Human-Robot Collision Detection Based on Neural Networks

Abdel-Nasser Sharkawy[1, 2*] and Nikos Aspragathos[2]
[1]Mechanical Engineering Department, Faculty of Engineering, South Valley University, Qena 83523, Egypt
[2] Department of Mechanical Engineering and Aeronautics, University of Patras, Rio 26504, Greece
Email: eng.abdelnassersharkawy@gmail.com, asprag@mech.upatras.gr

***Abstract*—In this paper, an approach based on a multilayer neural network has been proposed for human–robot collision detection. The neural network was trained using the Levenberg–Marquardt algorithm to the dynamics of the robot with and without external contacts to detect any unwanted collisions between the human operator and the robot using only the proprietary position and joint torque sensors of the manipulator. The proposed method was evaluated experimentally using the 7-DOF KUKA LWR manipulator, and the results indicate that the developed system is efficient and very fast in detecting collisions.**

***Index Terms*—Collision Detection, Neural Networks, Levenberg-Marquardt, Proprietary Sensors.**

I. Introduction

When robots and humans share the same workspace, safety is very important factor because the proximity of the operator to the robot can lead to potential injuries. Therefore, a system for safety based on collision avoidance or detecting the collision should be available. Collisions can be avoided by having knowledge of the environment using vision or proximity sensors. Mohammed et al. [1] introduced a solution based on vision using virtual 3D models of robots, real images of human operators from depth cameras and vision sensing units. Using proximity sensors, Lam et al. [2] presented an invisible sensitive skin built inside the robot arm using 5 contactless capacitive sensors and specially designed antennas. Although these methods can be used to avoid a collision, modifications to the robot body are required for installing the sensors and the cost is increased.

To further improve the safety of human-robot interaction (HRI) system, in addition to the level of collision avoidance, a second level of collision detection and reaction is required if the first level protection fails. Some researchers have sought to develop methods to detect collisions. Fault detection methods based on the principle that small changes or faults in a structure can cause significant deviations in its dynamic behavior have been proposed in robotics research. De Luca and Mattone's idea was to handle a collision at a generic point along the robot as a fault of its actuating system [3]. Cho et al. [4] continued their research and proposed the disturbance observer method to detect collisions based on the generalized momentum and on the joint torque sensors.

Another approach to detect collisions is based on impedance control. Morinaga and Kosuge [5] proposed a non-linear adaptive impedance control law without using external sensors, which is based on the difference between the reference and actual input torque to the manipulator.

Approaches based on fuzzy logic and neural network systems have been proposed. Dimeas et al. [6, 7] implemented two methods, one based on intelligent fuzzy identification and the other based on time series. The fuzzy system detected collisions rapidly and accurately and displayed the lowest threshold value. The time series system estimated the collision torque by only using the measured joint position signal, but its threshold was higher than the fuzzy system and similar to that of the model-based approach. Shujun Lu et al. [8] presented a collision detection approach using two six-axis force/torque sensors, one on the base and the other on the wrist, and they developed two systems, one based on neural network training and the other was model based. Although the results illustrated the validity of the developed collision detection scheme, using two sensors made the cost quite high.

In this paper, a neural network (NN)-based approach has been proposed considering the properties of the NN. The NN derives its computing power through its massively parallel distributed structure and its ability to learn and to consequently generalize. These two information processing capabilities make it possible for neural networks to find good approximate solutions to complex (large-scale) problems that are intractable. The NN also offers useful properties and capabilities, such non-linearity, input–output mapping, and adaptivity [9]. The NN can approximate any function, or in other words, they have a kind of universality [10], e.g., the approximation of smooth batch data containing the input, output and possibly the gradient information of a function [11], and approximating the derivatives of a function [12].

In the proposed method, the NN was trained using the Levenberg–Marquardt (LM) algorithm, which does not require any prior knowledge of the dynamic model of the robot and is simple and easily applied. This method was applied without using any external sensors, only using the position and joint torque sensors, which are proprietary to the KUKA LWR manipulator used for the experiments. Our method can be used in any robot containing joint torque sensors without any knowledge of its model since

Manuscript received November 14, 2017; revised February 6, 2018.

doi: 10.18178/ijmerr.7.2.150-157

the estimation of the external torque given by the robot controller is used only for training the NN and it can be replaced by an external sensor. Subsequently, the trained NN was used to estimate the external torque and detect the collision. The NN toolbox in Matlab was applied to the data for fast training and convergence.

II. Collision Detection Method

The dynamics of an n-link robot assumed for the flexible joint robot can be defined as follows [13]:

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau + DK^{-1}\dot{t} + \tau_{ext}, \quad (1)$$

$$B\ddot{\theta} + \tau + DK^{-1}\dot{t} = \tau_m, \text{ and} \quad (2)$$

$$\tau = K(\theta - q), \quad (3)$$

where the vectors $q, \dot{q}, \ddot{q} \in R^n$ contain the joint positions of the manipulator and their corresponding time derivatives, $\theta \in R^n$ is the measured motor position, $M(q) \in R^{n\times n}$ is the inertia matrix that depends on the variable q and contains unknown constant terms, $C(q,\dot{q}) \in R^{n\times n}$ is a matrix that contains the Coriolis and centrifugal terms that depends on the variables q and $\dot{q}$ and contains unknown constant terms, $G(q) \in R^n$ is the gravity vector that depends on the variable q and contains unknown constant terms. $K = diag(K_i) \in R^{n\times n}$ and $B = diag(B_i) \in R^{n\times n}$ are the diagonal, positive definite joint stiffness, and motor inertia matrices, respectively, and they are unknown constants, and $D = diag(D_i) \in R^{n\times n}$ is the diagonal positive semi-definite joint damping matrix and it is an unknown constant. The vector $\tau \in R^n$ represents the measured joint torque, $\tau_m \in R^n$ is the motor input torques vector and $\tau_{ext} \in R^n$ is the external torque vector from the collision acting on the robot and it can be calculated from (1) as follows:

$$\tau_{ext} = M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) - \tau - DK^{-1}\dot{t}. \quad (4)$$

Because it is difficult to determine the unknown dynamic coefficients in the dynamic model of the manipulator, as shown from (1) to (4), a neural network based a non-linear estimation model was used in this paper to approximate the τ_{ext} function given in (4). In the proposed method, a NN was trained using the data with and without the collision measured by the joint position and torque sensors. In this method, the estimation of the external torque given by the robot controller is used only for training the NN, but this could be measured by any other external sensor. Therefore, our method can be used in any robot equipped with joint torque sensors without any prior knowledge of its model.

After many trials and experiments using different sets of inputs to train the network using the LM algorithm, it was found that the main inputs to the neural network that give us the minimum mean squared error (mse) are the current position error $q_e(k)$ between the desired and actual joint position, the previous position error $q_e(k-1)$, the actual joint velocity $\dot{q}$, and the measured joint torque τ. We found that these inputs are greatly influenced by the presence of a collision. However, large alterations in these variables are not always indicators of a collision because these changes can also be observed under normal operation owing to the high inertial forces that occur during abrupt changes of the velocity. Since this phenomenon is related to the joint velocity, it is the reason for using the signal of the actual joint velocity $\dot{q}$ as an input for the training in order to distinguish the collision spikes; this is also discussed in [6, 7]. In addition, it was found that while using the actual joint velocity, the mean squared error is significantly reduced during training the NN. Although the signals from the torque sensors include a small noise, no digital filters were used to avoid any delay. To ensure that this assumption is correct, a sinusoidal motion q with variable frequency was commanded on a single joint of the KUKA LWR robot (Joint E1), and the collisions were performed by a human hand touching the manipulator, as shown in Fig. 1, where the $q_e, \dot{q}$, and τ inputs to the NN without and with a collision are illustrated.

From Fig. 1, it is clear that there are sudden changes and spikes in the variables in the case of a collision when compared with the corresponding variables without a collision. Without a collision, the small alterations and spikes in the position error and measured joint torques are a result of the friction and high inertial forces that appear on the links (blue dashed curves). In the case of a collision, the spikes in the position error and the measured joint torque diagrams (small black spot) that face the small spikes in the actual joint velocity diagram represent the collision and the other spikes come from the inertial forces. The spike at the end of the position error diagram (small blank spot) in the two curves means that the robot starts to brake. To show clearly what the small spikes in the actual joint velocity diagram represents, for example, the interval [0, 12] seconds from Fig. 1 is expanded in Fig. 2, where the small spikes at points **c** represent the collisions, whereas the spikes at points **i** come from the inertial forces.

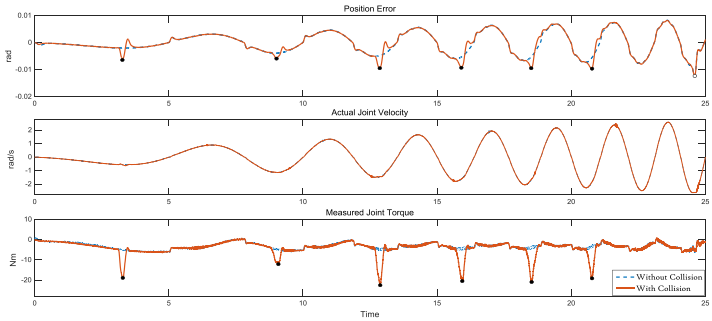


Figure 1. The main inputs of the neural network.

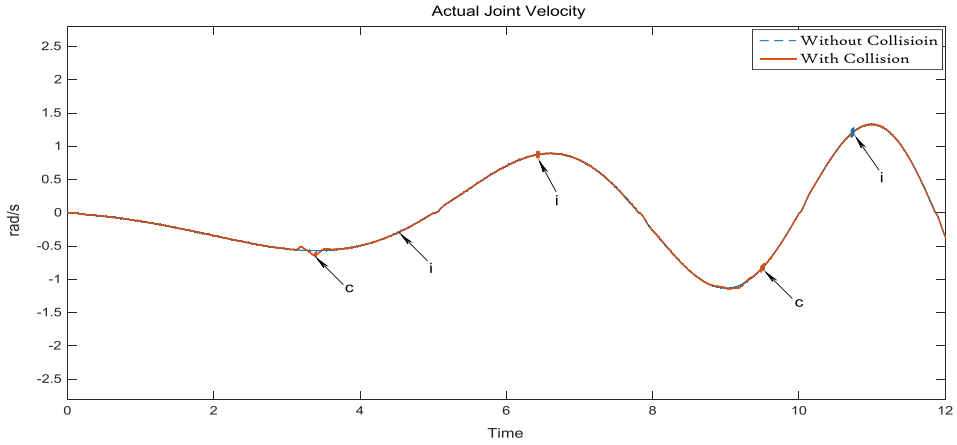


Figure 2. The small spikes in the actual joint velocity diagram that differentiate between the collisions and inertial forces.

III. Neural Network Design

A multilayer neural network is a powerful tool for non-linear system identification. It can adapt itself by changing the network parameters in a surrounding environment and can easily handle imprecise, fuzzy, noisy, and probabilistic information [14]. The NN has a vital role in the identification of dynamic systems and fault detection since it can not only detect the occurrence of the fault but also provides a postfault model of the robotic manipulator. This post-fault model can be effectively used to isolate and identify the fault and, if possible, for accommodation of the failure [15].

Levenberg–Marquardt learning was used in this study to train the network and perform the work in a fast and stable manner. The Levenberg–Marquardt algorithm is a type of the second-order optimization technique that has a strong theoretical basis and provides a significantly fast convergence; it is considered as an approximation to Newton's method [16, 17]. When compared with other learning methods, LM learning was used because it has the trade-off between the fast learning speed of the classical Newton's method and the guaranteed convergence of the gradient descent [16, 18]. The LM algorithm always suits larger data sets and converges in less iterations in shorter times than the other training methods.

Three layers were used to compose the network, as shown in Fig. 3. The first layer is the input layer, which contains the four inputs for the NN including the current joint position error $q_e(k)$, the previous position error $q_e(k-1)$, the actual joint velocity $\dot{q}$, and the measured joint torque τ. The second layer is the non-linear hidden layer, and the third is the output layer, which calculates the estimated external torque τ'_{ext} that is compared with the estimated external torque τ_{ext} derived using the Kuka robot controller (KRC). It should be noted here that the estimated external torque τ_{ext} is only used for training the network. The external collision force can be measured using any external sensor and transformed to the joint torque using the Jacobian.

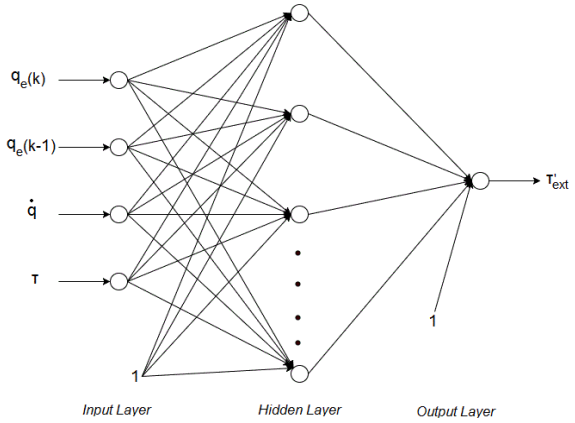


Figure 3. The multilayer neural network.

The training error $e(t)$ should be as small as possible. From the block diagram that illustrates the process of training the neural network shown in Fig. 4, the error is given as follows:

$$e(t) = \tau_{ext} - \tau'_{ext} \tag{5}$$

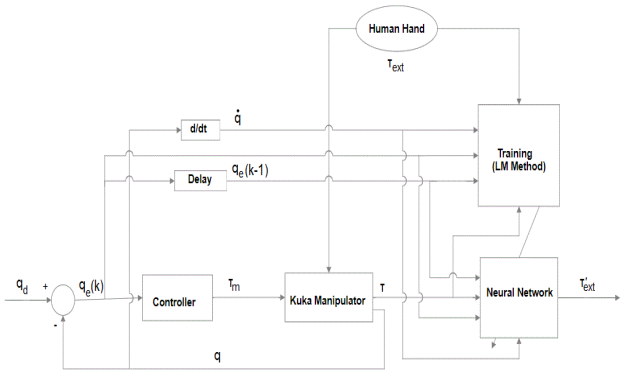


Figure 4. The block diagram of the neural network system trained using the LM algorithm.

IV. Experiments

The proposed contact detection method is implemented on a KUKA LWR IV manipulator (light weight robot), as shown in Fig. 5. The KUKA LWR robot is characterized by an extremely light anthropomorphic structure with 7

revolute joints driven by compact brushless motors via harmonic drives. The presence of such transmission elements introduces a dynamically time-varying elastic displacement at each joint between the angular position of the motor and that of the driven link. All joints are equipped with position sensors on the motor and link sides, and a joint torque sensor. The KR C5.6 lr robot controller unit, along with the so-called fast research interface (FRI) [19], is able to provide (at a 1 msec sampling rate) the link position q, velocity $\dot{q}$ and joint torque τ measurements, and an estimation of the external torque τ_{ext}.

Figure 5. The experimental set-up using the KUKA LWR manipulator.

The robot performs a single joint motion around the vertical axis. The selected excitation signal $q_d(t)$ is a sinusoidal profile of the joint position with variable frequency since it enables the sufficient dynamic excitation of the structure and the acquisition of rich signals. The motion of joint E1 is given as follows:

$$q_d(t) = -A + Acos(2\pi ft) \quad (6)$$

where $A = \frac{\pi}{4}$ and f is the frequency, which is linearly increased from 0.05 to 0.326 Hz. This frequency produces an angular velocity ω of up to 2.05 rad/s.

The training data are divided into two sets. In the first set, the robot joint performs the motion without any external force applied to the robot body and in the second set the same motion is performed with the user performing collisions suddenly and stochastically with their hand. During the experiments the robot is commanded to move with position control mode and no reaction strategy was implemented. The performed collisions were applied momentarily and the safety of the human operator during this experiment was considered.

A. NN Training

Using a combination of the data with and without a collision, the neural network system was created and trained. The total number of input-output pairs collected from the experiments and used was 56358. From these data, 90% are used for training, 5% for validation, and 5% for testing. After more trials and initializations, it was found that the best number of hidden neurons was 90 and the number of iterations was 932 to give the minimum mse and an adequate collision threshold. The training process was very fast and stable, and applied using Matlab on an *Intel(R) Core(TM) i3-6100 CPU @ 3.70 GHz* processor. The experiments for NN structure determination and training are discussed in APPENDIX A.

The trained NN was evaluated using the same data set used for the training process. The difference between the external torque τ_{ext} given by the KRC and the external torque τ'_{ext} estimated by the NN system is illustrated in Fig. 6. It is clear that the approximation error of the collision torque is high when compared to the error of contact-free motion (blue dashed curve), where the average of the absolute error values is very small (0.0955 Nm) and the maximum value of the absolute value of the error is 1.6815 Nm that was used as the collision threshold above which, a collision is assumed, when $|\tau'_{ext}| > 1.6815$. In the literature, the collision thresholds are defined in different ways. In [20], the threshold was defined as 10% of the maximum nominal torque of the robot. Dimeas et al. [6] used the maximum training error from the contact-free motion training, and in [8], it was defined as a value below the contact force that represents the unified pain tolerance limit of a human, which was determined in [21]. In this paper, the threshold was identified as that in [6], and after its calculation, it was also found to be near to 10% of the maximum nominal torque of the robot.

To examine carefully the approximation error, the estimated external torque resulting from the KRC and the estimated external torque from the NN were compared, as shown in Fig. 7.

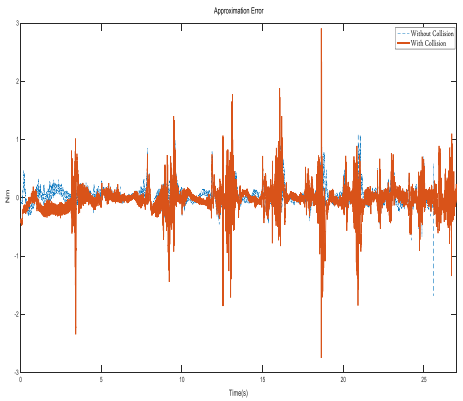


Figure 6. The neural network approximation error.

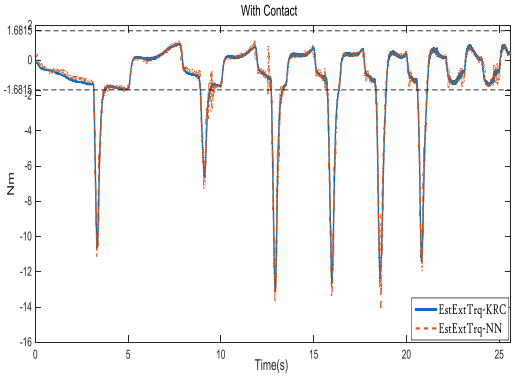


Figure 7. The two estimated external collision torques obtained from the KRC and NN.

Using the proposed method, the collision can be identified rapidly. For instance using the interval [2.5, 4.5] s from Fig. 7, where the first collision occurs, the collision detection time was easily calculated, as shown in Fig. 8. The collision detection time was calculated as the elapsed time from the start of a collision (point **a** in the curve, where the slope starts to monotonically increase) to the moment when the estimated external collision torque by the trained NN exceeds the threshold (point **b** in the curve). From Fig. 7, the collision detection time was determined to be 11.5 ms after the collision occurred.

B. Verification and Testing

Since the training data were obtained using the variable velocity only, the proposed method was tested in the experimental set-up by commanding the robot to perform a single joint motion around the vertical axis with constant velocity profiles to verify that the method was able to identify the collision and consider its ability to generalize under a variety of conditions. The trained NN was evaluated with two different speeds (0.5 rad/s and 1.0 rad/s), as shown in Fig. 9. Six collisions were used with different values and directions.

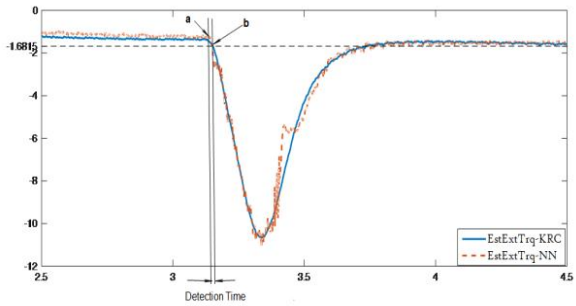
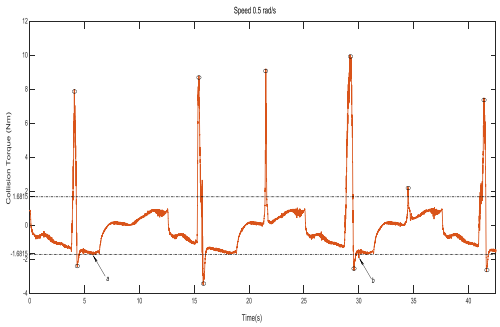
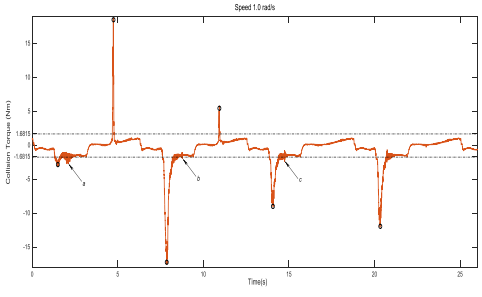


Figure 8. Calculation the collision detection time for the first collision.



(a) At 0.5 rad/s.



(b) At 1.0 rad/s

Figure 9. The estimated external collision torque using the proposed method.

At 0.5 rad/s (Fig. 9a), the trained NN is able to identify all the collisions (small rings in the curve), but two collision alerts were falsely detected since they did not correspond to an actual collision (points **a** and **b** in the curve). The collision detection time was low, e.g. during the second collision, the collision detection time was 31.92 ms. At 1.0 rad/s (Fig. 9b), all the collisions were detected (small rings in the curve), but three collision alerts were falsely detected (points **a**, **b,** and **c** in the curve). The collision detection time during the second collision was 13.3 ms.

To confirm the validity and efficiency of the proposed method under a wide range of operating conditions and to acquire a performance measure, another 25 trials of collisions (NC) were evaluated with various magnitudes, directions and different velocities of motion. Table I provides the performance in terms of the number of the correct detected collisions (CC), the number of false negatives (FN), which is the number of collisions not detected by the method and the number of the false positives (FP), which are the collision alerts provided by the method when there was no actual collision.

TABLE I. THE PERFORMANCE OF THE COLLISION DETECTION METHOD FOR DIFFERENT COLLISION SCENARIOS AND VELOCITIES.

Method	NC	CC	FN	FP
NN trained by LM	25	21	4	2
Percentage		84%	16%	8%

Conventions: NC = number of collisions, CC = number of correctly detected collisions, FN = number of false negatives and FP = number of the false positives.

Table I shows that the proposed method (NN) successfully detects the collisions with a high percentage (84%). Also, the number of the false positive collisions is low (8%), which means that our method is robust and less sensitive to the external disturbances and unmodelled parameters.

V. DISCUSSION

The proposed method is easily applied and understood, and the training is very fast. The fuzzy-based part of [6] was compared with our method. The estimated external torque τ_{ext} given by the Kuka robot controller is only used for NN training step, while Dimeas et al. [6] used the external torque measured by an ATI F/T Nano 25 force sensor for verification and training. The data used in our proposed method for training the NN comprises 56,358 input-output pairs, which is lower than the data used by Dimeas et al., which used 70,000 input–output pairs. The average error in the contact-free motion using our method was 0.0955 Nm, which is slightly lower than that found by Dimeas et al., which is 0.1 Nm. Using the previous position error $q_e(k-1)$ in the proposed method, the convergence and reliability were improved when compared to the work of Dimeas et al.

In [8], the cost of the method proposed by Lu et al. was quite high because two force sensors were used, whereas our method does not require any external sensors and can be used with any robot containing joint torque sensors without any prior model knowledge.

Our method also presents a low detection time for collisions. It should be noted that because of the different data and operating conditions used in our paper and the other two papers, it is difficult to quantitatively compare the time required for collision detection.

VI. Conclusions and Future Work

In this paper, a method has been proposed for human–robot collision detection based on a multilayer neural network trained using the Levenberg–Marquardt algorithm. The training was stable and very fast. The inputs used for the NN were derived from the position and joint torque sensors, and the method was able to detect the collision of the robot with a human hand very quickly. The NN system was designed by trial and error to evaluate the external collision torque and was trained using the data collected with and without a collision after conducting the experiments based on the motion of a single joint (Joint E1) of the 7-DOF KUKA LWR robot. Consequently, a rich signal was obtained and the training error was very small.

Owing to the good results obtained in this study, future work will involve the extension of the proposed approach by implementing the collision detection system to multiple joints in the manipulator, where apart from detecting the occurrence of a collision, the collided link will also be identified.

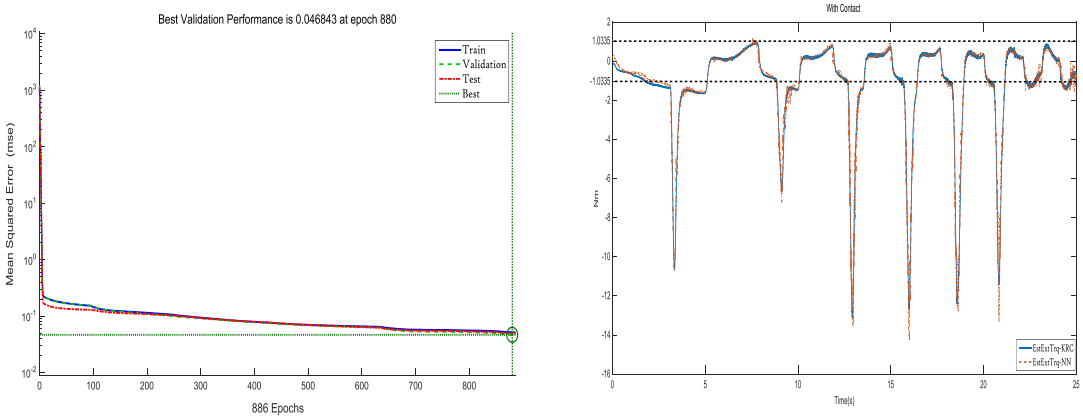
Appendix A

Many experiments were conducted using different sets of inputs to train the NN using the LM algorithm with the data obtained with and without contact. For every set of inputs, different numbers of hidden neurons, starting from 30 to 120 hidden neurons, were used to train the NN and with every number of hidden neurons, many different initializations of the NN were used until the minimum and best mean squared error, and an adequate collision threshold were obtained. Only three cases from the experiments are presented here to show the process of NN structure determination and training.

Case 1 (Fig. 10): The current position error $q_e(k)$, the actual joint velocity $\dot{q}(k)$, and the measured joint torque τ were used as the three inputs to the NN. After many trials of using different numbers of hidden neurons and initializations, it was found that the lowest mse (0.046843) was obtained using 120 hidden neurons, as shown in Fig. 10a. The average absolute error value of contact-free motion was 0.0856 Nm and the collision threshold was 1.0335 Nm. Using this threshold value, the proposed method fails to properly detect the collisions, and the performance was poor because most of the collisions start at a point below the threshold value and there are FP, which are the collision alerts when no actual collision occurs, as shown in the interval [22, 25] s of Fig. 10b.

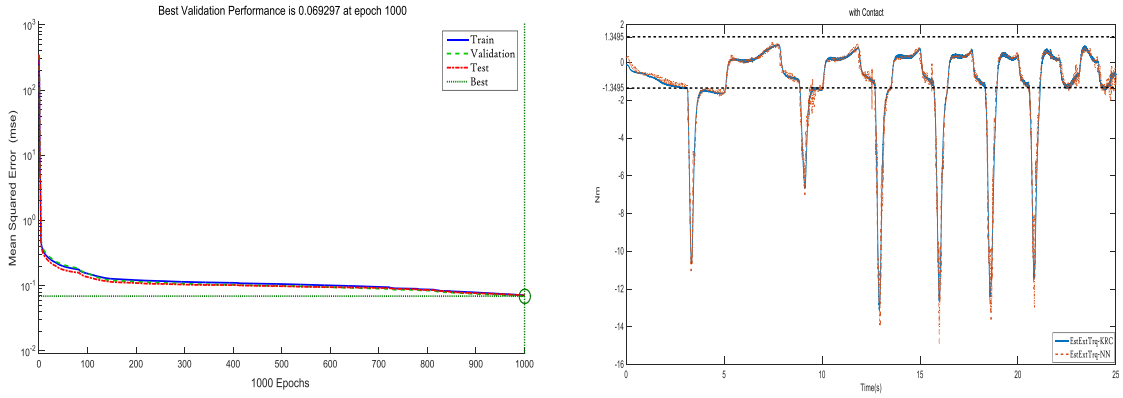
Case 2 (Fig. 11): The current position error $q_e(k)$, the current actual joint velocity $\dot{q}(k)$, the previous actual joint velocity $\dot{q}(k-1)$, and the measured joint torque τ were used as the inputs to the NN. After many trials using different numbers of hidden neurons and initializations, it was found that the lowest mse (0.069297) was obtained using 70 hidden neurons, which is higher than case 1, as shown in Fig. 11a. The average absolute error value of contact-free motion was 0.1010 Nm, which is also high when compared with the other cases. The collision threshold was 1.3495 Nm and using this value, the proposed method was better than case 1, but it gives a lot of FP (> 5 FP), as shown in Fig. 11b, so the performance of the method is also not good.

Case 3: This is the best case used in this paper, where the current position error $q_e(k)$, the previous position error $q_e(k-1)$, the actual joint velocity $\dot{q}(k)$ and the measured joint torque τ were used as the inputs to the NN. After many trials of using different numbers of hidden neurons and initializations, it was found that the lowest mse (0.040644) was obtained using 90 hidden neurons, which is the lowest value compared with the all the cases studied, as shown in Fig. 12. The average absolute error values of contact-free motion was 0.0955 Nm. The collision threshold was 1.6815 Nm, and by using this value, the proposed method is the best case, which can detect the collisions accurately and succeeds with a high percentage as discussed in the paper and shown in Fig. 7.



(a) (b)

Figure 10. Case 1: Using the current position error $q_e(k)$, the actual joint velocity $\dot{q}(k)$ and the measured joint torque τ as the inputs to the NN. (a) The lowest mse value. (b) The two estimated external collision torques obtained from the KRC and NN.



(a) (b)

Figure 11. Case 2: Using the current position error $q_e(k)$, the current actual joint velocity $\dot{q}(k)$, the previous actual joint velocity $\dot{q}(k-1)$ and the measured joint torque τ as the inputs to the NN. (a) The lowest mse value. (b) The two estimated external collision torques obtained from the KRC and NN.

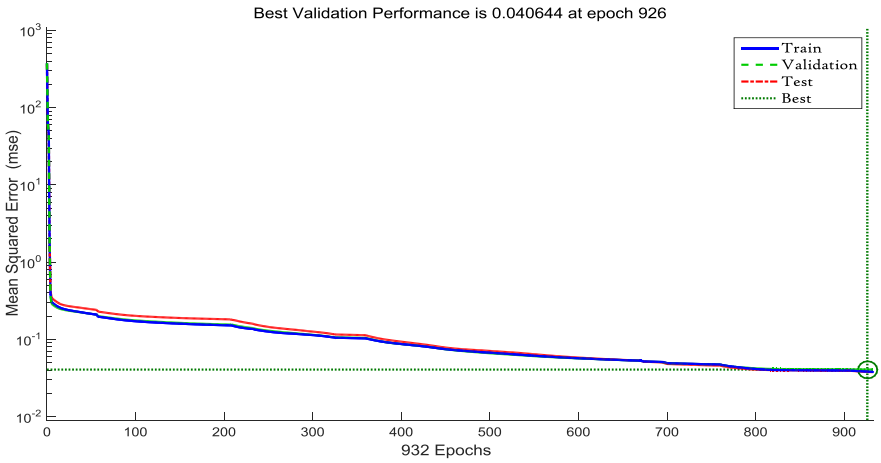


Figure 12. Case 3: The lowest mse value was obtained using the current position error $q_e(k)$, the previous position error $q_e(k-1)$, the actual joint velocity $\dot{q}(k)$ and the measured joint torque τ as the inputs to the NN.

ACKNOWLEDGMENT

Abdel-Nasser Sharkawy is funded by the "Egyptian Cultural Affairs & Missions Sector" and "Hellenic Ministry of Foreign Affairs Scholarship" for Ph.D. study in Greece. This paper is partly sponsored by the programme of introducing Talents of Discipline to University ("111 Program") under Grant No. B16034.

REFERENCES

[1] A. Mohammed, B. Schmidt, and L. Wang, "Active collision avoidance for human – robot collaboration driven by vision sensors," *Int. J. Comput. Integr. Manuf.*, vol. 30, no. 9, pp. 970–980, 2017.

[2] T. L. Lam, H. W. Yip, H. Qian, and Y. Xu, "Collision avoidance of industrial robot arms using an invisible sensitive skin," in *Proc. 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 4542–4543.

[3] A. De Luca and R. Mattone, "Sensorless robot collision detection and hybrid force / motion control," in *Proc. the 2005 IEEE International Conference on Robotics and Automation*, April 2005, pp. 999–1004.

[4] C. Cho, J. Kim, S. Lee, and J. Song, "Collision detection and reaction on 7 DOF service robot arm using residual observer," *J. Mech. Sci. Technol.*, vol. 26, no. 4, pp. 1197–1203, 2012.

[5] S. Morinaga and K. Kosuge, "Collision detection system for manipulator based on adaptive impedance control law," in *Proc. the 2003 IEEE International Conference on Robotics &Automation*, 2003, pp. 1080–1085.

[6] F. Dimeas, L. D. Avenda, E. Nasiopoulou, and N. Aspragathos, "Robot collision detection based on fuzzy identification and time series modelling," in *Proc. the RAAD 2013, 22nd International Workshop on Robotics in Alpe-Adria-Danube Region*, 2013.

[7] F. Dimeas, L. D. Avendano-valencia, and N. Aspragathos, "Human - Robot collision detection and identification based on fuzzy and time series modelling," *Robotica*, pp. 1–13, May 2014.

[8] S. Lu, J. H. Chung, and S. A. Velinsky, "Human-robot collision detection and identification based on wrist and base force / torque sensors," in *Proc. 2005 IEEE International Conference on Robotics and Automation*, April 2005, pp. 796–801.

[9] S. Haykin, *Neural Networks and Learning Machines*, Third Edit. Pearson, 2009, pp.1-6.

[10] M. A. Nielsen, *"Neural networks and deep learning."* Determination Press, 2015, ch.4.

[11] S. Ferrari and R. F. Stengel, "Smooth function approximation using neural networks," *IEEE Trans. NEURAL NETWORKS*, vol. 16, no. 1, pp. 24–38, 2005.

[12] K. Hornik, M. Stinchcombe, and H. White, "Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks," *Neurul Networks*, vol. 3, pp. 551–560, 1990.

[13] M. W. Spong, "Modeling and control of elastic joint robots," *IEEE J. Robot. Autom.*, vol. 109, no. 4, pp. 310–319, 1987.
[14] K. L. Du and M. N. S. Swamy, *Neural Networks in a Softcomputing Framework*. London: Springer, 2006.
[15] A. T. Vemuri and M. M. Polycarpou, "Neural-network-based robust fault diagnosis in robotic systems," *IEEE Trans. NEURAL Networks*, vol. 8, no. 6, pp. 1410–1420, 1997.
[16] K. Du and M. N. S. Swamy, *Neural Networks and Statistical Learning*. Springer, 2014, ch.5, pp.127-133.
[17] D. W. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters," *J. Soc. Ind. Appl. Math.*, vol. 11, no. 2, pp. 431–441, 1963.
[18] M. T. Hagan and M. B. Menhaj, "Training feedforward networks with the marquardt algorithm," *IEEE Trans. NEURAL NETWORKS*, vol. 5, no. 6, pp. 2–6, 1994.
[19] *KUKA.FastResearchInterface 1.0*, *KUKA System Technology (KST)*, version 2, D-86165 Augsburg, Germany, 2011.
[20] S. Haddadin, A. Albu-sch, A. De Luca, and G. Hirzinger, "Collision detection and reaction : A contribution to safe physical human-robot interaction," in *Proc. 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008, pp. 3356–3363.
[21] Y. Yamada, Y. Hirasawa, S. Huang, Y. Umetani, and K. Suita, "Human – robot contact in the safeguarding space," *230 IEEE/ASME Trans. MECHATRONICS*, vol. 2, no. 4, pp. 230–236, *1997.*

Abdel-Nasser Sharkawy was born in Qena, Egypt in September 29, 1991, received his B.Sc. degree and M.Sc. degree in Mechanical Engineering (Mechatronics Division) from Faculty of Engineering, South Valley University, Qena, Egypt in 2013 and 2016. He started working as an assistant lecturer at the same university from 2013. Currently he is pursuing his Ph.D. degree in Mechanical Engineering (Robotics Group) at University of Patras, Patras, Greece in the field of Human-Robot Cooperation. His PhD thesis is funded by the "Egyptian Cultural Affairs & Missions Sector" and "Hellenic Ministry of Foreign Affairs Scholarship". His research areas of interest include mechatronic systems, human - robot interaction, robot control and rehabilitation.

Nikos Aspragathos (Professor) leads the Robotics Group in Mechanical & Aeronautics Engineering Department, University of Patras, Greece. His main research interests are robotics, intelligent motion planning and control for static and mobile robots and for dexterous manipulation of rigid and non-rigid objects, knowledge-based design, industrial automation, and computer graphics. He is reviewer for about 40 Journals and more than 30 conferences, member of the editorial board of the Mechatronics Journal, ROBOTICA and ISRN Robotics. He published more than 70 papers in Journals and more than a 130 in conference proceedings. He was and is currently involved in research projects funded by Greek and European Union sources.