Hindawi
Journal of Robotics
Volume 2021, Article ID 5805232, 10 pages
https://doi.org/10.1155/2021/5805232

Research Article

Voice Recognition and Inverse Kinematics Control for a Redundant Manipulator Based on a Multilayer Artificial Intelligence Network

Mai Ngoc Anh and Duong Xuan Bien

Le Quy Don Technical University, 236 Hoang Quoc Viet, Hanoi, Vietnam

Correspondence should be addressed to Mai Ngoc Anh; maingocanh@lqdtu.edu.vn

Received 7 May 2021; Accepted 20 June 2021; Published 29 June 2021

Academic Editor: L. Fortuna

This study presents the construction of a Vietnamese voice recognition module and inverse kinematics control of a redundant manipulator by using artificial intelligence algorithms. The first deep learning model is built to recognize and convert voice information into input signals of the inverse kinematics problem of a 6-degrees-of-freedom robotic manipulator. The inverse kinematics problem is solved based on the construction and training. The second deep learning model is built using the data determined from the mathematical model of the system's geometrical structure, the limits of joint variables, and the workspace. The deep learning models are built in the PYTHON language. The efficient operation of the built deep learning networks demonstrates the reliability of the artificial intelligence algorithms and the applicability of the Vietnamese voice recognition module for various tasks.

1. Introduction

In recent years, control system designs have been developed in the trend of intelligent control systems but still ensure a fast and flexible response in real time to constantly changing control requirements and allow for high-precision human interaction.

In conventional intelligent control systems, research on voice-based control is attracting many scientists thanks to its user-friendly interaction. Among the voice-based control systems of industrial robotics, users can have the robots perform a variety of tasks through simple commands that carry control information related to the motion direction and the characteristics of the object.

In essence, the voice commands are used as the input of the control system to solve the problem of inverse kinematics (IK) and then converted into various operations of the manipulator. Due to the diverse nature of voice commands, the manipulator tasks change constantly, requiring the control system to be processed quickly to respond. The IK solving algorithms such as analytic methods [1] or numerical methods such as AGV [2], CLIK [3], and Jacobi transpose [4] are hardly suitable, especially for redundant manipulator systems.

The results of recent research on artificial intelligence (AI) show that neural networks (NN), deep learning, and reinforcement learning algorithms are extremely useful and effective for dealing with complex nonlinear problems with cost savings in computation time and system resources [5]. The most important point when applying these algorithms is to have a good understanding of the network structure built up and its functioning. The quality of the network and the performance of the network will be used as criteria to evaluate the effectiveness of the algorithms. In terms of programming languages, AI-related networks can be built on different languages like PYTHON, C ++, and Java [6]. However, the PYTHON language has recently become more suitable for building deep learning (DL) network structures with efficient support libraries such as Tensorflow, PyTorch, Numpy, Keras, and Sklearn. More importantly, these libraries support optimization problems in data science, machine learning, and control [7]. Based on the outstanding advantages of AI techniques, many intelligent control systems have been built to solve IK problems for redundant manipulation systems. Furthermore, these AI techniques are well suited for control systems that require constantly changing motion by voice commands that may not be preprogrammed.

Many solutions to apply voice control systems based on AI algorithms for industrial machines are mentioned in [8]. To determine the direction of the emitted sound source, Hwang et al. [9] designed an intelligent ear for the robot arm. To control the fabrication machine and industrial robotic arms, Rogowski [10] designed a VCS solution with good noise resistance. For serving services, multiple manipulators designed to be human friendly interactively through gesture recognition and voice feedback are introduced in [11–13]. The manipulator in [14] serving household chores is controlled by VCS to increase the usability and entertainment. An enhanced version of DL algorithm-based speech recognition is proposed in [15]. The medical robot arm in [16] is designed with a VCS that allows the nurses and patients to easily interact with the robot. The manipulator in [17] uses a VCS with visible light communication. An autonomous manipulator is controlled by voice through the Google Assistant application tool on the basis of IoT technology and is shown in [18]. A voice-controlled application that uses IoT technology in combination with an adaptive NN is proposed in [19] to improve the efficiency of solving IK problems for 6-degrees-of-freedom (DOF) robots. Differently, a Bayesian-BP NN is built to create an efficient control system for the root mean square (RMS) with fast and precise learning [20]. The simulation results show that the error of the method is extremely small. The IK problem for a 2DOF manipulator using NN is presented in [21], 3DOF robot in [22], and 4DOF robot with hybrid IK control system NN and genetic algorithm in [23]. The NN has output feedback to solve the IK problem of the 6DOF manipulator proposed in [24]. This is a technique with very high control efficiency. A new algorithm in 5DOF manipulator control in real time on the basis of the NN is proposed in [25].

This study presents setting up of two deep learning networks DL1 and DL2 to process voice signals to take the input of a 6DOF redundant manipulator to solve the IK control problem. Control information in the voice tag includes the direction of movement and the object whose attributes are given in the speech. The robot will then conduct image recognition to determine the object has the appropriate attributes for voice recognition results from the sentence. The image recognition is performed through the computer's built-in vision module and will not be deeply analyzed in this study. The center coordinates of the object will represent the position the end-effector point of the manipulator needs to move to. Training data for model DL2 are taken from the results of the forward kinetics problem based on the kinematics modeling according to Denavit–Hartenberg's (DH) theory. The DL network models are built using the PYTHON language. Successfully solving these two problems has a wide range of potential applications in response to the constantly changing trajectory of the manipulator without preprograming.

2. Materials and Methods

2.1. The Diagram of Voice-Based Controller. The manipulator receives voice commands from the operator using the voice recognition module. Then, the control system automatically analyzes, calculates, and gives the control signals for the motors at the joints of the manipulator (Figure 1).

Specifically, the voice recognition module converts from human voice containing control information to text in the program. The manipulator control information contained in the voice includes information such as the direction of the movement of the manipulator (turn to the left or right), what action the manipulator needs to perform (the action of grabbing or dropping), identifying the object (wheels, tray, boxes, etc.), and distinguishing features of objects (color, shape, size, etc.).

The input voice and the output control signal must be defined to solve the manipulator control target. In essence, the voice recognition module is a natural language processing problem, and a DL model is built in order for the network to learn how to convert information from voice to text. The steps to perform the VCS are depicted in Figure 2.

2.1.1. Preprocessing the Input Voice. This problem is solved through the following substeps: noise filtering, word separation, converting sound oscillation into sound energy in the frequency domain, and converting this energy into input data for the DL1 model.

The noise filter step can be handled through a number of methods such as noise reduction based on the hardware design of the receiver microphone or by electronic elements of the circuit recording or by the program adjust. Voices include the main expected sounds we need to record and noises (unwanted sounds or no control information). These acoustic noises can come from the sounds of outside environments such as traffic and industrial noise. They often negatively affect the accuracy of speech recognition results. To significantly reduce audio noise, a noise reduction transceiver is used in this study.

Each human sentence often consists of many words combined. Each word contains one or several syllables. Thus, the speech recognition program must perform two basic tasks: separating words in sentences and separating syllables in each word.

Interestingly, every Vietnamese word has only one syllable. Therefore, this study only needs to focus on the first task, which is separating words in sentences. To better understand this problem, let us consider the following example.

Let consider a Vietnamese voice command to control the manipulator: "Quay bên phải, lấy bánh xe màu vàng" ("turn right, grab the yellow wheel" in English). It is noticed that the Vietnamese sentence has 8 syllables, while the English one has 7 syllables, of which "yellow" has 2 syllables.

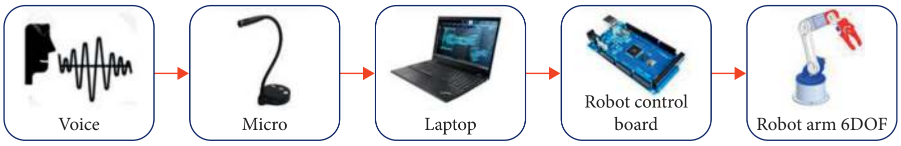


FIGURE 1: Diagram of voice control for the 6DOF manipulator arm.

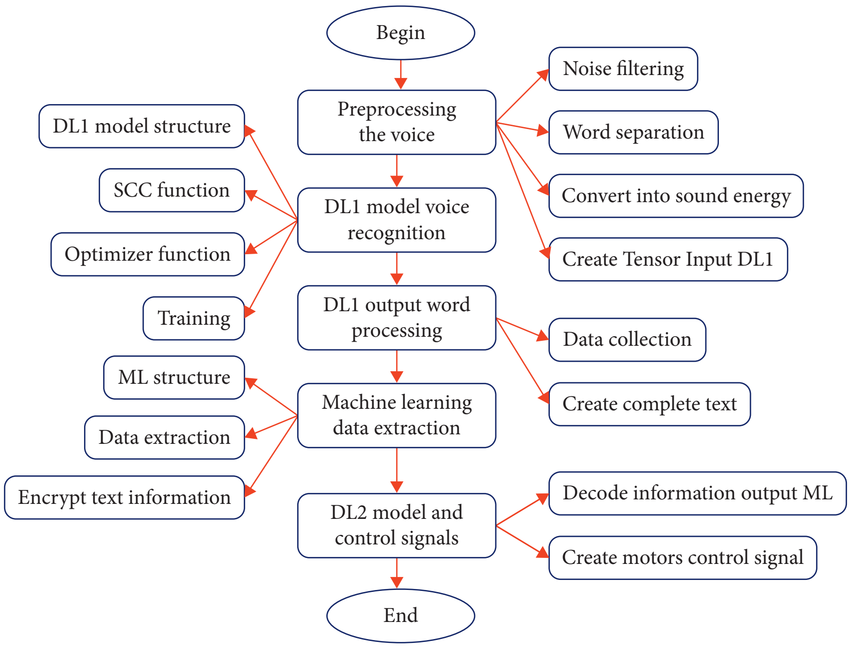


FIGURE 2: The steps to perform the VCS.

Voice is received through the microphone and recorded through the regular application Void Recorder available on Windows Microsoft operating system. The audio file can be read and written with *Scipy* library in PYTHON programming. Acoustic oscillation amplitude values are standardized so that the input signal does not contain a lot of suboscillations, making the separation process more efficient and easy to set a useful filter threshold. After normalization, decomposition is performed in the DL1 model with network node parameters that can be adjusted through a learning process on the sample to improve accuracy.

Acoustic oscillation amplitude values are normalized so that the input signal does not contain a lot of suboscillations, making the separation process more efficient and easy by setting a threshold filter. After normalization, the word decomposition is performed in the DL1 model with network node parameters that can be adjusted through a learning process on the sample to improve accuracy.

The acoustic oscillation amplitudes after normalization are shown in Figure 3. It can be seen that the difference of normalized amplitudes can be clearly distinguished when speaking and not speaking. This difference is used as a key feature to separate words in sentences.

However, it should be noted that areas with exceptionally large amplitude of sound fluctuations relative to other areas while speaking will be considered as acoustic noise in speech. In addition, oscillation regions with small

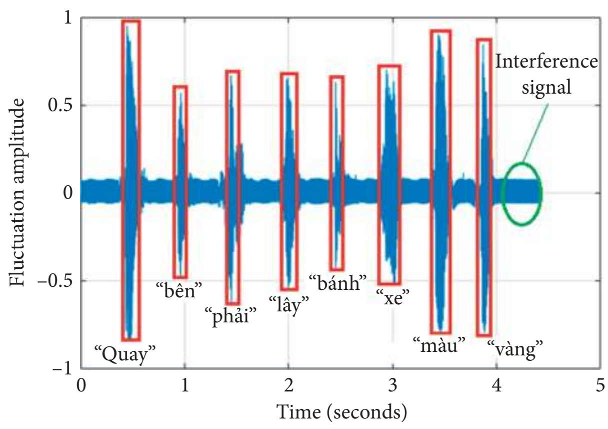


FIGURE 3: The normalized sound amplitude.

and fairly equal amplitudes are also considered noise signals that can be ignored. Therefore, if a user suddenly screams out a word or speaks all words in a sentence at low volume, the system may not understand the voice command.

The change in the amplitude of the sound oscillation is determined to separate the words using the gradient method [26]. After separating the words in the spoken sentence, the sound oscillation will be analyzed for the sound energy in the frequency domain through the Fourier transform. This sound energy value will be used to convert to Input Tensor for the DL model. The sound of the human voice is actually a combination of many signals with different frequencies. The oscillation function can be described through the following Fourier transform [17].

$$f(t) = \frac{1}{2}a_0 + \sum_{n=1}^{\infty} \left[a_n \cos(n\omega t) + b_n \sin(n\omega t)\right], \quad (1)$$

where a_0 is the original sound amplitude, a_n and b_n are the Fourier constants, n is the frequency ratio coefficient, ω is the angular velocity, and t is a time variable.

From equation (1), the sound energy value in the frequency domain can be specified [17]. Figure 4 shows the sound energy that illustrates the two words "Quay" (turn) and "Phải" (right) in the frequency domain.

A fundamental characteristic of sound is the energy value, which is used to convert the input data to the DL model. Considering the energy value of the sound at each frequency spaced by 1 (Hz), the limit frequency is 0/2 (kHz). *Tensor Input* is a vector of sound energy values in increasing order of frequency (Figure 5(a)). The values of Tensor Input after being created are usually very large. For the DL model to be better learned, the data level in the Tensor Inputs needs to be normalized by dividing all components by a certain value greater than the maximum value of the energy obtained. The Tensor Input for the DL model after normalization can be described in Figure 5(b).

2.1.2. Building the DL1 Model. After building the Tensor Inputs, the DL1 model is built with many inputs and many outputs (Figure 6) similar to the multilayer AI network in [27].

The number of inputs depends on the number of parameters in the Tensor Input vector. The output layer of network DL1 includes different nodes, and each of these nodes represents a certain word. The output words have the probability value of appearing in the range $[0, 1]$. The word with the highest probability value will be chosen as the result of the voice-to-text transition.

The layers hidden within the DL1 model determine the probability value of the words producing the correct output. The elements inside the *Tensor Input* and *Tensor Output* are scalar quantities, so the nonlinear activation function is used. According to [28], some nonlinear functions can be used such as *Sigmoid, Tanh,* and *Relu*, and the output layer is used with *Softmax* activation function to calculate the probability distribution across the classes. The DL model simulates how human biological neural work, so this needs to be trained to simulate the outputs with corresponding inputs and predict the results with other inputs. To train the DL model, the limiting criteria need to be defined and how it can be learned need to be outlined to distinguish between right and wrong. According to [29], the *Sparse Categorical Crossentropy (SCC)* function is used as follows: after each learning, the DL model needs to update these parameters again to create the actual output that converges gradually to the desired value or in other words, to make the error function value decrease with 0.

To update the DL model, [30] the ADAM optimization function is used to combine two momentum methods and RMSprop, whose learning rate changes with respect to time and can find the global minimum optimal value instead of the local minimum optimal value. Model DL1 is built through the *Tensorflow* library in PYTHON (Figure 7).

Line 47 declares the output layer with 17 nodes with the Softmax activation function. This output number represents 17 common words in the voice command framework. The *Softmax* activation function will calculate to give a sample with the highest probability to separate words and phrases from each other. A dictionary with words or phrases and the number of words that appear in the sentence is constructed and encoded as a vector. As such, network DL1 can ensure voice recognition, converting recognition data into text containing specific control information.

2.1.3. Extracting Control Information Using the Machine Learning Model. Technically, the Vietnamese sentence, after being separated into single words, will be classified according to the DL1 model to form a set of words that are necessary to combine into an equivalent complete text, free of noise and other redundant words. This complete text (meaningful Vietnamese words and phrases) is used as input to the machine learning (ML) model.

Actually, the algorithm TF-IDF is used to extract features of the text. Then, Naive Bayes algorithm is used to classify feature words and phrases of the text belonging to control information layer. The ML model is built in PYTHON language in combination with the math libraries *Sklearn* and *Pyvi*. The extracted information fields will be encoded numerically and transmitted to the manipulator control circuit via SERIAL communication. The output of the model is manipulator control information such as motion direction, robot's action, and object color.

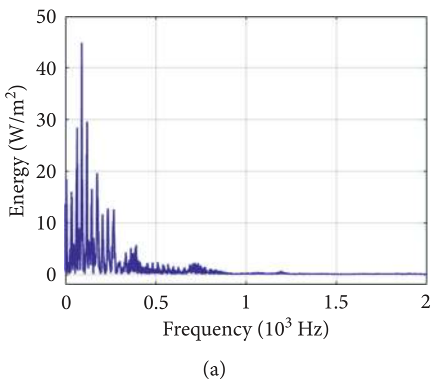
2.2. Inverse Kinematics Control for the Manipulator Using Deep Learning Network. The real 6DOF manipulator arm is presented in Figure 8 and its kinematics model is described in Figure 9.

In the kinematics model, the fixed global coordinate system is $(OXYZ)_0$. The local coordinate systems $(OXYZ)_i\,(i = 1/6)$ are placed on the joints accordingly. The joint variable i is denoted by q_i.

Let us denote $q = \begin{bmatrix} q_1 & q_2 & q_3 & q_4 & q_5 & q_6 \end{bmatrix}^T$ as the generalized coordinate vector of the 6 joint variables. The kinematics parameters of the 6DOF manipulator arm are determined according to DH rule [1], as given in Table 1.

Homogeneous transformation matrices $\mathbf{H}_i$, $(i = 1/6)$ on the six links are determined in [1] as the following general equation:

$$\mathbf{H}_i = \begin{bmatrix} \mathbf{R}_i^{i-1} & \mathbf{p}_i^{i-1} \\ 0 & 1 \end{bmatrix}, \quad (2)$$



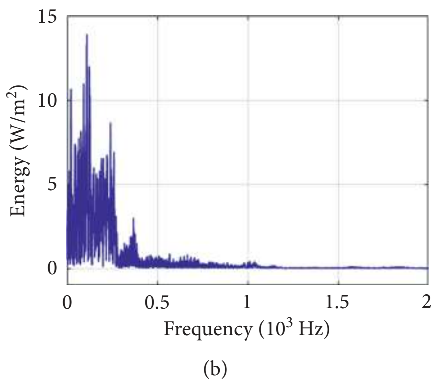


Figure 4: Sound energy of the word "Quay" and the word "Ph i".

30	45	20	10	13	26	18	11	8	...

(a)

0.3	0.45	0.2	0.1	0.13	0.26	0.18	0.11	0.08	...

(b)

Figure 5: Tensor Input before and after normalization.

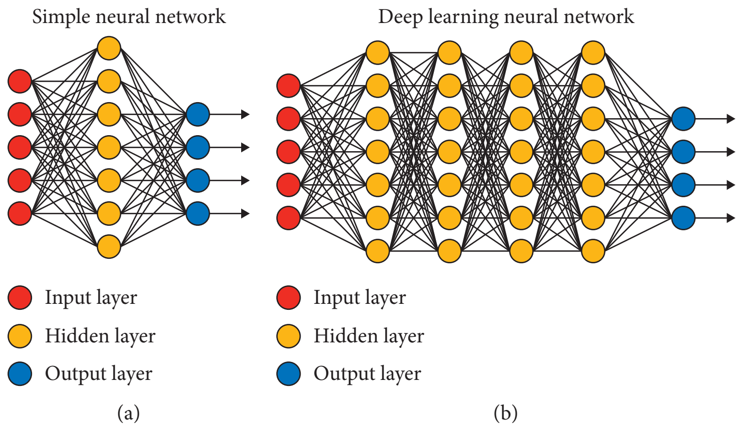
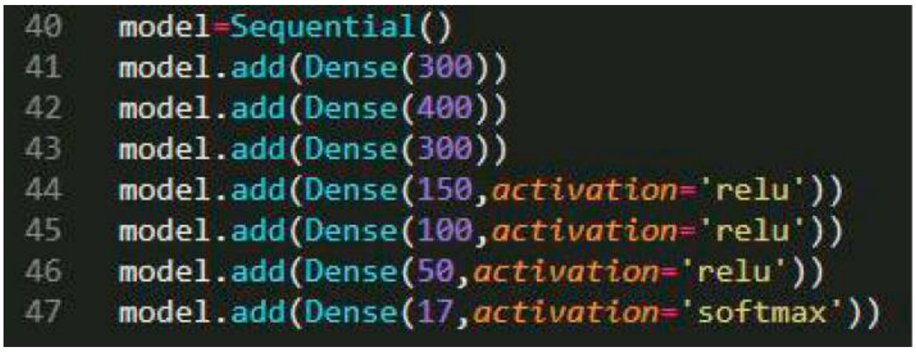


Figure 6: The multilayer artificial intelligence network [27]. (a) Simple neural network. (b) Deep learning neural network.



```
model=Sequential()
model.add(Dense(300))
model.add(Dense(400))
model.add(Dense(300))
model.add(Dense(150,activation='relu'))
model.add(Dense(100,activation='relu'))
model.add(Dense(50,activation='relu'))
model.add(Dense(17,activation='softmax'))
```

Figure 7: Model DL1 in PYTHON.

where $\mathbf{R}_i^{i-1}$ is a rotation matrix from the local coordinate system $(OXYZ)_{i-1}$ to the local coordinate system $(OXYZ)_i$ and $\mathbf{p}_i^{i-1}$ is a position vector of joint i on the coordinate system $(OXYZ)_{i-1}$.

The position and direction of the end-effector relative to the fixed global coordinate system are represented by the homogeneous transformation matrix $\mathbf{D}_6$. This matrix is calculated as follows:

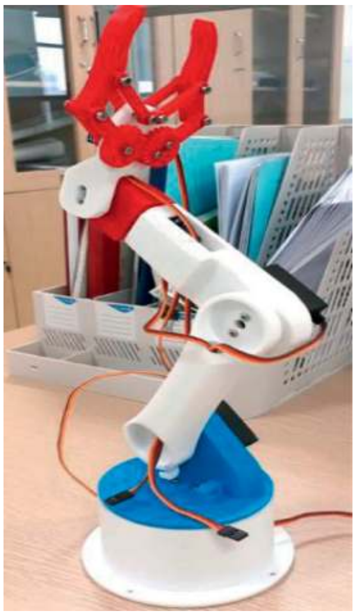


Figure 8: Real 6DOF manipulator arm.

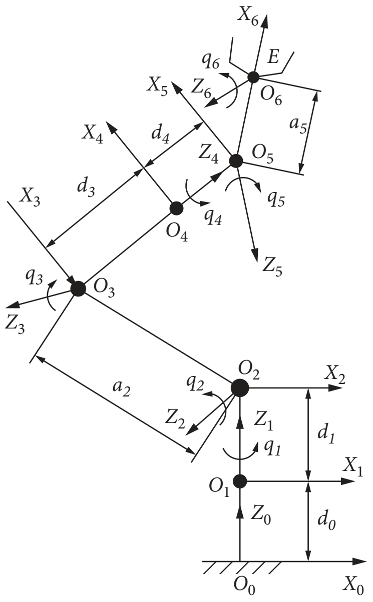


Figure 9: Kinematics model.

Table 1: DH parameters.

Parameters	θ_i^*	d_i	a_i	α_i
Link 1	q_1	$d_0 + d_1$	0	$\pi/2$
Link 2	q_2	0	a_2	0
Link 3	$q_3 - \pi/2$	d_3	0	$-(\pi/2)$
Link 4	q_4	d_4	0	$-(\pi/2)$
Link 5	q_5	0	a_5	0
Link 6	q_6	0	0	0

Note: ∗means variable. a_i is a length from the origin O_i to the origin O_{i+1} along the axis OX_{i+1}; α_i is an angle for rotating $(OXYZ)_i$ to $(OXYZ)_{i+1}$ around the axis OZ_{i+1}; d_i is a length from the origin O_{i-1} to the origin O_i along the axis OZ_i; and θ_i is an angle for rotating $(OXYZ)_{i-1}$ to $(OXYZ)_i$ around the axis OZ_i.

$$\mathbf{D}_6 = \mathbf{H}_1\mathbf{H}_2\mathbf{H}_3\mathbf{H}_4\mathbf{H}_5\mathbf{H}_6,$$
$$\mathbf{D}_6 = \begin{bmatrix} \mathbf{R}_E & \mathbf{p}_E \\ 0 & 1 \end{bmatrix}, \tag{3}$$

where $\mathbf{R}_\mathrm{E}$ is a direction matrix (3×3) for rotating global coordinate system $(OXYZ)_0$ to the local coordinate system of the end-effector $(OXYZ)_6$ and $\mathbf{p}_\mathrm{E} = [\,x_\mathrm{E} \;\; y_\mathrm{E} \;\; z_\mathrm{E}]^T$ is a position vector of the end-effector relative to the fixed global coordinate system $(OXYZ)_0$.

By applying the DH parameters into equations (2)–(4) and performing mathematical transformations (see the details in [1]), the position coordinates of the end-effector point are given as

$$\begin{aligned}
xE = {} & ((cq1 * cq2 * sq3 + cq1 * sq2 * cq3) * cq4 - sq1 * sq4) * a5 * cq5 + (-cq1 * cq2 * cq3 + cq1 * sq2 * sq3) * a5 * sq5 \\
& + (cq1 * cq2 * cq3 - cq1 * sq2 * sq3) * d4 + cq1 * cq2 * d3 * cq3 - cq1 * sq2 * d3 * sq3 + cq1 * a2 * cq2; \\
yE = {} & (sq1 * cq2 * sq3 + sq1 * sq2 * cq3) * cq4 + cq1 * sq4 * a5 * cq5 + (-sq1 * cq2 * cq3 + s1 * sq2 * sq3) * a5 * sq5 \\
& + (sq1 * cq2 * cq3 - sq1 * sq2 * sq3) * d4 + sq1 * cq2 * d3 * cq3 - sq1 * sq2 * d3 * sq3 + sq1 * a2 * cq2; \\
zE = {} & (sq2 * sq3 - cq2 * cq3) * cq4 * a5 * cq5 + (-sq2 * cq3 - cq2 * sq3) * a5 * sq5 + (sq2 * cq3 + cq2 * sq3) * d4 \\
& + sq2 * d3 * cq3 + cq2 * d3 * sq3 + a2 * sq2 + d1 + d0;
\end{aligned} \tag{4}$$

where cq_i stands for $\cos(q_i)$ and sq_i stands for $\sin(q_i)$.

The data for the network DL2 model are the spatial coordinate sets of the end-effector point and the corresponding set of joint variable parameters that have been collected and fed into the training DL2 network multiple times until the model can give control signals for the manipulator accurately, meeting the motion requirements. After training and assessing responsiveness well, the DL2 model is used as a model to predict manipulator rotation angle values with object positions in the manipulator workspace.

Figure 10 describes the entire process where the DL2 model is built with input as the request signal received after encoding the vector and feasible position data in the workspace. The output of the model is the corresponding joint variable values.

3. Experimental Results

The geometry parameters of the manipulator are as follows:

$$\begin{aligned}
d_0 &= 57\,\text{mm}, \\
d_1 &= 36\,\text{mm}, \\
a_2 &= 120\,\text{mm}, \\
d_3 &= 90\,\text{mm}, \\
d_4 &= 30\,\text{mm}, \\
a_5 &= 38\,\text{mm}.
\end{aligned} \tag{5}$$

The joint variable limits are as follows:

$$\begin{aligned}
-2.97 \le q_1 &\le 2.97\,(\text{rad}), \\
-0.52 \le q_2 &\le 2.7\,(\text{rad}), \\
-0.785 \le q_3 &\le 2.97\,(\text{rad}), \\
-2.7 \le q_4 &\le 2.7\,(\text{rad}), \\
-0.785 \le q_5 &\le 2.97\,(\text{rad}), \\
-3.57 \le q_6 &\le 3.57\,(\text{rad}).
\end{aligned} \tag{6}$$

The workspace of the manipulator arm is shown in Figure 11.

The drive motors are *Servo MG995*, Arduino Nano Circuit, Logitech B525-720p camera, Dell Precision M680 laptop, and Razer Seiren Mini microphone (Figure 12).

Network parameter DL2 controlling the manipulator is shown in Figure 13 with 5 outputs corresponding to 5 rotation angles of the manipulator joints. The network consists of 9 hidden layers with the *Relu* activation function. The number of nodes per layer is presented in Figure 13.

Training results and prediction results of motor control signals are shown in Figure 14. Check on the test data with input as the position vector of the end-effector point in the workspace is $x = \begin{bmatrix} 0 & 20 & 0 \end{bmatrix}^T$ (mm), and the output of the test data corresponds to the joint variable value. The q value obtained from the model is $q = \begin{bmatrix} 90 & 50 & 105 & 90 & 79 \end{bmatrix}^T$ (deg). Thus, the accuracy is 98.67% on the test dataset.

The actual experimental system with the circuit reading and writing the joint variable values and the feedback values on the 16×2 LCD is shown in Figure 15.

The joint variable values to control the manipulator arm to the position on the object (a yellow wheel) are shown in Figure 16.

4. Discussion

In actual operation, industrial robots in general and redundant manipulators in particular often perform not as perfectly as calculated in ideal conditions due to the influence of many different factors called noise that create the imperfect robot control system. According to [31], although imperfections are unavoidable in real production processes, the real devices still operate well in regimes far from ideality.

For example, mechanical imperfections may occur prior to operation due to mechanical manufacturing defects, assembly errors, or during operation due to mechanical system vibrations. Meanwhile, electrical imperfections can be caused by the electromagnetic interference of the surrounding environment, the instability of the power supply, or high-intensity electric pulses of welding machines. To overcome the imperfections, additional modules related to noise compensation, noise cancellation, or noise suppression will be studied in the next research stages.

This study only considers the problem of kinematics in ideal conditions or the impact of noise can be ignored. In fact, it is not possible to have a general anti-interference problem for all types of noise. Therefore, when practically applied, the research team will apply anti-interference solutions suitable for each context.

In the case of group coordination between multiple voice-controlled robots in a narrow space, naming or coding for each robot needs to be done through an independent module with name recognition or decoding capabilities.

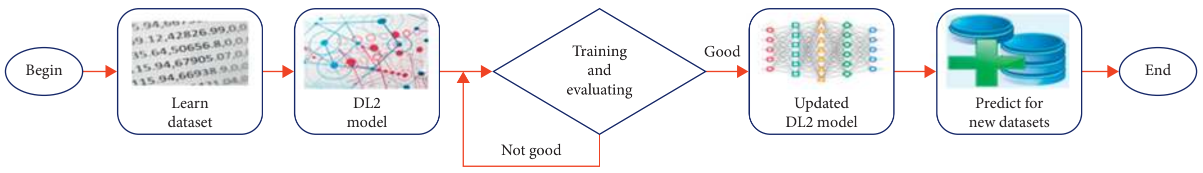


FIGURE 10: The building process for the DL2 model.

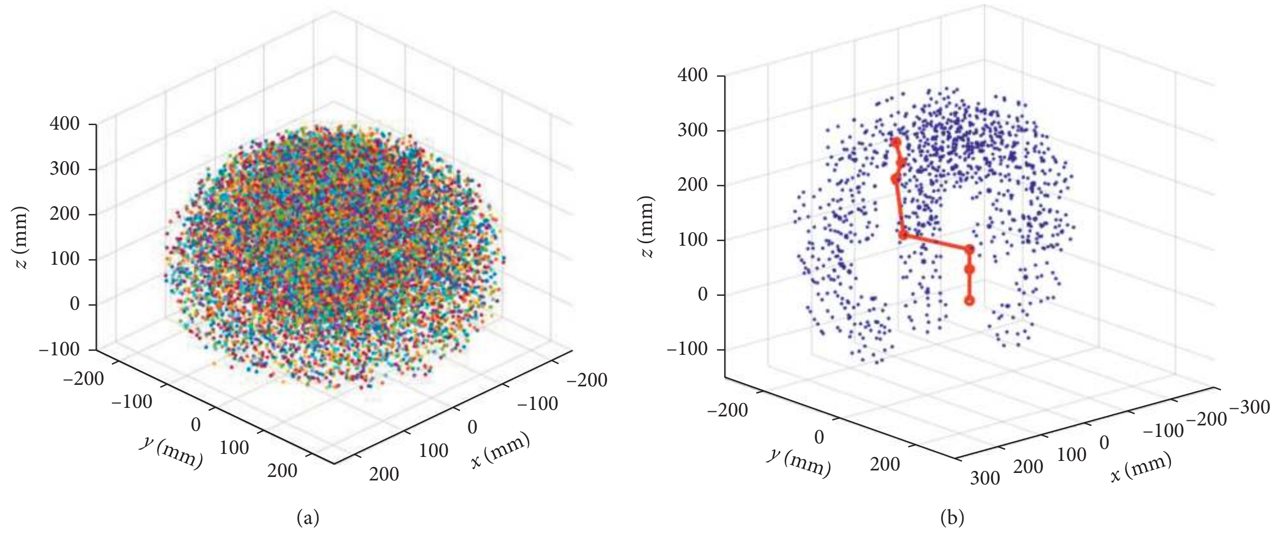


FIGURE 11: The workspace of 6DOF manipulator.

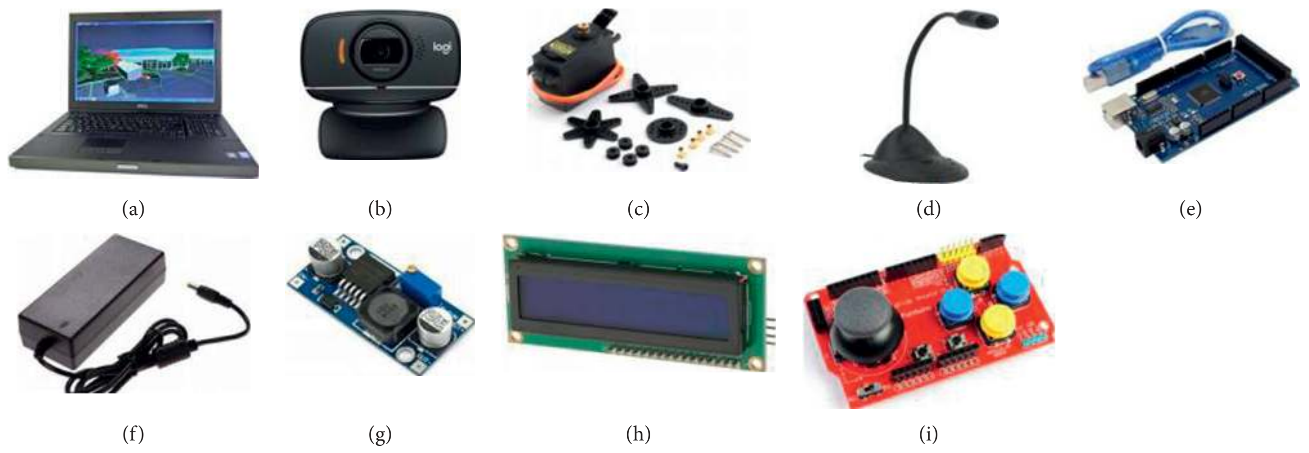
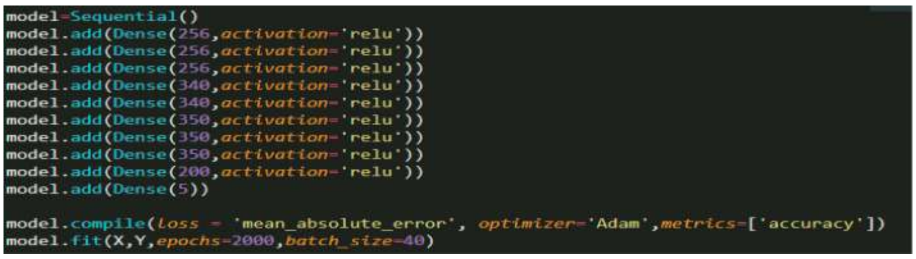


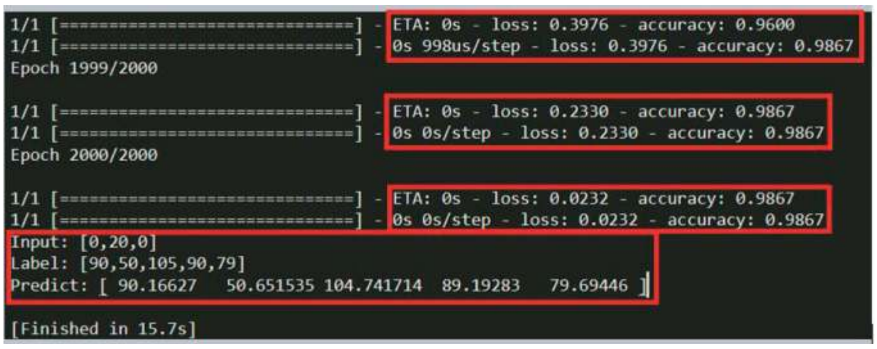
FIGURE 12: Devices in the experimental system. (a) Dell Precision laptop, (b) Logitech camera, (c) RC Servo MG995, (d) micro, (e) Arduino Mega 2560, (f) 12 V-5 A adapter, (g) XL4015 5A DC, (h) 16 × 2 LCD, and (i) joystick shield.



```
model=Sequential()
model.add(Dense(256,activation='relu'))
model.add(Dense(256,activation='relu'))
model.add(Dense(256,activation='relu'))
model.add(Dense(340,activation='relu'))
model.add(Dense(340,activation='relu'))
model.add(Dense(350,activation='relu'))
model.add(Dense(350,activation='relu'))
model.add(Dense(350,activation='relu'))
model.add(Dense(200,activation='relu'))
model.add(Dense(5))

model.compile(loss = 'mean_absolute_error', optimizer='Adam',metrics=['accuracy'])
model.fit(X,Y,epochs=2000,batch_size=40)
```

FIGURE 13: Network parameter DL2 for IK control.



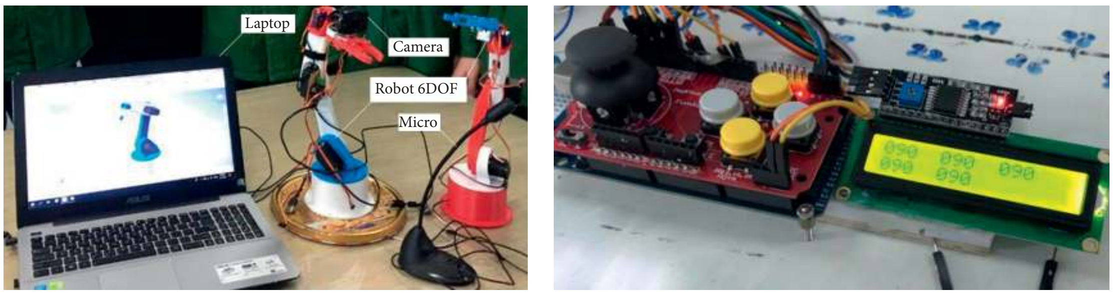
```
1/1 [==============================] - ETA: 0s - loss: 0.3976 - accuracy: 0.9600
1/1 [==============================] - 0s 998us/step - loss: 0.3976 - accuracy: 0.9867
Epoch 1999/2000

1/1 [==============================] - ETA: 0s - loss: 0.2330 - accuracy: 0.9867
1/1 [==============================] - 0s 0s/step - loss: 0.2330 - accuracy: 0.9867
Epoch 2000/2000

1/1 [==============================] - ETA: 0s - loss: 0.0232 - accuracy: 0.9867
1/1 [==============================] - 0s 0s/step - loss: 0.0232 - accuracy: 0.9867
Input: [0,20,0]
Label: [90,50,105,90,79]
Predict: [ 90.16627   50.651535 104.741714  89.19283   79.69446 ]

[Finished in 15.7s]
```

FIGURE 14: Training results and prediction on the test dataset.



(a) (b)

FIGURE 15: The actual experimental system.

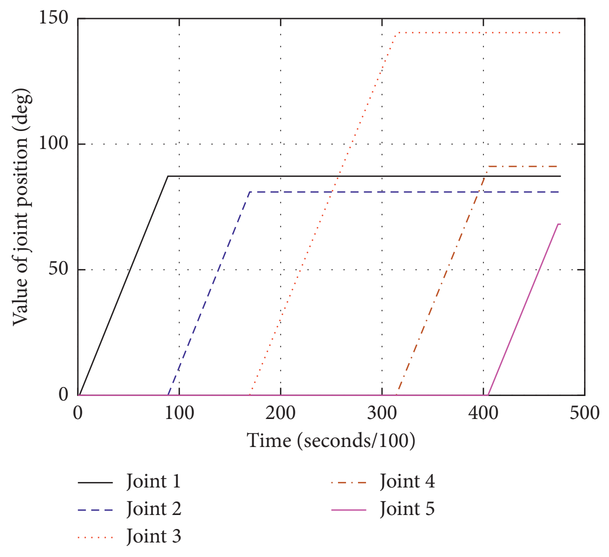


FIGURE 16: The joint values are received by the voice command.

When the operator calls the robot's name or activates the code, the related robot is ready to receive the next voice commands. Thus, when it is necessary to add a new robot to an existing robot network, it is possible to adjust the module of name recognition or decoding without any change in the entire control system.

Differently, in a robot network, the audio imperfections may come from the voice interference. The audio imperfections can be solved by the effect of different range connections controlled by a central dispatcher and the voice interference "can be improved by including long-range connections between the robots" [32].

5. Conclusion

In summary, the PYTHON language has been applied to build AI models for the Vietnamese voice recognition module and IK control for the 6DOF redundant manipulator. DL and ML techniques have been applied successfully with over 98% training accuracy. Data used for training models DL1 and DL2 are independently built according to the Vietnamese language and calculated data from 6DOF manipulator kinematics modeling. AI models are tested on real manipulator models and gave possible results. This study could serve as the foundation for developing applications for various types of manipulators (serial manipulators, parallel manipulators, hybrid manipulators, and mobile manipulators) for industrial production (welding robots, robot 3D printing, and machining robots), medical, service industries, home activities (surgical robots, flexible robots, soft robots, humanoid robots, UAVs, service robots in families, and restaurants).

Data Availability

The datasets generated during the current study are available from the corresponding authors on reasonable request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*, First edition, 2001.

[2] D. X. Bien, "On the effect of the end-effector point trajectory on the joint jerk of the redundant manipulators," *Journal of Applied and Computational Mechanics*, vol. 20, no. 10, pp. 1–8, 2021.

[3] C. A. My, D. X. Bien, H. B. Tung, L. C. Hieu, N. V. Cong, and T. V. Hieu, "Inverse kinematic control algorithm for a welding robot positioner system to trace a 3D complex curve," in *Proceedings of the International Conference on Advanced Technologies for Communications (ATC)*, pp. 319–323, Hanoi, Vietnam, October 2019.

[4] S. Lian, Y. Han, Y. Wang et al., "Accelerating inverse kinematics for high-DOF robots," in *Proceedings of the 54th Annual Design Automation Conference*, Austin, TX, USA, 2017.

[5] https://www.edureka.co/blog/artificial-intelligence-algorithms/.

[6] https://www.geeksforgeeks.org/top-5-best-programming-languages-for-artificial-intelligence-field/.

[7] https://www.cuelogic.com/blog/role-of-python-in-artificial-intelligence.

[8] D. P. Mital and G. W. Leng, "A voice-activated robot with artificial intelligence," *Robotics and Autonomous Systems*, vol. 4, no. 4, pp. 339–344, 1989.

[9] S. Hwang, Y. Park, and Y.-s. Park, "Sound direction estimation using an artificial ear for robots," *Robotics and Autonomous Systems*, vol. 59, no. 3-4, pp. 208–217, 2011.

[10] A. Rogowski, "Industrially oriented voice control system," *Robotics and Computer-Integrated Manufacturing*, vol. 28, no. 3, pp. 303–315, 2012.

[11] V. Alvarez-Santos, R. Iglesias, X. M. Pardo, C. V. Regueiro, and A. Canedo-Rodriguez, "Gesture-based interaction with voice feedback for a tour-guide robot," *Journal of Visual Communication and Image Representation*, vol. 25, no. 2, pp. 499–509, 2014.

[12] S. S. Turakne and P. Loni, "Intelligent interactive robot with gesture recognition and voice feedback," *International Journal of Engineering Research & Technology*, vol. 5, no. 4, pp. 276–280, 2016.

[13] M. Meghana, Ch. U. Kumari, J. S. Priya et al., "Hand gesture recognition and voice controlled robot," *Materials Today: Proceedings*, vol. 33, no. 7, pp. 4121–4123, 2020.

[14] M. F. Rafael and D. S. Manuel, "Design in robotics based in the voice of the customer of household robots," *Robotics and Autonomous Systems*, vol. 79, pp. 99–107, 2016.

[15] M. Buyukyilmaz and A. O. Cibikdiken, "Voice gender recognition using deep learning," *Advances in Computer Science Research*, vol. 58, pp. 409–411, 2017.

[16] K. Gundogdu, S. Bayrakdar, and I. Yucedag, "Developing and modeling of voice control system for prosthetic robot arm in medical systems," *Journal of King Saud University-Computer and Information Sciences*, vol. 30, no. 2, pp. 198–205, 2018.

[17] V. P. Saradi and P. Kailasapathi, "Voice-based motion control of a robotic vehicle through visible light communication," *Computers & Electrical Engineering*, vol. 76, pp. 154–167, 2019.

[18] S. Sachdev, J. Macwan, C. Patel, and N. Doshi, "Voice-controlled autonomous vehicle using IoT," *Procedia Computer Science*, vol. 160, pp. 712–717, 2019.

[19] A. T. Hasan, A. M. S. Hamouda, N. Ismail, and H. M. A. A. Al-Assadi, "An adaptive-learning algorithm to solve the inverse kinematics problem of a 6 D.O.F serial robot manipulator," *Advances in Engineering Software*, vol. 37, no. 7, pp. 432–438, 2006.

[20] Y. Zhou, W. Tang, and J. Zhang, "Algorithm for multi-joint redundant robot inverse kinematics based on the bayesian-BP neural network," in *Proceedings of the 2008 International Conference on Intelligent Computation Technology and Automation (ICICTA)*, pp. 173–178, Changsha, China, 2008.

[21] B. Daya, S. Khawandi, and M. Akoum, "Applying neural network architecture for inverse kinematics problem in robotics," *Journal of Software Engineering and Applications*, vol. 3, no. 3, pp. 230–239, 2010.

[22] A.-V. Duka, "Neural network based inverse kinematics solution for trajectory tracking of a robotic arm," *Procedia Technology*, vol. 12, pp. 20–27, 2014.

[23] R. Köker and T. Çakar, "A neuro-genetic-simulated annealing approach to the inverse kinematics solution of robots: a simulation based study," *Engineering with Computers*, vol. 32, no. 4, pp. 553–565, 2016.

[24] A. R. J. Almusawi, L. C. Dülger, and S. Kapucu, "A new artificial neural network approach in solving inverse kinematics of robotic arm (denso VP6242)," *Computational Intelligence and Neuroscience*, vol. 2016, Article ID 5720163, 10 pages, 2016.

[25] P. M. Shailendrasingh and L. P. Pratap, "A real-time approach using feed forward neural network for a 5 DOF robot manipulator," in *Proceedings of the IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI-2017)*, pp. 1240–1245, Chennai, India, September 2017.

[26] A. Garzelli, L. Capobianco, and F. Nencini, "Fusion of multispectral and panchromatic images as an optimization problem," *Image Fusion Algorithms and Applications*, pp. 223–250, Academic Press, Cambridge, MA, USA, 2008.

[27] https://www.securityinfowatch.com/video-surveillance/video-analytics/article/21069937/deep-learning-to-the-rescue.

[28] https://www.programmersought.com/article/10025152444/.

[29] https://www.Tensorflow.org/api_docs/python/tf/keras/losses/sparse_categorical_crossentropy.

[30] https://www.programmersought.com/article/33553292079/.

[31] M. Bucolo, A. Buscarino, C. Famoso, L. Fortuna, and M. Frasca, "Control of imperfect dynamical systems," *Nonlinear Dynamics*, vol. 98, no. 4, pp. 2989–2999, 2019.

[32] A. Buscarino, L. Fortuna, M. Frasca, and A. Rizzo, "Dynamical network interactions in distributed control of robots," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 16, no. 1, Article ID 015116, 2006.