ĐIỀU KHIỂN CÂN BẰNG CON NÊM NGƯỢC DÙNG PHƯƠNG PHÁP NƠRON MỜ

BALANCING CONTROL OF INVERTED WEDGE SYSTEM USING NEURO FUZZY METHOD

[1]Nguyen Thanh Tan, [2]Nguyen Minh Tam, [2]Le Thi Thanh Hoang, [2]Nguyen Van Dong Hai
[1] Tra Vinh University, Vietnam
[2] Ho Chi Minh City University of Technology and Education Vietnam

Received 02/2/2018, Peer reviewed 02/4/2018, Accepted for publication 19/7/2018

ABSTRACT

In this article, the authors use three control algorithms on the inverted wedge, including of LQR, Fuzzy Control and neuro-fuzzy. Simulation results show that all methods can balance the inverted wedge. Besides, the authors build successfully the experimental inverted wedge model which is based on computer communication between Matlab software and DSP TMS320F28335 card. Experimental results show that neuro-fuzzy control method can stabilize inverted wedge in vertical position. The values of angle and position of load vibrate around the expected equilibrium position.

Keywords: *Balance, inverted wedge, fuzzy control, neuro fuzzy, LQR*

TÓM TẮT

Trong bài viết này, tác giả đã sử dụng ba phương pháp điều khiển trên hệ con nêm ngược là: phương pháp LQR (Linear Quadratic Regulator), phương pháp điều khiển mờ và phương pháp nơron-mờ (ANFIS). Kết quả mô phỏng cho thấy cả ba phương pháp điều khiển trên đều có khả năng cân bằng ổn định hệ con nêm ngược. Bên cạnh đó, tác giả đã xây dựng thành công mô hình thực nghiệm hệ con nêm ngược thông qua giao tiếp máy tính giữa phần mềm Matlab với card DSP TMS320F28335. Kết quả thực nghiệm cho thấy phương pháp điều khiển nơron mờ hoàn toàn có thể điều khiển cân bằng hệ con nêm ngược theo phương thẳng đứng. Giá trị góc nghiêng và vị trí vật nặng thu được luôn dao động xung quanh vị trí cân bằng mong muốn.

Từ khóa: *Cân bằng, con nêm ngược, điều khiển mờ, nơron mờ, LQR.*

1. INTRODUCTION

There are many methods to control nonlinear systems, such as linearity method, sliding mode control (SMC), artificial neural network control, fuzzy control, adaptive control or many optimization algorithms (GA: Genetic Algorithm, PSO: Particle Swarm Optimzation). The selection of a suitable control method for a certain nonlinear object requires a lot of time and experiments.

In [1], the authors used Sliding Mode-Fuzzy-PID controller to balance the inverted wedge system. The model balancing controller designing is based on control the weight moving on the sliding plane. The simulation result shows that the stable balancing within the limit angle of $\pm \pi/2$ and the response time of about 2,5 seconds. In [3], Neuro-Sliding Mode controller is used. In a simulation, the system is stable at 0.55 seconds and the angle deviation is about 5^0. In [4], the authors used the fuzzy control and sliding mode control to balance the inverted wedge. They also used a duty belt to move the load weight across the horizontal plane. The simulation results show that the system

is stable in equilibrium position within a response time of about 7 seconds. In [5], the authors used the belt to drag a load weight at the two sides of the inverted wedge to balance the system. The main control method of the research was to design the LQR and the SMC controllers to the stabillize system. The settling time is only 2.5 seconds with a small angle deviation.

The idea of applying modern control methods to actual object control approaches to some references on the inverted wedge system-a nonlinear system. This model describes the balanced model of the ship in the maritime field. The system used the weight of the load through the traction of the motor to balance the center of the object. Because the inverted wedge is a complex nonlinear system, it is difficult to accurately determine the mathematical model. Besides that, the parameters of the system must be absolutely accurate. However, it is important to design a suitable controller for controlling the balance system and this is also the main objective of this paper.

2. MODEL AND CONTROLLER

2.1 Model of inverted wedge

Inverted wedge modeling is shown in figure 1 below

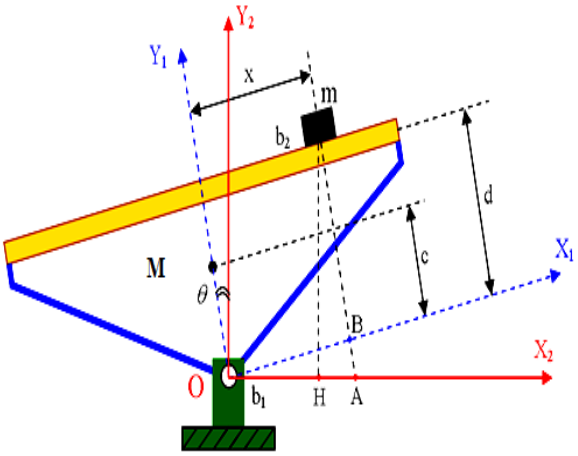

Figure 1. *Inverted wedge modeling [1]*

The symbols and the simulation parameters of system are listed in Table 1.

Table 1. *Parameters simulation of the inverted wedge system [1]*

Symbol	Explanation	Value
M	Weight of inverted wedge	3,0 kg
m	Weight of loads	0,65 kg
c	Distance between rotary axis and center	0,06 m
d	Distance between rotary axis and slide surface	0,12 m
b1	Coefficient of friction on rotary axis	0,4 N/m/s
b2	Coefficient of friction on sliding surface	10 N/m/s
g	Acceleration of gravity	9,81 m/s^2
Km	Induction coefficient of motor	5 Nm/A

Applying the Euler-Lagrange method, we obtain

$$\frac{d}{dt}\left(\frac{\partial K}{\partial \dot{q}_i}\right)-\frac{\partial K}{\partial q_i}+\frac{\partial P}{\partial q_i}=T_i \ (i=1,2) \qquad (1)$$

Where: K is the total kinetic energy; P is total potential; T is the momentum; $q = \begin{bmatrix} q_1 & q_2 \end{bmatrix}^T$ is the state variables.

For the inverted wedge system, we have $q_1=\theta$; $q_2=x$. Define the nonlinear equation of the inverted wedge system as follows:

$$\begin{cases} \ddot{\theta}=\dfrac{\left[\begin{matrix}-b_1\dot{\theta}-2mx\dot{x}\dot{\theta}-b_2d\dot{x}+\\ +mdx\dot{\theta}^2-mgxcos\theta+Mgcsin\theta\end{matrix}\right]}{Mc^2+mx^2}+ \\ +\left(\dfrac{2d}{Mc^2+mx^2}\right)F \\ \ddot{x}=\dfrac{\left(\begin{matrix}-b_1d\dot{\theta}-2mx\dot{x}\dot{\theta}-b_2d^2\dot{x}+\\ +md^2x\dot{\theta}^2-mgdxcos\theta+\\ +Mgdcsin\theta\end{matrix}\right)}{Mc^2+mx^2}-\dfrac{b_2\dot{x}}{m}+x\dot{\theta}^2+ \\ -gsin\theta+\left(\dfrac{2d^2}{Mc^2+mx^2}+\dfrac{1}{m}\right)F \end{cases} \qquad (2)$$

then, we obtain the linearization equations ($F=K_mU$ is the force of motor) as below:

$$\begin{bmatrix}\dot{\theta}\\ \ddot{\theta}\\ \dot{x}\\ \ddot{x}\end{bmatrix}=\begin{bmatrix}0 & 1 & 0 & 0\\ \frac{g}{c} & \frac{-b_1}{Mc^2} & \frac{-mg}{Mc^2} & \frac{-b_2 d}{Mc^2}\\ 0 & 0 & 0 & 1\\ \frac{gd}{c}-g & \frac{-b_1 d}{Mc^2} & \frac{-mgd}{Mc^2} & -\left(\frac{b_2 d^2}{Mc^2}+\frac{b_2}{m}\right)\end{bmatrix}\begin{bmatrix}\theta\\ \dot{\theta}\\ x\\ \dot{x}\end{bmatrix}+ \\ +\begin{bmatrix}0\\ K_m\left(\frac{2d}{Mc^2}\right)\\ 0\\ K_m\left(\frac{2d^2}{Mc^2}+\frac{1}{m}\right)\end{bmatrix}U \quad (3)$$

2.2 LQR method

LQR is built on the principle of state feedback. The controller receives the input signals as the state of the system, this values will calculate and converted into control signals for the system.

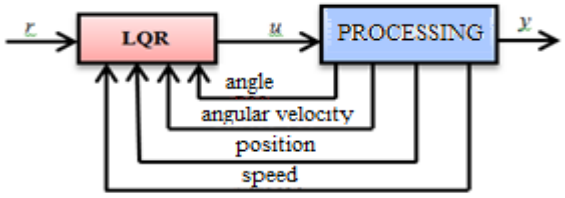

Figure 2. *Structure of LQR controller*

Consider that the linear system is

$$\dot{x}=Ax+Bu \quad (4)$$

then, the matrix K_u matrix of the optimal vector control is selected to obtain control signal as below:

$$u(t)=-K_u x(t) \quad (5)$$

and function J below to reach minimum

$$J=\int_0^{\infty}\left(x^T Qx+u^T Ru\right)dt \quad (6)$$

Where: Q is the positive determinant matrix (or semi-positive), R is the positive determinant matrix.

The K_u matrix is defined by the Riccati equation as in equation (7):

$$K_u=R^{-1}B^T P \quad (7)$$

Substituting (7) into (5), we have the law controller as follows:

$$u(t)=-K_u x(t)=R^{-1}B^T Px(t) \quad (8)$$

So, the matrix P must be based on the Riccati equation:

$$PA+A^T P+Q-PBR^{-1}B^T P=\dot{P} \quad (9)$$

When P does not change follow the time ($\dot{P}=0$), we get Riccati algebra as follows:

$$PA+A^T P+Q-PBR^{-1}B^T P=0 \quad (10)$$

Considering that equilibrium point of system is at ($\theta=0,\dot{\theta}=0,x=0,\dot{x}=0$ and u = 0), basing on the Matlab software, we find the state matrix A, B and select the matrix Q, R in the target function J as follows:

$$A=\begin{bmatrix}0 & 1 & 0 & 0\\ 196.2 & -6.67 & -784.8 & -720\\ 0 & 0 & 0 & 1\\ 13.734 & -0.8 & -94.176 & -161.4\end{bmatrix};B=\begin{bmatrix}0\\ 160\\ 0\\ 27.533\end{bmatrix};$$

$$Q=\begin{bmatrix}100 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix};R=10$$

The optimal K_u matrix is:

K_u = [15.797 1.373 -55.214 -4.295]

2.3 The direct fuzzy controller

Schematic diagram of the direct fuzzy controller:

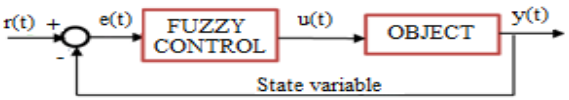

Figure 3*. The direct fuzzy controller structure*

The direct fuzzy controller is designed as a MISO (Multi Input Single Output) controller with 4 inputs: angle, angle velocity,

the position of load, the speed of the load and one output: the motor voltage. The limitation of each variable is:

- Angle deviation θ(t): [-pi/2,pi/2] (rad)
- Angle velocity $\dot{\theta}(t)$: [-1,1] (rad/s)
- Position x(t): [-0.3,0.3] (m)
- Speed of the loads $\dot{x}(t)$: [-1,1] (m/s)
- Voltage supplied to the motor DC:[-24,24] (V)

The fuzzy rule is based on the experience and knowledge on the operation of the inverted wedge system. We choose the number of the fuzzy set for every input is 3, so we have the number of a fuzzy rule is 3^4 = 81 and the fuzzy set output is 7. The membership functions are as shown in Fig.4.

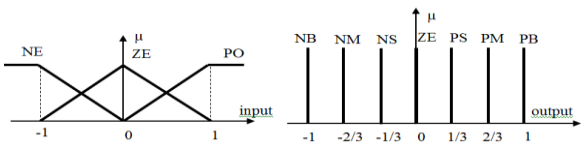

***Figure 4**. Membership functions of inputs/outputs*

The input/output language variable and the membership functions are defined as follows: {NE (negative), ZE (zero) and PO (positive)} and {NB (large negative), NM (medium negative), NS (small negative), ZE (zero), PS (small positive), PM (medium positive), PB (large positive). We construct the fuzzy control rules by:

- Fuzzy encode by MAX – MIN method.
- Fuzzy decode by the weighted method.

2.4 Neural-Fuzzy controller [2]

ANFIS (Adaptive Neuro Fuzzy Inference System) is an adaptive neural network based on the fuzzy inference system. The fuzzy neural network consists of 5 layers, 2 inputs and 1 output as the diagram (5). However, we could design many different structures of the neural network.

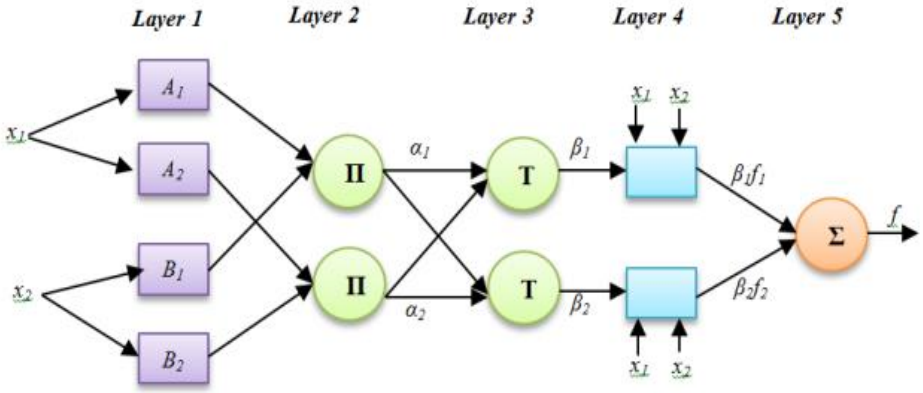

***Figure 5.** The Fuzzy neural network structure with 2 inputs and 1 output*

- The 1st Layer: As the input layer, every ith neuron has the corresponding input signal. The output of the nodes are the membership functions that change with fuzzy sets.
- The 2nd Layer: Each node is the output value of the fuzzy rule. They are labeled as Π, because it may be having many different t-norm operations for a node. The value of the output coefficients of the law is calculated as follows:

$$\alpha_1 = \min\ A_1, B_1\ ; \alpha_2 = \min\ A_2, B_2 \qquad (11)$$

- The 3rd Layer: The ith node is calculated as the ratio of the output coefficient of the rule at that node on the sum of the previous coefficients.

$$\beta_1 = \frac{\alpha_1}{\alpha_1 + \alpha_2}; \beta_2 = \frac{\alpha_2}{\alpha_1 + \alpha_2} \quad (12)$$

- The 4th Layer: The value of each neuron is calculated as follows:

$$f_i = \beta_i(\mathrm{P}_0 + \sum_{i=1}^{2} P_i x_i); \mathrm{i} = 1..2 \quad (13)$$

Therein, P_i is the adjustment parameters.

- The 5th Layer: The network output is the sum of the previous neuron output values.

$$f = \frac{\sum_{i=1}^{2} \alpha_i f_i}{\sum_{i=1}^{2} \alpha_i} = \sum_{i=1}^{2} \beta_i f_i; i = 1..2 \quad (14)$$

The designing steps of the fuzzy neural network controller are based on ANFIS toolbox in the Matlab on the inverted wedge system.

- Step 1: Developing a training data sets based on a fuzzy controller and performing some other data processing.
- Step 2: Loading the training data set and testing data set into the ANFIS interface.
- Step 3: Selecting the membership functions as the Gaussian function, the number of input variables is 4. Thence, we need to choose the number of membership functions for each input variable is 3.
- Step 4: Setting the training error and the number of iterations for the training process is 50, the training algorithm is backpropagation hybrid with the least square error.
- Step 5: Checking the training values after the training process is completed and save the fuzzy inference system of the trained.
- Step 6: Simulating the testing results.

The structure diagram of the neural network training and training data are shown in Figure 6 and Figure 7.

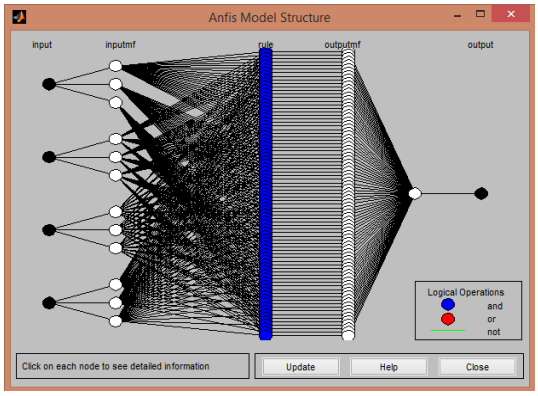

***Figure 6**. The structure diagram of the fuzzy-neural network after training*

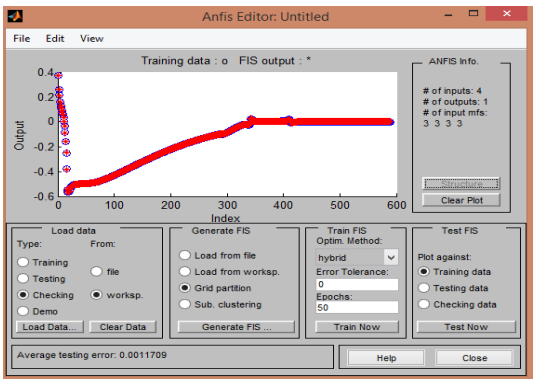

***Figure 7.** The data after training*

The neural network training structure diagram is shown in figure 6, the number of membership functions is 3. The data after training in figure 7 shows that the error training is $1{,}17.10^{-3}$ this value is small and acceptable.

3. SIMULATION AND EXPERIMENTAL RESULTS

3.1. Simulation results

LQR controller has the most simple structure (with only linear structure) compared to Fuzzy and fuzzy-neuro controllers. The complexity of fuzzy and fuzzy-neuro structure depends on decision of designer. If the designer knows features of the system well, they can choose more complicated memberships and bigger rule-table for fuzzy controller. With fuzzy-neuro structure, the designer can choose the more complicated network of ANFIS to make a controller to have a more ability to adapt the control situation. The more complex the controller has more ability to be better if we can choose

suitable control parameters. The disadvantages of a more complicated structure of controller are the big quantity of control parameters requires more time and effort to choose better values. Relatively, in this paper, we spend time to try each controller:

- In LQR control: the best controller is designed from most valuable weighing matrixes through trial and error test. After a period of time, the quality of a controller is improved very slowly. That is the limitation of simplicity of the linear structure of the controller.
- In fuzzy control: the input/output memberships are chosen as Figure 4. Then, trial and error test are used until the improvement of control quality meet saturation.
- In fuzzy-neuro control: the data of fuzzy control are used to train ANFIS structure. Due to more complicated structure of ANFIS, compared to fuzzy structure, the training can give better or worse results. Then, the best results after a period of trial and error testing in training are obtained. Actually, the purpose of ANFIS is to imitate another controller. Because of the more its complicated structure, a better control results, which do not depend on the ability of training but depend on mutation of learning, are obtained.

These processes give results that more complicated structure of controllers give better ability to obtain better control quality from trial and error test.

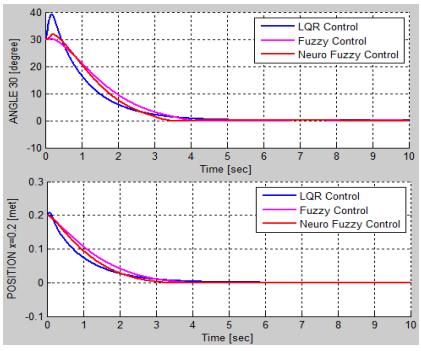

Figure 8. *With initial condition as $\theta = 30^0$, $x = 0.2m$*

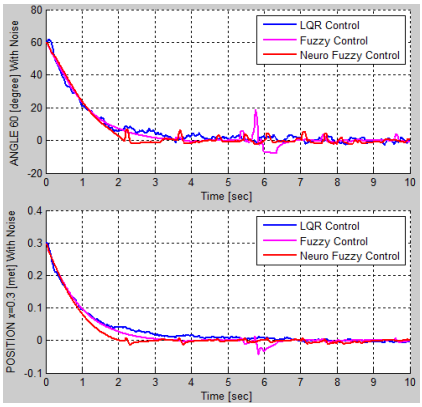

Figure 9. *With initial condition as $\theta = 60^0$, $x = 0.3m$ and noise signal*

Simulation results are shown in Fig. 8 and 9. Simulation results show that all three controllers can stabilize inverted wedge system. The fuzzy-neural control method produces the best result with large initial angle control, short setting time.

3.2. Experimental results

The Figure 10 is the block diagram of the experimental model on the inverted wedge system.

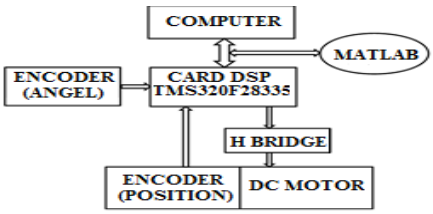

Figure 10. *The block diagram of experimental model*

The computer is capable of collecting and processing information collected from sensors through a DSP TMS320F28335 card. Control programs are programmed entirely on Matlab/Simulink and CCS v3.3. From the signals that are collected from the real-time embedded system, the computer will process and send the control output signal to the driver as a DC motor by generating PWM signals for H-bridge circuit. The feedback values of the angle and the position of the load are collected by two rotary encoders. The sampling time of real-time operation is 0.01 seconds. Fig. 11 is an inverted wedge model and experimental results obtained by the fuzzy neural control are shown in Fig. 12, 13 and 14.

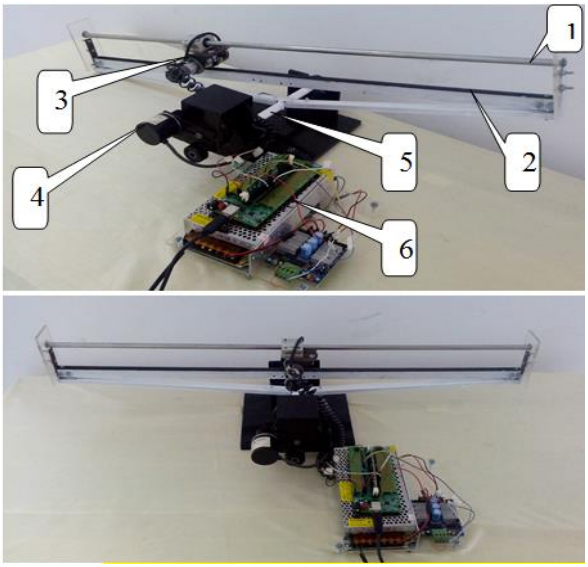

Figure 11. *The real-time inverted wedge model*

1: Slider;

2: Ray Slider;

3: Weight (motor and position sensor);

4: Angle sensor;

5: Rotating shaft;

6: DSP card and circuits

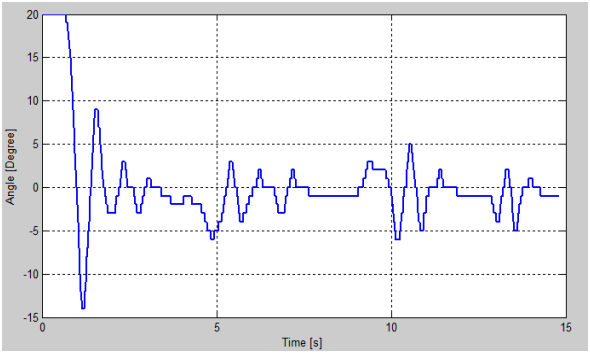

Figure 12. *Angle of wedge model (degree)*

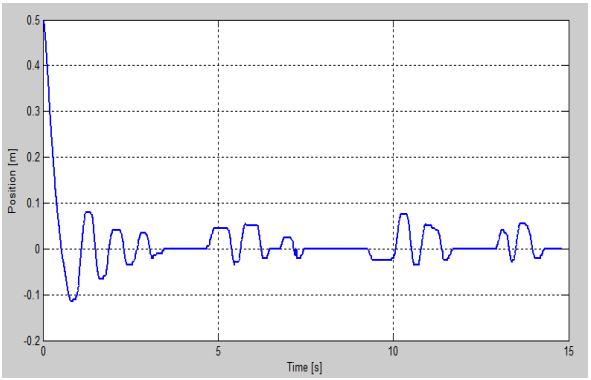

Figure 13. *Position of wedge model (m)*

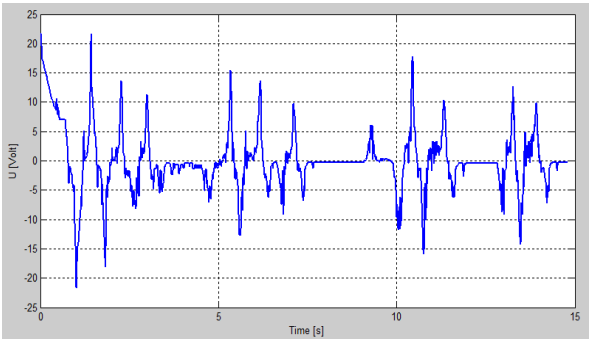

Figure 14. *The voltage for DC motors (V)*

4. CONCLUSION

Control results show that modern controllers in general nonlinear systems and the inverted wedge system have yielded satisfactory results. The fuzzy neuron method (using ANFIS) can stabilize the system vertically better than other controllers. Its explicit knowledge representation is easy to test, repair and capable of learning through datasets of the system.

REFERENCES

[1] Đặng Hữu Phúc, *Thiết kế, thi công và điều khiển mờ hệ con nêm ngược*, Luận văn tốt nghiệp Thạc sĩ, Đại học Giao Thông Vận Tải, TP. Hồ Chí Minh, 2012.

[2] Nguyễn Như Hiền, Lại Khắc Lãi, *Hệ mờ và nơron trong kỹ thuật điều khiển*, NXB Khoa học Tự nhiên và Công nghệ Hà Nội, 2007.

[3] Chun-Hsien Tsai, Hung-Yuan Chung, Fang-Ming Yu, *Neuro-Sliding Mode Control With Its Applications to Seesaw Systems*, IEEE Transactions on, pp. 124 – 134, 2004.

[4] Jeng-Hann Li, Tzuu-Hseng S. Li, Ting-Han Ou, *Design and Implementation of Fuzzy Sliding-Mode Controller for a Wedge Balancing System*, Journal of Intelligent and Robotic Systems, Volume 37 Issue 3, pp. 285-306, 2003.

[5] Shinq-Jen Wu, Cheng-Tao Wu, Yung-Yi chiou, Chin-Teng Lin, Yi-Nung Chung, *Balancing Control of Sliding Inverted-Wedge System: classical- method- based compensation*, IEEE International Conference on SMC '06, pp. 1349 – 1354, 2006.

Corresponding author:
Nguyen Thanh Tan
Tra Vinh University, Vietnam
Email: thanhtantvu@gmail.com