasu Homma

In the context of controlling a robot arm with multiple joints, the method of estimating the joint angles from the given end-effector coordinates is called inverse kinematics, which is a type of inverse problems. Network inversion has been proposed as a method for solving inverse problems by using a multilayer neural network. In this paper, network inversion is introduced as a method to solve the inverse kinematics problem of a robot arm with multiple joints, where the joint angles are estimated from the given end-effector coordinates. In general, inverse problems are affected by ill-posedness, which implies that the existence, uniqueness, and stability of their solutions are not guaranteed. In this paper, we show the effectiveness of applying network inversion with regularization, by which ill-posedness can be reduced, to the ill-posed inverse kinematics of an actual robot arm with multiple joints.

1. Introduction

In the context of controlling a robot arm with multiple joints, the problem of estimating the joint angles from the given end-effector coordinates is called the inverse kinematics problem, which is a type of inverse problems [1]. Inverse problems that estimate the cause from the given results are studied in various engineering fields [2]. There are a number of methods for solving inverse kinematics problems: analytical method, iterative calculation by using an algorithm, and so forth [1]. In addition, neural-network-based inverse modeling has been proposed [3], and we can use it as a solution of inverse kinematics [4, 5]. The network inversion method has been proposed for solving inverse problems by using a multilayer neural network [6]. In this method, inverse problems are solved by using a trained multilayer neural network inversely to estimate the corresponding input from the given output [7, 8]. The advantages of this method are easiness of the direct modeling by learning and adaptive estimation of the inverse solution. It has been applied to actual problems [9, 10]. In addition, it was introduced as a method to solve the inverse kinematics problem of estimating the multiple joint angles of a robot arm from the given end-effector coordinates [11–13].

In general, inverse problems are affected by ill-posedness, which implies that the existence, uniqueness, and stability of their solutions are not guaranteed [2]. Ill-posedness also affects the solution when a problem is solved using network inversion. The regularization method to decrease ill-posedness by limiting the solution space of the inverse problem has been proposed for the ill-posed inverse problems [14, 15]. It has also been examined for network inversion [16, 17]. The inverse-modeling approach with a multilayer neural network and the approach that uses the network inversion method have been examined as neural-network-based methods. The problem of ill-posedness is an important aspect of inverse problems, and it has often been studied in the field of mathematical engineering [2, 15]. However, only a few studies have examined the possibility of solving the problem of ill-posedness by using a neural network [18]. In particular, a few systematic investigations of ill-posedness have been carried out by applying a neural network to the actual problem, but these investigations considered only the uniqueness of the solution [19].

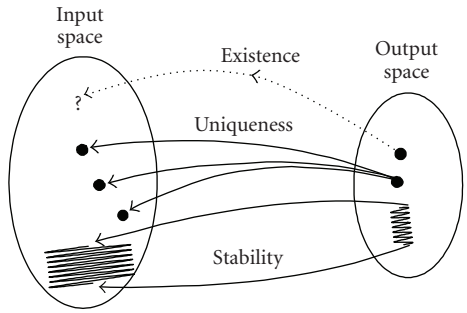

Figure 1: Concept of inverse problem and ill-posedness.

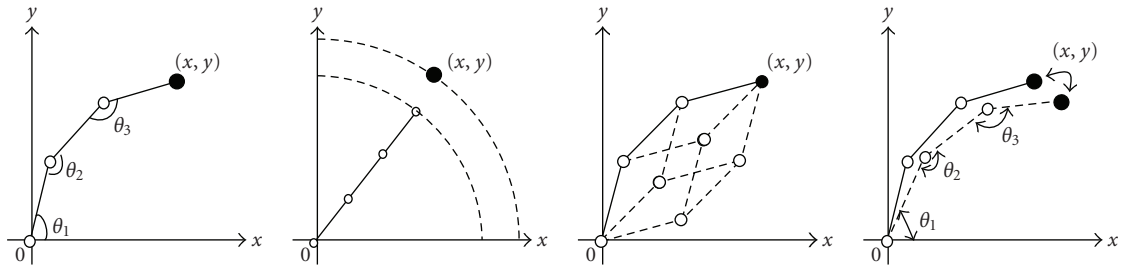

Figure 2: Inverse kinematics problem of 3-DOF robot arm in 2D plane. (a) Three joints and an end-effector coordinate. Ill-posed solutions in terms of (b) existence, (c) uniqueness, and (d) stability.

In this study, we aim at examining the applicability of the network inversion method to the inverse kinematics of a robot arm with multiple joints. That is, we aim to demonstrate the applicability of this method to the ill-posed inverse kinematics in terms of the existence, uniqueness, or stability of the solutions provided by the method. In addition, we aim to examine the applicability of the network inversion method to all problems that involve ill-posedness. We consider a three-degree-of-freedom (DOF) robot arm that moves in two-dimensional space and set the ill-posed problem as one in which the existence, uniqueness, or stability of the solution is not satisfied. We then examine the problem and solution by using the network inversion method so as to demonstrate the effectiveness of this method in addressing the inverse kinematics of a robot arm with multiple joints. We consider that the results suggest the effectiveness of the method when applied to inverse not only kinematics but also a general ill-posed inverse problem.

2. Inverse Estimation of Joint Angles of Robot Arm

An inverse problem determines the inner mechanism or cause from observed phenomena. In the case of a direct problem, the result is derived from a given cause by using a fixed mathematical model, whereas in the case of an inverse problem, the cause is estimated from the fixed model and given results. In the case of a direct problem, the existence, uniqueness, and stability of the solution are guaranteed. Problems in which these three conditions are satisfied are called as well-posed problems. In the case of an inverse problem, it may be difficult to obtain a solution because the well-posedness of the problem is not guaranteed [2]. An example of ill-posedness is shown in Figure 1.

Direct kinematics implies a problem of calculating the end-effector coordinates from the joint angles of robot arms. On the other hand, inverse kinematics inversely estimates the joint angles from the provided end-effector coordinates of robot arms. A number of analytical methods, algorithms of iterative calculation, and so forth exist for solving inverse kinematics problems. Analytical methods are effective for a robot arm of a low degree-of-freedom. They include the method of solving kinematics equations as nonlinear simultaneous algebraic equations, the method of solving an equation as a problem in planar geometry, and so on. Because an analytical method is not a general method, the iterative approximation algorithm that is based on the Newton method is generally used [1]. In addition, there is a method of solving inverse kinematics problems by using the adaptively estimated inverse model of kinematics [5]. The use of a neural network is proposed as a method of this inverse modeling [11]. Motion control of a robot arm is often achieved by the point-to-point or continuous-path methods, and it is required to solve the inverse kinematics problem.

In this paper, we consider a 3-DOF robot arm that moves in two-dimensional space, which is shown in Figure 2(a). We examine the problem of estimating the joint angles for the specified end-effector coordinate of the 3-DOF robot arm. In an inverse kinematic problem, the problem of ill-posedness in terms of the existence, uniqueness, and stability of the solution also arises. For instance, the combination of joint

angles for the specified end-effector coordinate may not exist, as shown in Figure 2(b). This is an ill-posed problem where a solution does not exist. Figure 2(c) shows an example where there are a number of combinations for the joint angles of an end-effector coordinate; here, the problem is ill-posed in terms of its uniqueness. In addition, a minor measurement error for the end-effector coordinate may cause a large estimation error of the joint angles according to scaling of the joint angles and end-effector coordinate, as shown in Figure 2(d); the problem is ill-posed because the solution is unstable [2].

In this paper, we introduce the network inversion method in order to solve the inverse estimation problem of joint angles for a 3-DOF robot arm. We examine the solutions to the above-mentioned ill-posed problems.

3. Network Inversion

An ordinary multilayer neural network is used to solve a direct problem. In a usual multilayer network in which the training is complete, the input-output relation is given by $y = f(w, x)$, where x, y, and f represent the input vector, output vector, and function defined by the interlayer weights w of the network, respectively. If the input vector x is provided, the network calculates the output vector y.

Linden and Kindermann proposed the network inversion method [6]. In this method, the observed output data y can be applied with f fixed after finding the forward relation f by training. The input x can then be updated according to the calculated input correction signal based on the duality of the weights and input. The input is estimated from the output by iterative updating of the input based on the output error, as shown in Figure 3. In this manner, the inverse problem for estimating the input x from the output y is solved with the multilayer neural network by inversely using the forward relation.

We use a network in two phases so as to solve the inverse problem by network inversion: forward training and inverse estimation. The procedure is shown in Figure 4. In the training phase, we provide the training input x and training output y to calculate the output error E. The weight w is then updated by

$$w(n+1) = w(n) - \varepsilon_t \frac{\partial E}{\partial w}, \tag{1}$$

where ε_t denotes the training gain, because the output error is due to maladjustments of the weights. By repeating this update procedure, the forward relation is obtained. This procedure is based on the usual back propagation method.

In the inverse estimation phase, we fix the relation obtained in the training, provide the random input x and test output y, and calculate the output error E. The input x is then updated by

$$x(n+1) = x(n) - \varepsilon_e \frac{\partial E}{\partial x}, \tag{2}$$

where ε_e denotes the input update gain, because the output error is due to the error in the input. By repeating this update procedure, the input is estimated from the output.

4. Ill-Posedness and Regularization

In general, inverse problems are ill-posed in that the existence, uniqueness, and stability of their solutions are not guaranteed [2]. Regularization, which limits the solution space of an inverse problem, is a method that reduces ill-posedness. In this method, a functional intended to provide some constraints is added to the error functional of the iterative method. For example, when the transform from the element x in space X to the element y in space Y is expressed as $Kx = y$ by mapping K, we search the element x to minimize the functional

$$E = ||y - Kx||^2 + \lambda F(x), \tag{3}$$

where $F(x)$ and λ represent the regularization functional and regularization coefficient, respectively.

We can also use regularization for network inversion [17]. In order to provide the constraint condition, we minimize the regularization functional in accordance with the output error in the inverse estimation phase. The energy function $E(x)$ with the regularization functional $F(x)$ is defined as

$$E(x) = ||\tilde{y} - Kx||^2 + \lambda F(x), \tag{4}$$

where $\tilde{y}$, K, and x indicate the network output, transform, and input, respectively. In (4), the first and second terms represent the output error of the network and the regularization functional, respectively. The parameter λ is the regularization coefficient. The objective of the regularization term $F(x)$ is to express the restraint condition that the input x should satisfy and to be minimized simultaneously with the error term. By using the regularization term, we aim to obtain a feasible solution that minimizes the error and satisfies the restraint condition. Since $F(x)$ is a restraint condition concerning the input, we can use the minimum norm to select a specific solution, the minimum difference to smooth the solution, and so forth. Moreover, we can use the minimum and maximum norms of the vector of the joint angles, the minimum and maximum inclinations of the arm, and so forth by expressing them as the regularization term $F(x)$ for the robot arm considered in this paper. The coefficient λ is usually fixed and determined empirically or by trial and error. However, the balance between the output error minimization and regularization is often uncertain. The regularization coefficient may actually be reduced with a decrease in the output error. We refer to this as dynamic regularization. For example, the regularization coefficient $\lambda(t)$ is reduced for each input correction as

$$\lambda(t) = \lambda(0) \exp(-mt), \tag{5}$$

where $\lambda(0)$, t, and m denote the initial value of λ, current epoch number, and decay coefficient, respectively. The parameter $\lambda(t)$ decays from $\lambda(0)$ to zero with the epoch number t.

Table 1: Network parameters.

Input neurons		3
Hidden neurons		10
Output neurons		2
Training rate ε_t		0.01
Inverse estimation rate ε_e		0.01
Training error to be attained		<0.001
Estimation error to be attained		<0.0001
Joint angles for training data (°)	θ_1	0, 15,. . ., 90
	θ_2	45, 60,. . ., 315
	θ_3	45, 60,. . ., 315
Arm length l (cm)		30

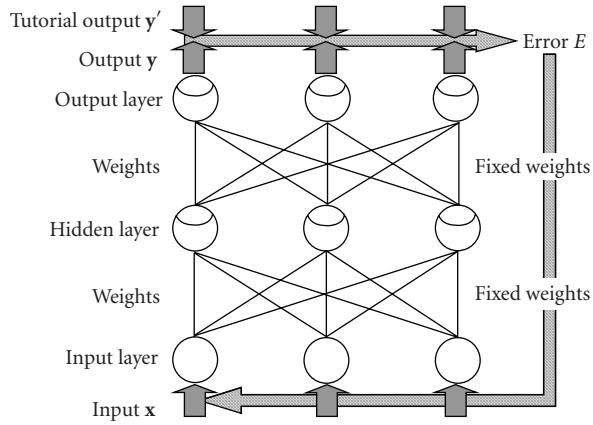

Figure 3: Iterative update of input from provided output by network inversion.

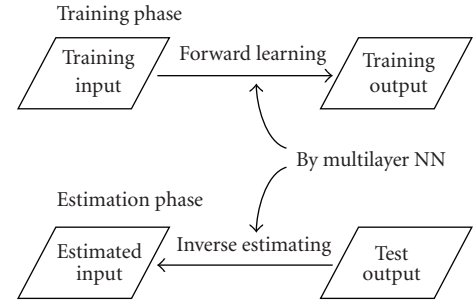

Figure 4: Two-step procedure to solve inverse problem by using network inversion.

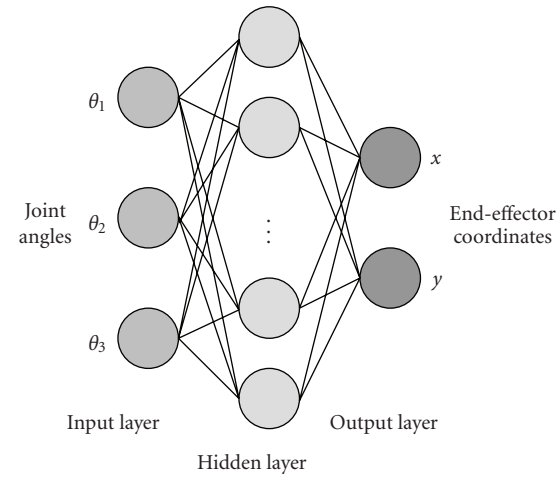

Figure 5: Network architecture.

In this paper, we introduce the regularization method to reduce the ill-posedness of the problems in terms of the uniqueness and stability of their solutions. For the uniqueness problem, we examine the addition of a regularization term for conditioning and attempt to selectively estimate a specific solution. We select a solution that meets the requirement by specifying conditions for the joint angles. This method is shown to be effective when the solution for a specific shape of the arm is required to avoid some obstacle. Moreover, we attempt to stabilize the solutions by decreasing the difference between multiple solutions. We consider a regularization term to minimize the difference in solutions to successive data. This method is shown to be effective for estimating joint angles when the robotic arm is successively operated.

The principle of network inversion is the iterative correction of the input so as to minimize the error for a given output. That is, the output error converges to almost zero when it reaches an appropriate input for the given output. Therefore, the output error remains when the input to a given output does not exist. Thus, we can estimate that a solution may not exist by observing decreases in the output error in response to the corrections made to the input by network inversion. As a result, we consider it possible to deal with ill-posedness in terms of the existence of a solution.

5. Simulation

In order to demonstrate the application of the network inversion method to an inverse kinematics problem and its effect on the ill-posedness, we carry out four simulations considering an ill-posed problem with no solution, with a nonunique solution, and with an unstable solution and a well-posed problem. These four simulations aim to cover all situations concerning ill-posedness when the network inversion method is applied to actual inverse kinematics problems.

We examine the inverse estimation of joint angles from the end-effector coordinates of the 3-DOF robot arm, shown in Figure 2(a), using network inversion. First, we train the network by providing the joint angles and end-effector coordinate corresponding to the joint angles as the tutorial input and output, respectively. The training of the network is conducted by the usual error back-propagation method. The inputs are then iteratively updated due to the error between the network outputs and the end-effector coordinate value. Initially, the inputs are set to random values, and the trained weights are assigned to the values obtained during learning. In this manner, the joint angles are estimated from the provided end-effector coordinates.

The network architecture used in the simulation is shown in Figure 5, and the network parameters are listed in Table 1.

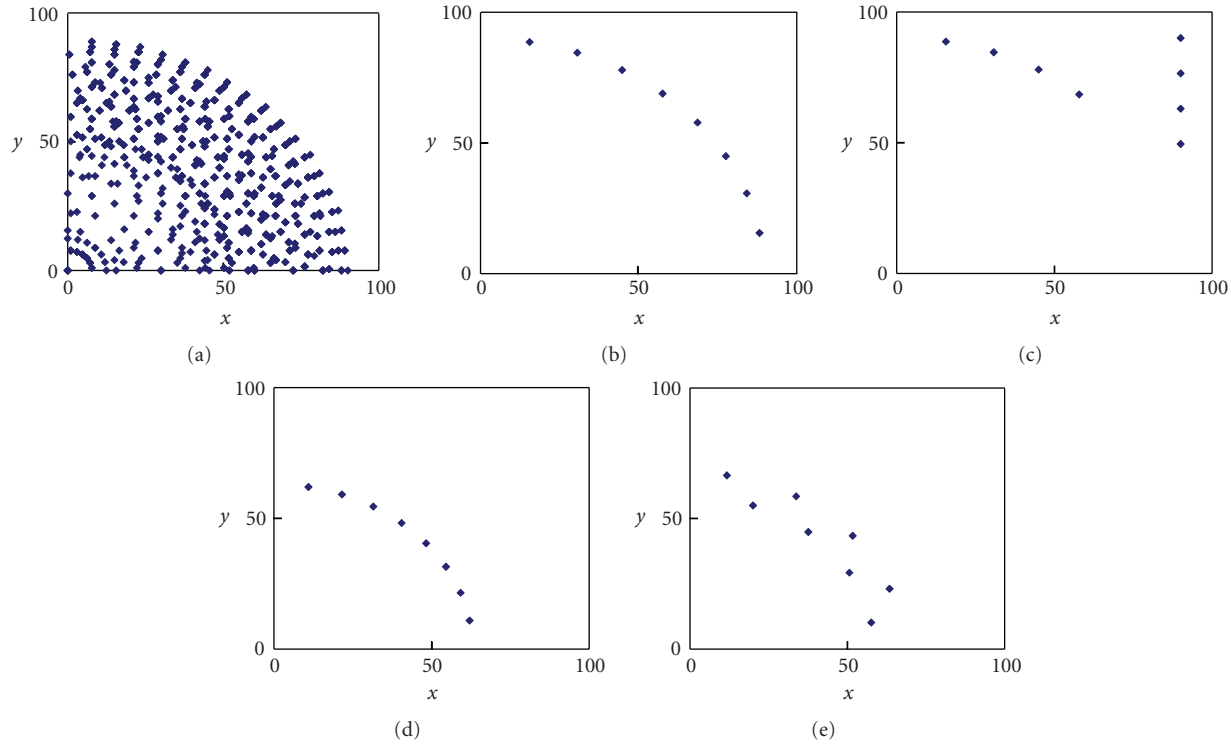

Figure 6: Distribution of end-effector coordinates (a) for training data and for test data of (b) well-posed problem and a problem with (c) nonunique solution, (d) no solution, and (e) unstable solution.

Because there are three joint angles (θ_1, θ_2, and θ_3) and two end-effector coordinates (x and y) for the arm, we use a network with three inputs and two outputs. The hidden neurons are decided to be 10 units by trial and error. The input and output values of the network are normalized to the range [0.0, 1.0] by the maximum and minimum values of the joint angles and coordinates.

First, we prepare the training data by calculating the end-effector coordinates for the three joint angles. We obtain 1136 data points in the tutorial training dataset by varying each of the joint angles θ_1, θ_2, and θ_3 to 7, 19, and 19 different values, respectively, and by selecting only positive coordinates (x, y). The length of each arm is 30 cm. The distribution of the end-effector coordinates in the training data is shown in Figure 6(a). The training is continued until the mean square error decreases to less than 0.001. The training rate ε_t is set to 0.01. The actual training is completed after approximately 8000 epochs.

In the inverse estimation, the joint angles are estimated from the given end-effector coordinates by means of network inversion. In order to confirm the effectiveness of the network inversion to the ill-posed inverse kinematics, we set each of the end-effector coordinates for well-posed, nonexistent, nonunique, and unstable solutions. The end-effector coordinates shown in Figures 6(b), 6(c), 6(d), and 6(e) correspond to well-posed, nonexistent, nonunique, and unstable solution data, respectively. The inverse estimation is continued until the mean square error decreases to less than 0.0001. The inverse estimation rate ε_e is set to 0.001. In order to confirm the effect of the regularization method, the initial input values are set to the constant value of 0.5 in this simulation, although they are generally set to random values for network inversion.

Table 2: Estimated joint angles, calculated end-effecter coordinates, and mean square errors for well-posed problem.

	Joint angles (°) (θ_1, θ_2, θ_3)	End-effector coordinates (x, y)	MSE of end-effector coordinates
1	(15.56, 163.07, 189.15)	(88.61, 11.40)	4.23
2	(38.55, 153.58, 175.06)	(82.56, 28.75)	2.86
3	(42.80, 166.39, 173.72)	(75.84, 46.69)	2.70
4	(44.49, 176.89, 178.63)	(66.89, 60.14)	3.07
5	(45.67, 184.19, 181.96)	(58.85, 67.98)	1.39
6	(47.29, 194.13, 186.55)	(45.96, 76.19)	2.00
7	(64.68, 190.92, 168.88)	(33.22, 83.25)	2.77
8	(73.66, 194.68, 169.71)	(15.52, 88.13)	0.51

5.1. Results for Well-Posed Problem. In order to demonstrate the fundamental ability of inverse estimation by network inversion, we simulate a well-posed problem. We use the abovementioned training data. The eight points of the end-effector coordinates that have unique solutions and are shown in Figure 6(b) are used as data for inverse estimation. Regularization is not used for this simulation. The estimated results for the joint angles are shown in Figure 7 and Table 2. The end-effector coordinates calculated from the estimated

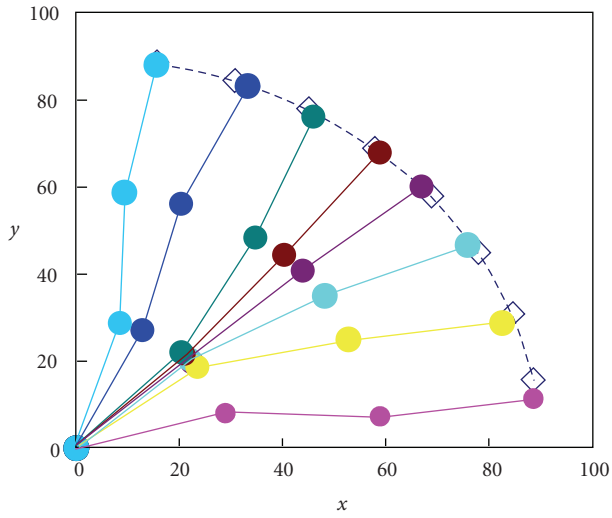

Figure 7: Graph showing the coordinates of robot arm whose joint angles are estimated by network inversion for well-posed problem.

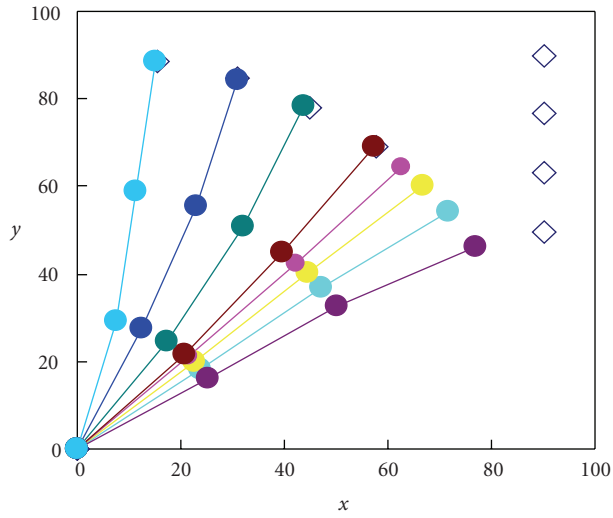

Figure 8: Graph showing the coordinates of robot arm whose joint angles are estimated by network inversion for ill-posed problem with no solution.

joint angles closely corresponded to the given end-effector coordinates shown in Figure 7. Table 2 shows that the error between the estimated and given coordinates is small. According to these results, the joint angles that realized the given end-effector coordinates are accurately estimated. We confirm that the joint angles are correctly estimated from the given end-effector coordinates by network inversion in the case of the well-posed problem.

5.2. Results for Ill-Posed Problem with No Solution. We consider measures to reduce the ill-posedness when no solution exists. When an impossible end-effector coordinate is given, the system is required to estimate the joint angles that realize the end-effector coordinate as much as possible for an approximate or provisional solution. Further, it is necessary to specify that the estimated solution is an approximate one and that a rigorous solution does not exist.

Here, we present the inverse estimation of an approximate solution by means of network inversion when a rigorous solution does not exist. The abovementioned training data are used for training. For inverse estimation, we use data that consist of four possible and four impossible end-effector coordinates, as shown in Figure 6(c). We do not use regularization in this simulation. We consider the transition and the converged values of the output error in the inverse estimation as criteria specifying the nonexistence of a solution.

The estimated joint angles are shown in Figure 8 and Table 3. According to the results, the joint angles are approximately estimated for the impossible end-effector coordinates, while they are correctly estimated for the possible end-effector coordinates. In other words, Figure 8 shows the results of the estimated end-effector coordinates in the possible range for impossible coordinates. Figure 9 shows the transition of the output error in the inverse estimation. We found that the transition and converged value of the output error were obviously different for the impossible

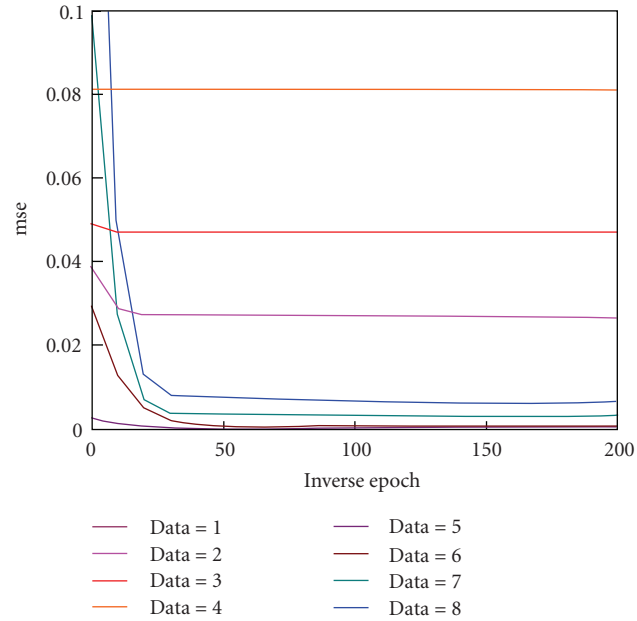

Figure 9: Error curves of inverse estimation for each estimation data.

data points no.1 to no.4 and the possible data points no.5 to no.8. We confirmed that the approximated joint angles for a given end-effector coordinate can be estimated by network inversion when a solution does not exist. Moreover, we confirmed that the nonexistence of a solution can be determined according to the output error of the inverse estimation.

5.3. Results for Ill-Posed Problem with Nonunique Solution. We consider the effect of using regularization as a selection method when a solution is not unique. Here, we examine the maximization and minimization of the joint angles.

Table 3: Estimated joint angles, calculated end-effector coordinates, and errors for ill-posed problem with no solution.

	Joint angles (°) $(\theta_1, \theta_2, \theta_3)$	End-effector coordinates (x, y)	MSE of end-effector coordinates
1	(32.20, 181.72, 172.81)	(77.07, 46.23)	13.34
2	(37.13, 181.99, 175.97)	(71.74, 54.29)	20.23
3	(41.25, 181.77, 178.91)	(66.81, 60.29)	28.29
4	(44.54, 181.36, 181.13)	(62.71, 64.53)	37.33
5	(45.88, 185.27, 182.78)	(57.37, 69.15)	0.52
6	(57.85, 185.84, 186.30)	(43.69, 78.30)	1.36
7	(65.69, 183.78, 185.06)	(30.87, 84.35)	0.24
8	(75.49, 187.07, 180.73)	(14.91, 88.59)	0.72

For example, for the arm shape, we assumed the restraint condition as a case where an obstacle exists. The training dataset is the same, as described above. For inverse estimation, we use data consisting of eight end-effector coordinates, as shown in Figure 6(d).

We examine two types of regularization terms, where the regularization term $F(x)$ of (4) is set as

$$F_1(x) = x_1^2 + (x_2 - 0.5)^2,$$
$$F_2(x) = (1 - x_1)^2 + (x_2 - 0.5)^2, \tag{6}$$

where $\mathbf{x}$ represents a normalized joint angle. $F_1(x)$ forces the angles θ_1 and θ_2 to 90° and 180°, respectively, while $F_2(x)$ forces the angles θ_1 and θ_2 to 0° and 180°, respectively. Moreover, we use a dynamic regularization method that reduces the regularization coefficient according to the updated input

$$\lambda(t) = \lambda_0\left(1 - \frac{t}{T}\right), \tag{7}$$

where the number of input updates, maximum inverse estimation epoch, and initial regularization coefficient are expressed as t, $T = 10000$, and $\lambda_0 = 0.001$, respectively. The inverse estimation is carried out for two regularization methods by using $F_1(x)$ and $F_2(x)$ of (6).

The estimated joint angles are shown in Figure 10 and Table 4. According to the results shown in Figure 10(a), we found that the estimated joint angle θ_1 became maximum for all end-effector coordinates when the regularization term $F_1(x)$ was added. Because of the ratio between the regularization coefficient and the inverse estimation rate, we found that the regularization term only had a slight effect on the estimated joint angle θ_2. Similarly, we found that the estimated joint angle θ_1 was minimized to all end-effector coordinates when the regularization term $F_2(x)$ was added, as shown in Figure 10(b).

From the above-mentioned results, we confirmed that the joint angles for the shape of the robot arm could be estimated on the basis of specific conditions from the given end-effector coordinates by using network inversion with regularization. In other words, an arbitrary shape can be estimated using network inversion with regularization by providing conditions for the joint angles when the joint angles of a 3-DOF robot arm are inversely estimated.

Table 4: Estimated joint angles, calculated end-effector coordinates, and errors with (a) regularization $F_1(x)$ and (b) regularization $F_2(x)$, for ill-posed problem of nonuniqueness.

(a)

	Joint angles (°) $(\theta_1, \theta_2, \theta_3)$	End-effector coordinates (x, y)	MSE of end-effector coordinates
1	(0.00, 145.90, 284.08)	(65.11, 11.37)	3.10
2	(0.00, 170.03, 285.01)	(56.91, 24.69)	3.89
3	(0.00, 182.40, 282.07)	(52.48, 30.30)	2.40
4	(0.00, 199.61, 284.23)	(41.55, 34.99)	8.68
5	(0.00, 217.91, 264.82)	(37.45, 43.67)	5.51
6	(0.00, 234.76, 245.68)	(32.11, 50.37)	4.23
7	(0.00, 252.69, 225.30)	(24.85, 55.13)	5.24
8	(0.00, 281.08, 189.01)	(13.93, 57.62)	5.34

(b)

	Joint angles (°) $(\theta_1, \theta_2, \theta_3)$	End-effector coordinates (x, y)	MSE of end-effector coordinates
1	(90.00, 85.51, 152.49)	(55.35, 11.76)	6.74
2	(90.00, 106.39, 132.56)	(54.48, 23.00)	4.94
3	(90.00, 123.27, 117.75)	(51.33, 31.93)	3.26
4	(90.00, 139.97, 103.46)	(46.13, 39.56	2.33
5	(90.00, 158.16, 89.18)	(38.85, 46.29)	2.57
6	(90.00, 177.82, 87.58)	(31.04, 57.57)	3.04
7	(90.00, 184.35, 90.19)	(27.63, 62.29)	6.82
8	(90.00, 121.83, 290.82)	(1.64, 64.02)	9.51

5.4. Results for Ill-Posed Problem of Instability. We consider the effect of using regularization as a stabilization method when a solution is instable. Here, we attempt to stabilize the solution by assuming that the data for estimation is successively provided and impose the restraint condition between data. This method effectively estimates the joint angles for a given orbit of the end-effector coordinates by the point-to-point method. The above-mentioned training data are used in this simulation. We use the data shown in Figure 6(e) for the inverse estimation; the data are created by adding a disturbance to the amplitude of 10% for every eight points shown in Figure 6(d).

The regularization term $F_3(x)$ of (4) isset as

$$F_3(x) = \|x(n) - x(n-1)\|^2. \tag{8}$$

This regularization term minimizes the difference between the estimation data. We also use dynamic regularization as expressed in (6) and use the same parameters as those mentioned in the previous subsection.

The estimated results of the joint angles with and without the regularization term are shown in Figures 11(a) and 11(b), respectively. The estimated values are shown in Table 5. In Table 5, the mean square error of the end-effector coordinates represents the difference between the

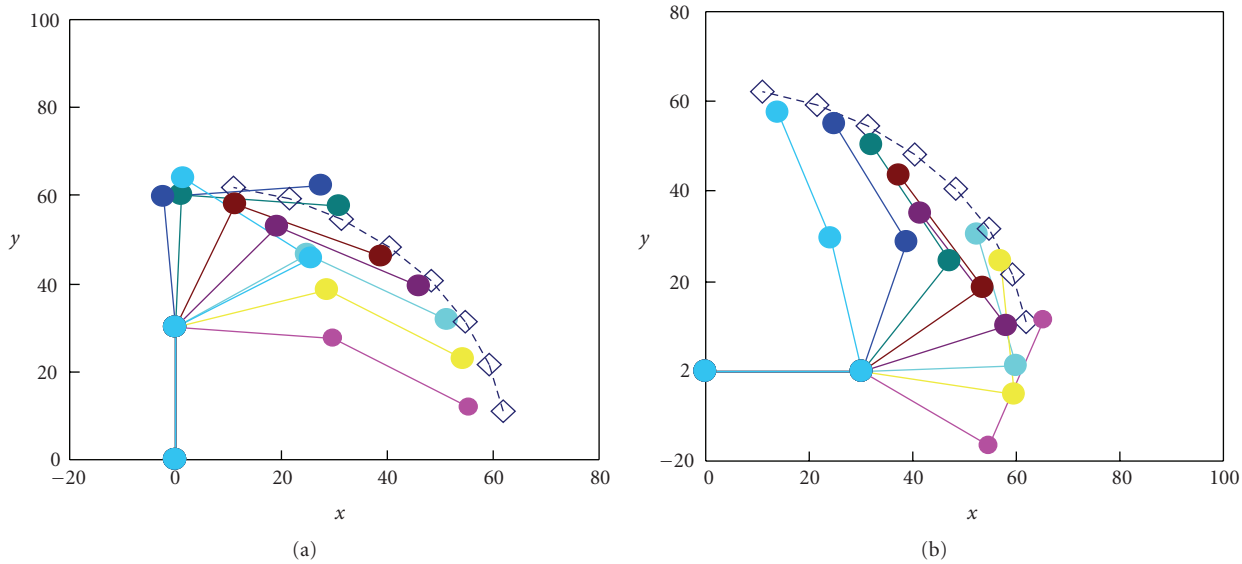

FIGURE 10: Graphs showing the coordinates of robot arm whose joint angles are estimated by network inversion with (a) regularization $F_1(\mathbf{x})$ and (b) regularization $F_1(\mathbf{x})$, for ill-posed problem with nonunique solution.

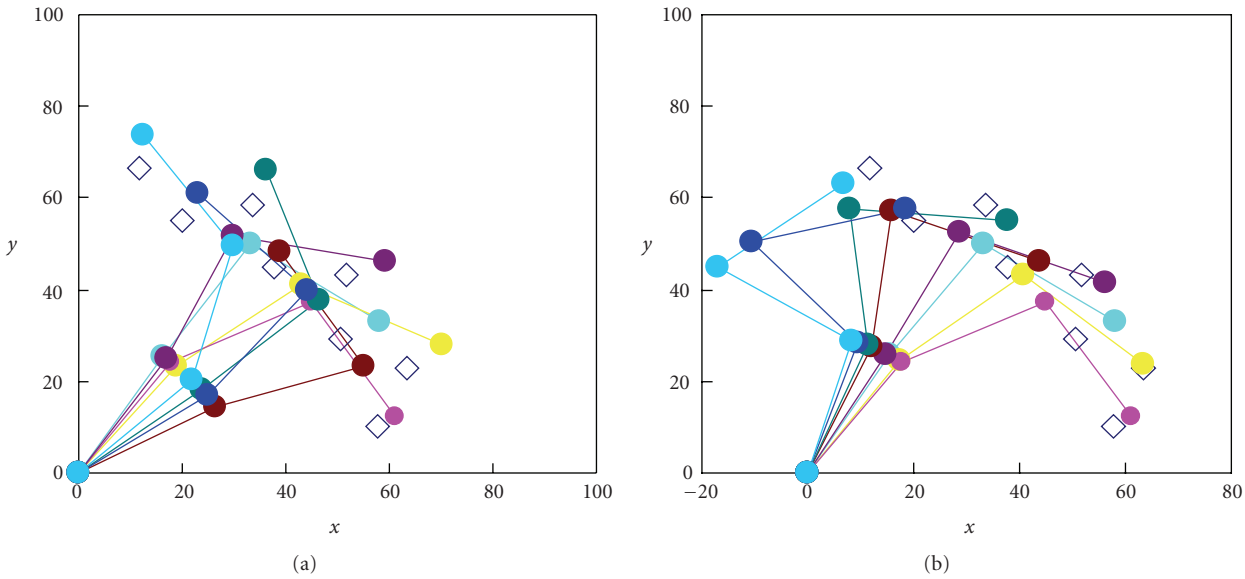

FIGURE 11: Graphs showing the coordinates of robot arm whose joint angles are estimated by network inversion (a) with regularization $F_3(x)$ and (b) without regularization for ill-posed problem of instability.

estimated and correct values without disturbance. According to the results obtained without regularization (Figure 11(a)), the difference between the estimated and correct joint angles were large and unstable. In contrast, with regularization, we found that the fluctuation of the joint angles became small and that they could be estimated with stability (Figure 11(b)). The large variance in the mean square error is clearly shown in Table 5(a), while the stable mean square error is shown in Table 5(b). These results confirm that we could estimate stable joint angles from given uneven end-effector coordinates through network inversion with regularization.

6. Conclusion

In this paper, we applied network inversion to the inverse kinematics problem of estimating the joint angles of a robot arm from the given end-effector coordinates. We then examined different aspects of ill-posedness in the inverse kinematic problem for a robot arm. We showed that network inversion could detect the nonexistence of a solution and estimate an approximate or provisional solution. We also confirmed that network inversion with regularization provided a unique or stable solution to the ill-posed problems. Through our results, we demonstrated

Table 5: Estimated joint angles, calculated end-effector coordinates, and errors(a) with regularization $F_3(x)$ and (b) without regularization for ill-posed problem of instability.

(a)

	Joint angles (°) $(\theta_1, \theta_2, \theta_3)$	End-effector coordinates (x, y)	MSE of end-effector coordinates
1	(53.67, 152.29, 96.80)	(60.98, 12.08)	1.56
2	(50.75, 165.75, 117.70)	(70.11, 28.02)	12.69
3	(57.45, 178.11, 90.36)	(57.96, 33.22)	3.81
4	(55.85, 188.36, 104.63)	(59.33, 46.03)	12.38
5	(28.37, 168.91, 285.48)	(38.81, 48.39)	1.69
6	(37.31, 183.61, 248.99)	(36.31, 66.04)	12.45
7	(34.15, 195.49, 265.50)	(22.99, 60.86)	2.20
8	(42.89, 211.92, 230.95)	(12.31, 73.71)	11.75

(b)

	Joint angles (°) $(\theta_1, \theta_2, \theta_3)$	End-effector coordinates (x, y)	MSE of end-effector coordinates
1	(53.67, 152.29, 96.80)	(60.98, 12.08)	1.56
2	(55.03, 163.23, 101.18)	(63.54, 23.65)	4.82
3	(59.11, 174.72, 92.30)	(58.02, 33.24)	3.87
4	(60.25, 182.58, 94.85)	(56.34, 41.34)	8.12
5	(66.21, 196.54, 75.65)	(43.78, 46.17)	3.89
6	(67.99, 208.19, 78.62)	(37.89, 54.92)	6.40
7	(71.44, 241.13, 61.29)	(18.38, 57.72)	3.50
8	(74.08, 253.47, 69.53)	(6.84, 63.03)	4.22

the effectiveness of applying network inversion to ill-posed inverse kinematic problems. We consider that the results suggest the applicability of network inversion to general ill-posed inverse problems.

In the future, we intend to search for a more effective regularization method for actual inverse kinematics problems by examining different types of regularization methods. In addition, we intend to examine the inverse estimation of joint angles for a multi-DOF robot arm.

References

[1] J. J. Craig, *Introduction to Robotics Mechanics and Control*, Addison-Wesley, Reading, Mass, USA, 1989.

[2] C. W. Groetsch, *Inverse Problems in the Mathematical Sciences*, Friedrich Vieweg & Sohn, 1993.

[3] M. Kawato, K. Furukawa, and R. Suzuki, "A hierarchical neural-network model for control and learning of voluntary movement," *Biological Cybernetics*, vol. 57, no. 3, pp. 169–185, 1987.

[4] W. T. Miller, R. S. Sutton, and P. J. Werbos, *Neural Networks for Control*, MIT Press, Cambridge, Mass, USA, 1990.

[5] E. Oyama and S. Tachi, "A study of human hand position control learning–output feedback inverse model," in *Proceedings of the International Joint Conference on Neural Networks*, pp. 1434–1443, 1991.

[6] A. Linden and J. Kindermann, "Inversion of multilayer nets," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN '89)*, pp. 425–430, June 1989.

[7] B.-L. Lu, H. Kita, and Y. Nishikawa, "Inverting feedforward neural networks using linear and nonlinear programming," *IEEE Transactions on Neural Networks*, vol. 10, no. 6, pp. 1271–1290, 1999.

[8] C. A. Jensen, R. D. Reed, R. J. Marks II et al., "Inversion of feedforward neural networks: algorithms and applications," *Proceedings of the IEEE*, vol. 87, no. 9, pp. 1536–1549, 1999.

[9] W. R. Murray, C. T. Heg, and C. M. Pohlhammer, "Iterative inversion of a neural network for estimating the location of a planar object," in *Proceedings of the World Congress on Neural Networks*, vol. 3, pp. 188–193, 1993.

[10] I. Valova, K. Kameyama, and Y. Kosugi, "Image decomposition by answer-in-weights neural network," *IEICE Transactions on Information and Systems*, vol. E78-D, no. 9, pp. 1221–1224, 1995.

[11] J. Wang, Q. Hu, and D. Jiang, "A Lagrangian network for kinematic control of redundant robot manipulators," *IEEE Transactions on Neural Networks*, vol. 10, no. 5, pp. 1123–1132, 1999.

[12] T. Ogawa, H. Matsuura, and H. Kanada, "A solution of inverse kinematics of robot arm using network inversion," in *Proceedings of the International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce*, pp. 858–862, November 2005.

[13] N. Sekiguchi, T. Ogawa, and H. Kanada, "Inverse estimation of joint angles of robot arm by network inversion," in *Proceedings of the Society of Instrument and Control Engineers Annual Conference (SICE '07)*, pp. 991–995, September 2007.

[14] A. N. Tikhonov and V. Y. Arsenin, *Solutions of Ill-Posed Problems*, Winstion and Sons, 1977.

[15] Y. P. Petrov and V. S. Sizikov, *Well-Posed, Ill-Posed, and Intermediate Problems with Application*, Koninklijke Brill NV, 2005.

[16] Y. Kosugi and K. Kameyama, "Inverse use of BP net in answer-in-weights scheme for arithmetic calculations," in *Proceedings of the World Congress on Neural Networks*, vol. 3, pp. 462–465, 1993.

[17] T. Ogawa, Y. Kosugi, and H. Kanada, "Neural network based solution to inverse problems," in *Proceedings of the IEEE International Joint Conference on Neural Networks*, pp. 2471–2476, May 1998.

[18] T. Poggio and F. Girosi, "Regularization algorithms for learning that are equivalent to multilayer networks," *Science*, vol. 247, no. 4945, pp. 978–982, 1990.

[19] B. Lu and K. Ito, "Regularization of inverse kinematics for redundant manipulators using neural network inversions," in *Proceedings of IEEE International Conference on Neural Networks*, pp. 2726–2731, December 1995.