Speed and Position Estimation and DSP Implementation in Permanent Magnet Synchronous Motors (PMSM) Drives using Neural Networks

Tong Liu

Malik Elbuluk
Senior Member IEEE

Iqbal Husain
Member IEEE

Department of Electrical Engineering
The University of Akron
Akron, OH 44325-3904
Phone: (330)-972-7649
Fax: (330)-972-6487

Abstract

Rotor speed and position estimations in permanent magnet synchronous motor (PMSM) suffer from accuracy due to variation of the machine parameters such as torque constant, stator resistance and inductance, especially at low speeds. Also, due to nonlinearities, conventional linear estimators are not adaptive to the operation points.

This paper presents two neural network-based model reference adaptive systems (MRAS) for speed and position estimation in PMSM drives. The first network estimates the rotor speed and adapts on-line to any change in stator resistance with a rough rotor position sensor. The second network estimates the rotor position and speed without rotor position sensor. A q-axis model for stator inductance according to the current is used. In both cases, the model reference adaptive system adjusts the neural weights to give the optimal performance over a wide speed range.

The neural network estimators are simulated and implemented using a DSP TMSC320C30 (40 MHz) controller which utilizes a voltage vector control to produce PWM gate signals for an IGBT VSI inverter. Simulation and experimental results have shown that the neural estimation adjusts very well to changes in the motor operating points.

1.0 Introduction

In a PMSM, the torque constant (or rotor flux linkage), stator resistance and stator inductance are time varying parameters. Accurate acquisition of these parameters is necessary for deadbeat controller, speed and position estimation in sensorless algorithms. Since a high-resolution position sensor is one of research targets on PMSM drive, high quality of position and speed estimation became a trend. It has been shown in [1] and [4], for low-resolution encoder or sensorless operation, that the speed and/or position estimations are sensitive to parameter variation. In [2], [3], [4] and [5], the rotor positions were extracted from the back-emf estimation via current observers. The observer gain design in those papers was tedious and the adaptive observer pole design [4] over a wide speed range is obtained using trial and error method.

This paper presents neural network estimators of rotor speed and/or position with on-line parameter adaptation in PMSM. Two neural network estimators are proposed. The first uses a crude position sensor. The rotor speed is estimated even at low speed, with adaptation of the stator resistance. The q-axis inductance is modeled off-line according to q-axis current. A constant magnetizing current is injected into the phase windings, and low-pass filters and limiters are used to facilitate implementations of the estimation. The effect of mismatch of torque constant on the speed estimation has been studied. The second network estimates the rotor position and speed in a sensorless algorithm.

The neural network estimators are able to track the varying parameters, speed and position at different speeds with consistent performance. Compared to other methods, they are adaptive to operation conditions and easy in design. The simulation and experimental results justifying the claim are presented.

2.0 PMSM Model

A PMSM can be conventionally modeled in the stationary (αβ) and synchronous (dq) reference frames as shown in the Fig. 1. The abc and αβ reference frames are fixed in stator. Abc windings have 120° spatial angle differences. The α-axis is in line with the a-phase and β-axis leads α-axis by 90° of spatial angle. The dq reference frame is locked with the rotor. The d-axis is aligned to the magnet flux direction and q-axis lags the d-axis by 90° of spatial angle. The rotor position is used to regulate the stator currents so that the current frequency is always in synchronism with the rotor. The most commonly used dq-reference model is:

$$\frac{di_d}{dt} = -\frac{R}{L_d} i_d + \frac{L_q}{L_d} 0.5 p\omega \; i_q + \frac{v_d}{L_d} \qquad (1)$$

$$\frac{di_q}{dt} = -\frac{R}{L_q} i_q - \frac{L_d}{L_q} 0.5 p\omega \; i_d - 0.5 p\omega k_e + \frac{v_q}{L_q}$$

$$\frac{d\omega}{dt} = 1.5 * \frac{k_e}{J} i_q + 1.5 * \frac{(L_d - L_q)}{J} i_d i_q - \frac{T_l}{J} - \frac{B}{J}\omega - \frac{F}{J}\frac{\omega}{|\omega|}$$

$$\frac{d\theta}{dt} = 0.5 p\omega$$

where i_d, i_q are the dq axes stator currents, v_d, v_q are the dq axes stator voltages, R is the stator phase resistance, L_d, L_q are the dq axes phase inductances, J is

the rotor and load inertia, F is the coulomb friction, B is the viscous load, T_l is the load torque, p is the number of poles, ω is the rotor speed, θ is the rotor position and k_e is the torque constant.

The rotor speed state equation shows that if the d-axis current is controlled to be a constant, the generated torque is proportional to q-axis current. Therefore, for the motor torque, the q-axis current is the only control variable. This is similar to the torque control technique used in DC machines.

The αβ model is given below.

$$\frac{di_\alpha}{dt} = -\frac{R}{L} i_\alpha + \frac{k_e}{L} 0.5 p\omega \sin\theta + \frac{v_\alpha}{L} \qquad (2)$$

$$\frac{di_\beta}{dt} = -\frac{R}{L} i_\beta - \frac{k_e}{L} 0.5 p\omega \cos\theta + \frac{v_\beta}{L}$$

$$\frac{d\omega}{dt} = -1.5 * \frac{k_e}{J} i_\alpha \sin\theta + 1.5 * \frac{k_e}{J} i_\beta \cos\theta - \frac{T_l}{J} - \frac{B}{J}\omega - \frac{F}{J}\frac{\omega}{|\omega|}$$

$$\frac{d\theta}{dt} = 0.5 p\omega$$

where v_α and v_β are the $\alpha\beta$ stator voltages, i_α and i_β are the $\alpha\beta$ stator currents and L is the phase inductance $L \approx 0.5 * (L_d + L_q)$.

The current dynamics involves the position information and can be used for position estimaton.

3.0 Rotor Speed and Stator Resistance Estimation with A Crude Position Sensor

The speed estimator is shown in Fig.2. A parallel model reference adaptive system (MRAS) is used. The procedure is in 3 steps. First, a model of q-axis stator inductance presented in [7] is used and the d axis current is kept constant, and the variation of d-axis inductance is negligible. Secondly, the model of PMSM is used as an adjustable model in parallel with the actual motor used as reference model. Thirdly, the errors between reference and adjustable model currents are inputs to a feed-forward neural network for the adaptation algorithm to adjust for changes in the rotor speeds and stator resistance.

The PMSM synchronous d-q reference frame model is

$$i_d(k+1) = i_d(k) + \frac{Ts}{L_d} * (v_d(k) + L_d 0.5 p\omega(k) i_q(k) - R i_d(k)) \qquad (3)$$

$$i_q(k+1) = i_q(k) + \frac{Ts}{L_q} * (v_q(k) - L_q 0.5 p\omega(k) i_d(k) - k_e 0.5 p\omega(k) - R i_q(k))$$

where Ts is the sampling period, $i_d(k+1)$ $i_q(k+1)$ are the dq axis stator currents, $v_d(k), v_q(k)$ are the dq axis stator voltages, R is stator resistance per phase, L_d, L_q are the dq axis phase inductance, k_e is the torque constant and ω is the rotor speed.

The adjustable model has a similar formula with the estimated rotor speed $\hat{\omega}$, estimated resistance $\hat{R}$ and torque constant $\hat{k}_e$ replacing the real values. The current error dynamics are

$$e_{id}(k+1) = i_d(k+1) - \hat{i}_d(k+1) \approx (\hat{R} - R) i_d(k) \frac{Ts}{L_d} \qquad (4)$$

$$e_{iq}(k+1) = i_q(k+1) - \hat{i}_q(k+1) \approx \frac{0.5 pTs}{L_q} (\hat{k}_e \hat{\omega}(k) - k_e \omega(k)) + (\hat{R} - R) i_q(k) \frac{Ts}{L_q}$$

The error currents vanish when the speed and all the parameters are correctly estimated. In the real implementation, if the d-axis current is controlled to be zero, the d-axis current error will not be sufficient to give the correct estimation. To solve this problem, a constant magnetizing current is injected into the stator windings to facilitate the estimation. The undesirable effect is additional copper loss and lower efficiency. In this method, the mismatch of torque constant will cause some steady error in speed estimation.

To avoid the effect of current noise, low pass filters for the current errors are used. The filtered current errors and the changes of these filtered current errors are the inputs to the neural network. The outputs of the network are the changes of the estimated parameters.

The low pass current filters used are as follows:

$$e_{id}^{\Sigma} = \frac{1}{m} \sum_{n=0}^{m-1} z^{-n} e_{id}, \; e_{iq}^{\Sigma} = \frac{1}{m} \sum_{n=0}^{m-1} z^{-n} e_{iq} \qquad (5)$$

where m is the number of sub-cycles for the low pass filter, and z is the delay operator.

The other two inputs to the neural network are the changes of the current errors described below:

$$ce_{id}^{\Sigma} = (1 - z^{-1}) e_{id}^{\Sigma}, \; ce_{iq}^{\Sigma} = (1 - z^{-1}) e_{iq}^{\Sigma} \qquad (6)$$

The error index is defined as:

$$E = 0.5 * e_{id}{}^{\Sigma} * e_{id}{}^{\Sigma} + 0.5 * e_{iq}{}^{\Sigma} * e_{iq}{}^{\Sigma} \qquad (7)$$

The neural network weight adaptation is based on the least square method given by:

$$v_{ij}(k+1) = v_{ij}(k) - \gamma \frac{\partial E}{\partial v_{ij}} \qquad (8)$$

where E is the index of the filtered current errors, v_{ij} is one of the neural weights and γ is the learning rate. For example, v_{11} is the weight from the first input neuron to the first hidden neuron (Fig.3) and the derivative of the error index to the weight is:

$$\frac{\partial E}{\partial v_{11}} = \left(\frac{\partial E}{\partial e_{iq}{}^{\Sigma}} \frac{\partial e_{iq}{}^{\Sigma}}{\partial v_{11}} + \frac{\partial E}{\partial e_{id}{}^{\Sigma}} \frac{\partial e_{id}{}^{\Sigma}}{\partial v_{11}}\right) \qquad (9)$$

$$= \left(e_{iq}{}^{\Sigma} \frac{\partial e_{iq}{}^{\Sigma}}{\partial d\hat{\omega}} \frac{\partial d\hat{\omega}}{\partial v_{11}} + e_{id}{}^{\Sigma} \frac{\partial e_{id}{}^{\Sigma}}{\partial d\hat{R}} \frac{\partial d\hat{R}}{\partial v_{11}}\right)$$

$$= \left(e_{iq}{}^{\Sigma} \frac{\partial e_{iq}{}^{\Sigma}}{\partial d\hat{\omega}} \frac{\partial d\hat{\omega}}{\partial y_{in1}} \frac{\partial y_{in1}}{\partial z_1} \frac{\partial z_1}{\partial z_{in1}} \frac{\partial z_{in1}}{\partial v_{11}} + e_{id}{}^{\Sigma} \frac{\partial e_{id}{}^{\Sigma}}{\partial d\hat{R}} \frac{\partial d\hat{R}}{\partial y_{in2}} \frac{\partial y_{in2}}{\partial z_1} \frac{\partial z_1}{\partial z_{in1}} \frac{\partial z_{in1}}{\partial v_{11}}\right)$$

$$= \left(e_{iq}{}^{\Sigma} f'(y_{in1}) w_{11} f'(z_{in1}) x_1 + e_{id}{}^{\Sigma} f'(y_{in2}) w_{12} f'(z_{in1}) x_1\right)$$

where $\frac{\partial e_{iq}{}^{\Sigma}}{\partial d\hat{\omega}}, \frac{\partial e_{id}{}^{\Sigma}}{\partial d\hat{R}}$ are the Jacobean functions of the system. They can be calculated by $\frac{\partial d\hat{\omega}}{\partial e_{iq}{}^{\Sigma}}, \frac{\partial d\hat{R}}{\partial e_{id}{}^{\Sigma}}$

in the neural net. For simplicity, $\frac{\partial e_{iq}^{\Sigma}}{\partial d\hat{\omega}} = 1$ and $\frac{\partial e_{id}^{\Sigma}}{\partial d\hat{R}} = 1$ are used. $f'(y_{in1})$ is the derivative of the threshold nonlinear function, z_{in1} is the summing output of the first hidden neuron, z_1 is the output of the first hidden neuron, y_{in1}, y_{in2} are the summing outputs of the first and second output neurons, x_1 is the input to the first input neuron.

The adaptation law can finally be written as:

$$v_{11}(k+1) = v_{11}(k) - \gamma \frac{\partial E}{\partial v_{11}} = v_{11}(k) - \gamma\ ((e_{iq}^{\Sigma} f'(y_{in1}) w_{11} f'(z_{in1}) x_1 + e_{id}^{\Sigma} f'(y_{in2}) w_{12} f'(z_{in1}) x_1) \qquad (10)$$

3.1 Simulation Results of Rotor Speed and Stator Resistance Estimation

A 920 W, 3-phase 6-pole PMSM was used in the simulation to test the neural estimators. The other parameters are:

R =0.035 Ω, L_q =34 μH, L_d =34 μH, k_e =0.0094 N-m/A, rated torque T_N =2.9 N-m, rated speed ω_N =314 rad/s.

A standard feed-forward neural network is designed with 3 layers and a 4-4-2 neural structure (4 input neurons, 4 hidden neurons and 2 outputs). Back propagation algorithms is used to train the neural network with a learning rate at 0.1, a nonlinear threshold function of $f(x) = (1-e^{-x})/(1+e^{-x})$,

and initial neuron weights between -0.01 ~0.01. The output neuron gains for rotor speed, resistance and $\alpha\beta$ back-emf changes, are: $K_\omega = command\ \ speed/15$, $K_R = off-line\ R\ \ measurement/15$, and $K_E = dc\ link\ voltage/15$.

The limiter of estimated speed is [-5~5] times of the reference speed. The limiter of estimated back-emf is [-1~1] times of the dc link voltage. The switching frequency, the electrical control and estimation frequencies are 4.0 kHz and mechanical control frequency is 1.0 k Hz.

At first, the neural speed estimator is used without any parameter adaptation at 3.5 and 15 rad/s. To show the effect of detuning of R and k_e on the speed estimation. Fig.4 and Fig.10 are the accurate speed neural estimation with the accurate k_e and R. (Fig.4 at 3.5 rad/s, Fig.10 at 15 rad/s respectively.) When the k_e is 150% detuned (Fig.5 at 3.5 rad/s, Fig.11 at 15 rad/s respectively.) or R is 150% detuned, (Fig.6 at 3.5 rad/s, Fig.12 at 15 rad/s respectively.), the steady errors in the speed estimation appear. The estimation is the worst if both of the k_e and R are 150% detuned in Fig.7 and Fig.13. (Fig.7 at 3.5 rad/s, Fig.13 at 15 rad/s respectively.)

The neural estimator is then used for both of the rotor speed and stator resistance with the d-axis current command set to 5 A. When k_e is accurate, the speed estimations work well both at 3.5rad/s (Fig.8) and 15 rad/s (Fig.14). When the torque constant is 150% guessed, a steady error appears in speed estimation as shown in Fig.9a and Fig.15a and the resistance estimations are nearly the same as the previous one. (Fig.9b at 3.5 rad/s, Fig.15b at 15 rad/s respectively.)

3.2 Experimental Results: Rotor Speed and Stator Resistance Estimation

The NN estimators are implemented using a DSP TMSC320C30 (40 MHz) controller. The DSP controller utilizes a voltage vector control to produce PWM gate signals for IGBT VSI inverter. (Fig.16) 2 line currents and dc link voltage is fed into DSP through 3, 12-bit A/D converters. An 18,000 ppr encoder monitors the rotor position, speed and shaft torque. It is also used as a 300 line crude encoder in the algorithm.

At first, the neural speed estimator is used without any parameter adaptation at 3.5 and 15 rad/s. To show the effect of detuning of R and k_e on the speed estimation. Fig.17 and Fig.23 are the accurate speed neural estimation with the accurate k_e and R. (Fig.17 at 3.5 rad/s, Fig.23 at 15 rad/s, respectively). When the k_e is 150% detuned (Fig.18 at 3.5 rad/s, Fig.24 at 15 rad/s respectively.) or R is 150% detuned, (Fig.19 at 3.5 rad/s, Fig.25 at 15 rad/s) steady errors appear in the speed estimation. The estimation is the worst if both of the k_e and R are 150% detuned (Fig.20 at 3.5 rad/s, Fig.26 at 15 rad/s).

The neural estimator is then used for both of the rotor speed and stator resistance with the d-axis current command set to 5 A. When k_e is accurate, the speed estimations given in Fig.21and Fig.27 work well both at 3.5 rad/s (Fig.21) and 15 rad/s (Fig.27). When the torque constant is 150% guessed, a steady error appears in speed estimation (Fig.22a at 3.5 rad/s Fig.28a at 15 rad/s) and the resistance estimations are nearly the same. (Fig.22b at 3.5 rad/s, Fig.28b at 15 rad/s respectively).

4.0 Rotor Position, Speed and Torque Constant Estimation in Sensorless Control

Since the synchronous frame does not have the position information, the $\alpha\beta$ reference frame is used. As the model is highly nonlinear, direct estimation of the rotor position by neural net requires lots of neurons or hidden layers, which makes it impractical for the real time control. The alternative way is to get the rotor position from the estimation of the $\alpha\beta$ back-emfs. The position estimator with the torque constant and speed is shown in Fig.29. The $\alpha\beta$ discrete currents are

$$i_\alpha(k+1) = i_\alpha(k) + \frac{Ts}{L} * (v_\alpha(k) - E_\alpha(k) - Ri_\alpha(k)) \qquad (11)$$

$$i_\beta(k+1) = i_\beta(k) + \frac{Ts}{L} * (v_\beta(k) - E_\beta(k) - Ri_\beta(k))$$

$$where \quad E_\alpha(k) = -k_e 0.5 p\omega(k)\sin(\theta(k)),\ \ E_\beta(k) = k_e 0.5 p\omega(k)\cos(\theta(k)).$$

The adjustable model is similar to the previous one except that the back-emfs and resistance are their estimated values. The current errors are

$$e_{i\alpha}(k+1) = i_\alpha(k+1) - \hat{i}_\alpha(k+1) = (\hat{E}_\alpha(k) - E_\alpha(k))\frac{Ts}{L} + (\hat{R} - R)\frac{Ts}{L} i_\alpha(k)$$

$$e_{i\beta}(k+1) = i_\beta(k+1) - \hat{i}_\beta(k+1) = (\hat{E}_\beta(k) - E_\beta(k))\frac{Ts}{L} + (\hat{R} - R)\frac{Ts}{L} i_\beta(k) \qquad (12)$$

If the resistance and the inductance are correctly guessed, the current errors will go to zero when the $\alpha\beta$ back-emfs are accurately estimated. The estimated position, speed and torque constant are obtained from the estimated back-emfs.

$$\hat{\theta}(k+1) = -\tan^{-1}\left(\frac{\hat{E}_\alpha(k+1)}{\hat{E}_\beta(k+1)}\right), \quad \hat{\omega}(k+1) = \frac{\sqrt{\hat{E}_\alpha^{\,2}(k+1) + \hat{E}_\beta^{\,2}(k+1)}}{0.5p\hat{k}_e} \qquad (13)$$

4.1 Simulation Results: Rotor Position Estimation

For the estimations of rotor speed and rotor position, the motor runs at the operation point of 3.5 rad/s and 20% of rated torque. At this low speed, signal to noise ratio is low and the estimations become difficult. In Fig.30, the initial position guess is 0 rad. Fig.31 shows the $\alpha\beta$ back-emf estimations. Fig.32 shows the estimated and real speeds. Initial speed guess before 5 sec is 0 rad/s.

At low speed of 15 rad/s, in Fig.33, the initial position guess is 0 rad. Fig.34 shows the $\alpha\beta$ back-emf estimations. Fig.35 shows the estimated and real speeds. Initial speed guess before 5 sec is 0 rad/s.

4.2 Experimental results: Rotor Position Estimation

For the estimations of rotor speed and rotor position, the motor runs at the operation point of 3.5 rad/s and 20% of rated torque. At this low speed, signal to noise ratio is lower and the estimations become difficult. In Fig.36, the initial position guess is 0 rad. The training of random neural weights needed about 500 msec, which is 80% of an electrical cycle. Fig.37 shows the $\alpha\beta$ back-emf estimations. Fig.38 shows the estimated and real speeds. Initial speed guess before 5 sec is 0 rad/s.

At low speed of 15 rad/s, in Fig.39, the initial position guess is 0 rad. The training of random neural weights needed about 80 msec, which is 60% of an electrical cycle. Fig.40 show the $\alpha\beta$ back-emf estimations. Fig.41 show the estimated and real speeds. Initial speed guess before 5 sec is 0 rad/s.

5.0 Conclusions

Neural network estimators are shown to track the varying parameters, speed and position at different speeds of PMSM drives with consistent performance. Compared to other methods, the NN are adaptive to operation conditions and are easy in design. Simulation and experimental results are used to justify the claim.

References

[1] K.H. Kim, S.K.Chung, G.W. Moon, I.C. Baik, M.J.Youn. "Parameter estimation and control of permanent magnet synchronous motor drive using model reference adaptive technique." IEEE IECON 1995, pp387-392.

[2] Zelia M.A. Peixoto " Speed control of permanent magnet motors using sliding mode observers for induced emf , position and speed estimation", IEEE 1995 IAS, Annual Meeting, pp.1023-1028.

[3] Taskeshi Furuhashi " A position and velocity sensorless control of brushless dc motors using an adaptive sliding observer" IEEE 1990 IAS, Annual Meeting, pp1188-1192.

[4 Joohn sheok Kim, Seung-ki Sul. " High performance PMSM drives without rotational position sensor using reduced order observer" IEEE 1995 IAS, Annual Meeting, pp75-82.

[5] Jorge Solona, Maria I.Valla, Carlos Muravchik. " A nonlinear reduced order observer for PMSM" IEEE Tran. IE. Vol.43, No.4. August 1996 pp.492-497 .

[6] T.J.E. Miller " Brushless Permanent Magnet and Reluctance Motor Drives." Clarendon Press. 1989.

[7] A. Wijenayake, J..Bailey, M. Naidu. "A DSP based position sensor elimination method with an on-line parameter identification scheme for permanent magnet synchronous motor drives". IEEE IAS Meeting 1995, pp. 207-215.

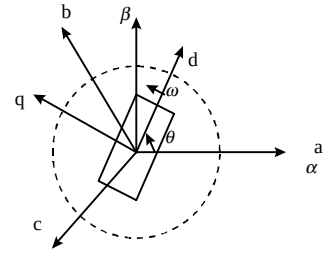

Fig.1 Stationary and synchronous frames in PMSM.

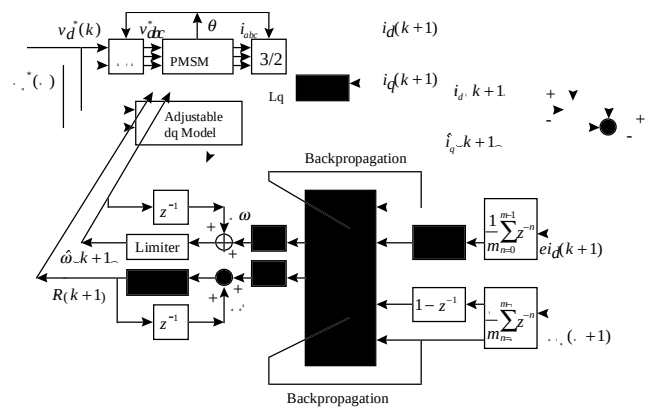

Fig.2 Block diagram of NN estimator for rotor speed and stator resistance.

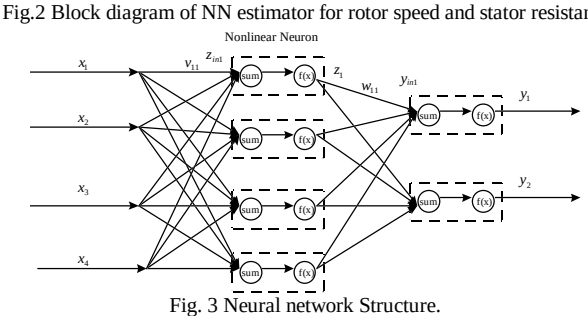

Fig. 3 Neural network Structure.

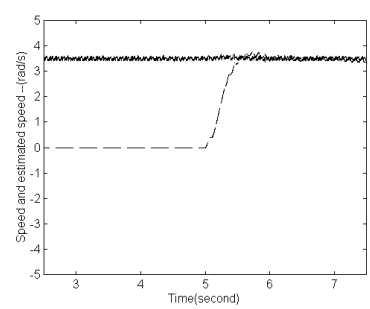

Fig.4 Neural estimator for rotor speed only at 3.5 rad/s. The k_e and R are at their accurate values.

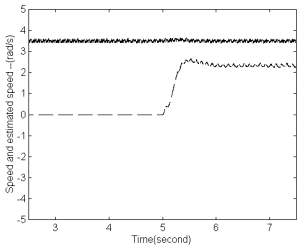

Fig.5 Neural estimator for rotor speed with k_e 150% detuned and R is at accurate value.

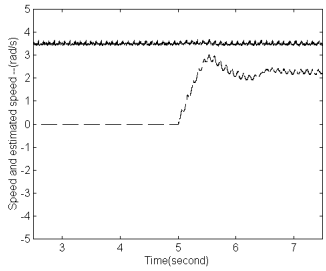

Fig.6 Neural estimator for rotor speed only at 3.5 rad/s. The k_e is at its accurate value and R is 150% detuned.

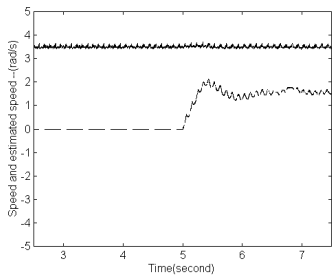

Fig.7 Neural estimator for rotor speed only at 3.5 rad/s. The k_e and R are at their 150% values.

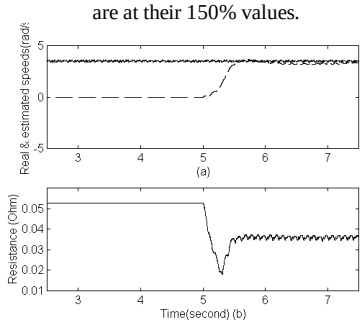

Fig.8 Neural estimator for rotor speed with stator resistance adaptation at 3.5 rad/s. k_e is at its accurate value (a) real and estimated speeds. (b) R estimation.

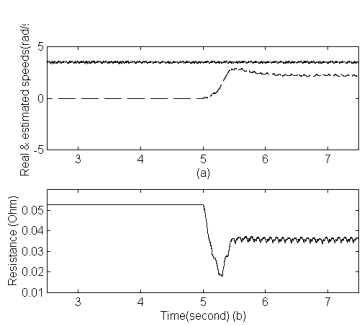

Fig.9 Rotor speed estimation with stator resistance adaptation at 3.5 rad/s. k_e is 150% detuned. (a) real and estimated speeds. (b) R estimation.

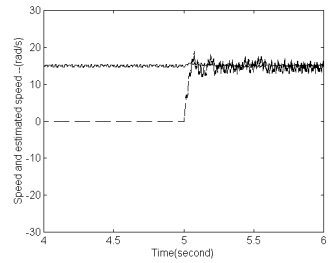

Fig.10 Neural estimator for rotor speed at 15 rad/s. k_e and R are at their accurate values.

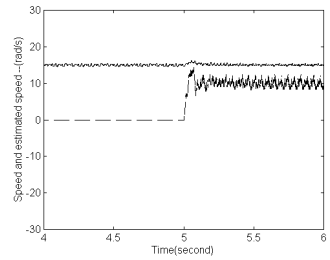

Fig.11 Neural estimator for rotor speed only at 15 rad/s. The k_e is 150% detuned and R is at its accurate values.

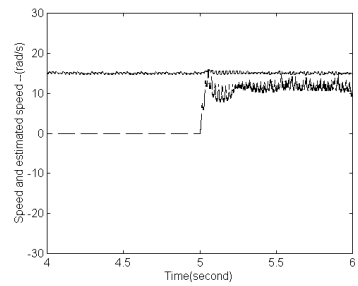

Fig.12 Neural estimator for rotor speed only at 15 rad/s. The k_e is at its accurate value and R is 150% detuned.

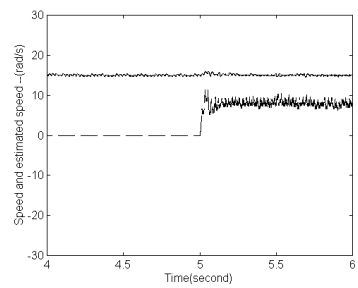

Fig.13 Neural estimator for rotor speed only at 15 rad/s. The k_e and R are at their 150% values.

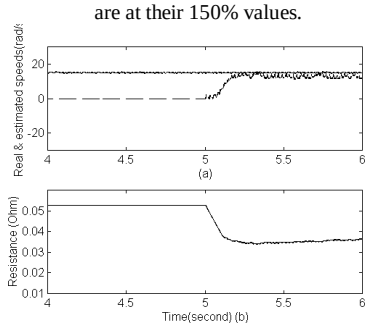

Fig.14 Rotor speed estimation with stator resistance adaptation at 15 rad/s. k_e is at its accurate value (a) real and estimated speeds. (b) R estimation.

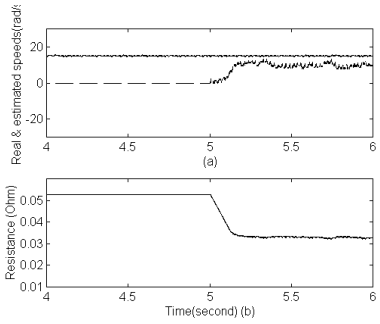

Fig.15 Rotor speed estimation with stator resistance adaptation at 15 rad/s. k_e is 150% value (a) actual and estimated speeds. (b) R estimation.

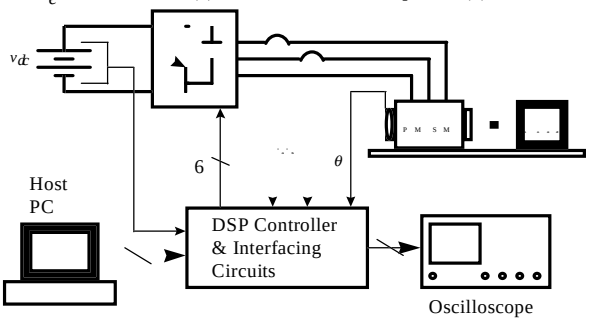

Fig.16 The PMSM experimental set-up.

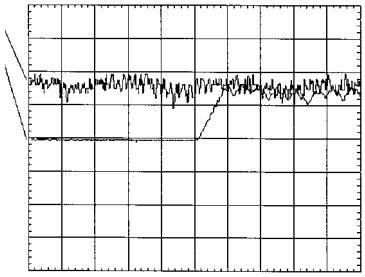

Fig.17 Speed estimation at 3.5 rad/s, real speed (upper one) 2.5 rad/s /div, estimated speed (lower one) 2.5 rad/s/div, time 500ms/div. Estimation starts at midpoint. Before the estimation, both parameters Ke and R are 100% tuned at off-line values.

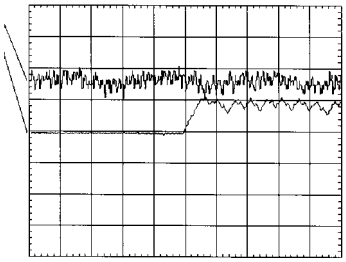

Fig.18 speed estimation at 3.5 rad/s, real speed (upper one) 2.5 rad/s /div, estimated speed (lower one) 2.5 rad/s/div, time 500ms/div. Estimation starts at midpoint. Before the estimation, Ke 150% detuned, R is 100% tuned.

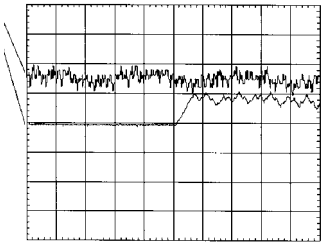

Fig.19 Speed estimation at 3.5 rad/s, actual speed (upper trace) 2.5 rad/s /div, estimated speed (lower trace) 2.5 rad/s/div, time 500ms/div. Estimation starts at midpoint. Before the estimation, R is 150% detuned and Ke is 100% tuned.

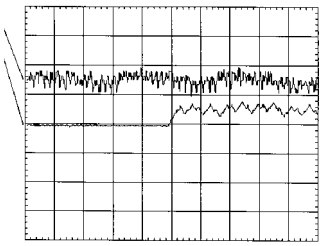

Fig.20 Speed estimation at 3.5 rad/s, real speed (upper one) 2.5 rad/s /div, estimated speed (lower one) 2.5 rad/s/div, time 500ms/div. Estimation starts at midpoint. Before the estimation, both Ke and R are 150% detuned.

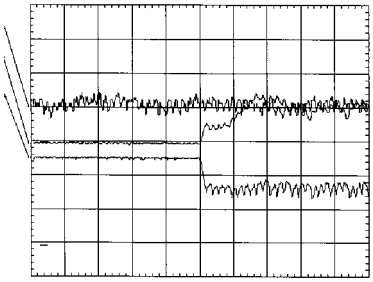

Fig.21 Speed and resistance estimation at 3.5 rad/s, real speed (upper one) 2.5 rad/s /div, estimated speed (middle one) 2.5 rad/s/div, resistance (lower one) 0.02 ohm/div, time 500ms/div. Estimation starts at midpoint. Before the estimation, R is 150% tuned. Ke is 100% tuned.

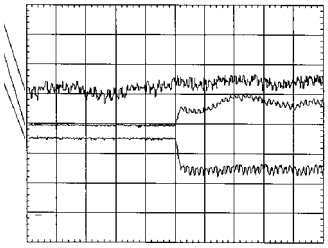

Fig.22 Speed and resistance estimation at 3.5 rad/s, real speed (upper one) 2.5 rad/s /div, estimated speed (middle one) 2.5 rad/s/div, resistance (lower one) 0.02 ohm/div, time 500ms/div. Estimation starts at midpoint. Before the estimation, R is 150% tuned. Ke is 150% tuned

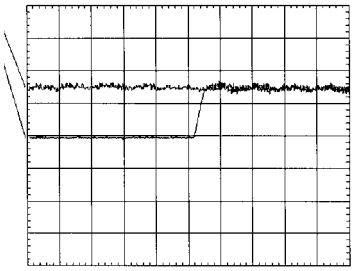

Fig.23 speed estimation at 15 rad/s, real speed (upper one) 10 rad/s /div, estimated speed (lower one) 10 rad/s/div, time 500ms/div. Estimation starts at midpoint. Before the estimation, both Ke and R are 100% tuned.

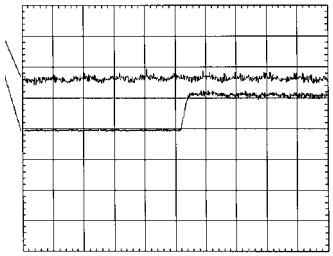

Fig.24 speed estimation at 15 rad/s, real speed (upper trace) 10 rad/s /div, estimated speed (lower trace) 10 rad/s/div, time 500ms/div. Estimation starts at midpoint. Before estimation, Ke 150 % detuned, R 100% tuned.

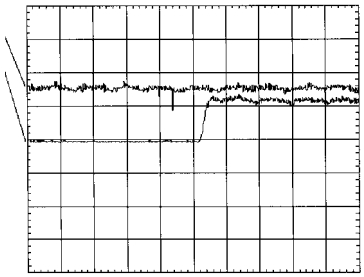

Fig.25 speed estimation at 15 rad/s, real speed (upper one) 10 rad/s /div, estimated speed (lower one) 10 rad/s/div, time 500ms/div. Estimation starts at midpoint. Before the estimation, R 150 % detuned , Ke 100% tuned.

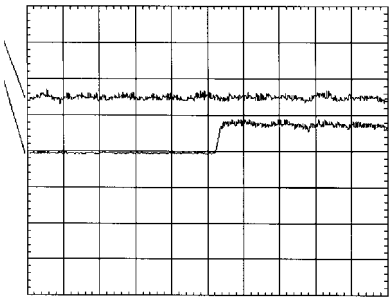

Fig.26 speed estimation at 15 rad/s, real speed (upper one) 10 rad/s /div, estimated speed (lower one) 10 rad/s/div, time 500ms/div. Estimation starts at midpoint. Before the estimation, both Ke and R are 150 % detuned.

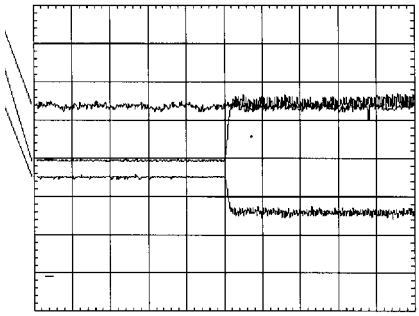

Fig.27 speed and resistance estimation at 15 rad/s, real speed (upper one) 10 rad/s /div, estimated speed (middle one) 10 rad/s/div, resistance (lower one) 0.02 ohm/div, time 500ms/div. Estimation starts at midpoint. Before the estimation, R is 150% detuned. Ke is 100% tuned.

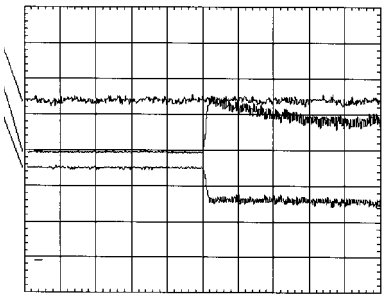

Fig.28 speed and resistance estimation at 15 rad/s, real speed (upper one) 10 rad/s /div, estimated speed (middle one) 10 rad/s/div, resistance (lower one) 0.02 ohm/div, time 500ms/div. Estimation starts at midpoint. Before the estimation, R is 150% detuned. K_e is 150% detuned.

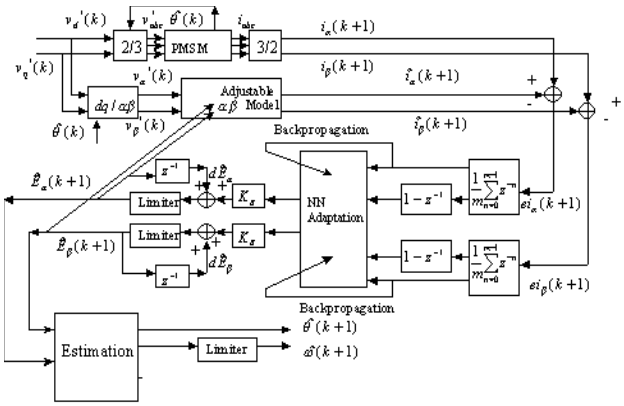

Fig.29 NN estimator block diagram for rotor position and speed.

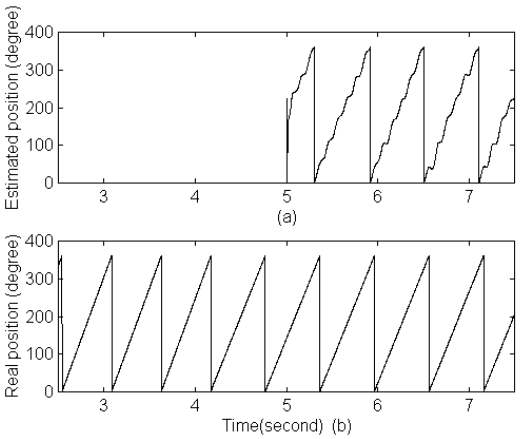

Fig.30 Position estimation at 3.5-rad/s. (a) estimated (b) actual position.

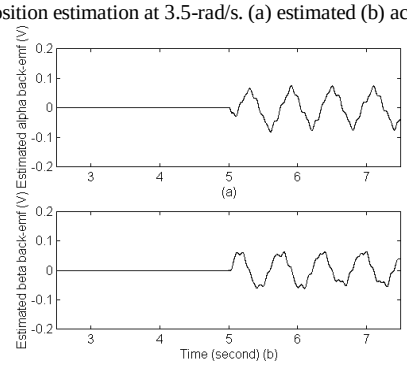

Fig.31 Back-emfs estimation at 3.5 rad/s

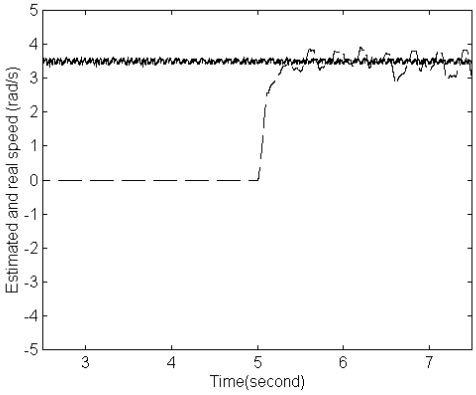

Fig.32 Estimated and actual rotor speeds at 3.5 rad/s. Dotted line is estimated speed.

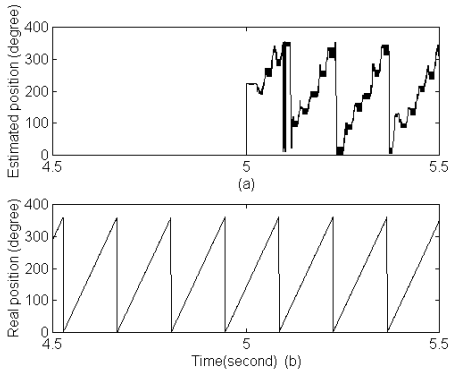

Fig.33 Rotor position estimation at 15 rad/s. (a) estimated position. (b) actual position.

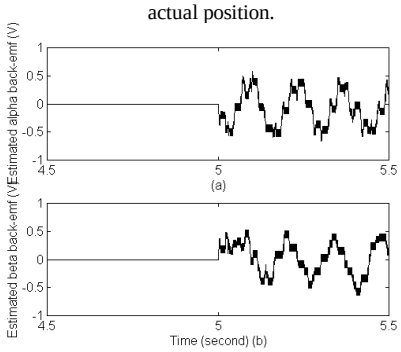

Fig.34 $\alpha\beta$ back-emf estimation at 15 rad/s.

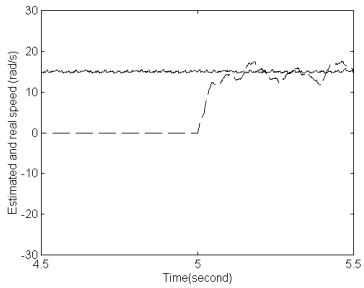

Fig.35 Estimated and actual rotor speeds at 15 rad/s. Dotted line is estimated speed.

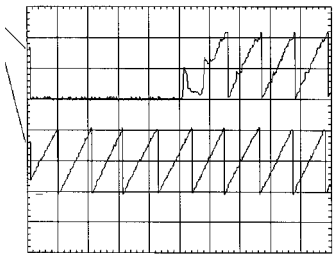

Fig.36 Position estimation at 3.5 rad/s, estimated position (upper) 3.14 rad/div, real position (lower) 3.14 rad/div, time 500ms/div. Estimation starts at midpoint.

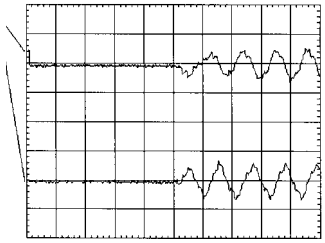

Fig.37 Position estimation at 3.5 rad/s, α backemf (upper) 0.2 v/div, β backemf (lower) 0.2 v/div, 500ms/div. Estimation starts at midpoint.

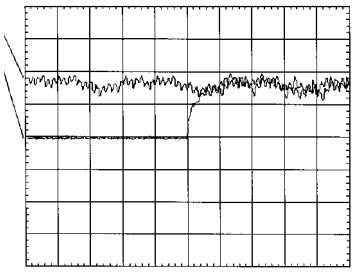

Fig.38 Speed estimation at 3.5 rad/s, real speed (upper one) 2.5 rad/s/div, estimated speed (lower one) 2.5 rad/s/div, time 500ms/div. Estimation starts at midpoint.

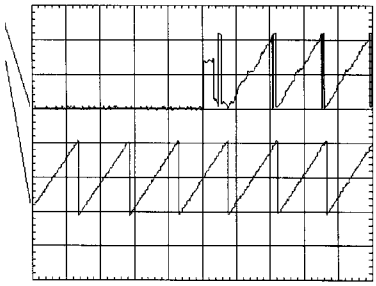

Fig.39 Position estimation at 15 rad/s, estimated position (upper one) 3.14 rad/div, real position (lower one) 3.14 rad/div, time 100ms/div. Estimation starts at midpoint.

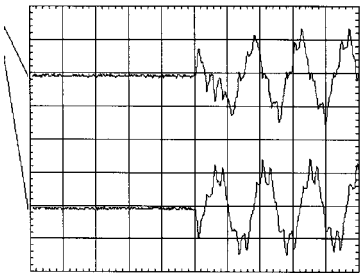

Fig.40 Position estimation at 3.5 rad/s, α backemf (upper) 0.4 v/div, β backemf (lower) 0.4 v/div, 100ms/div. Estimation starts at midpoint.

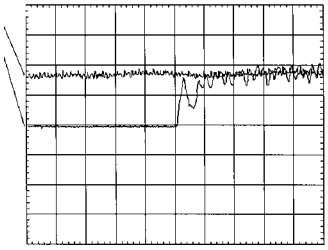

Fig. 41 Speed estimation at 15 rad/s, actual speed (upper) 10 rad/s/div, estimated (lower) 10 rad/s/div, 100ms/div. Estimation starts at midpoint.