PSO-Based PI Controller for Speed Sensorless Control of PMSM

A.M. Nazelan, M. K. Osman, A.A.A. Samat and N. A. Salim
Faculty of Electrical, Universiti Teknologi Mara
13500 Permatang Pauh, Pulau Pinang, Malaysia
E-mail: annazelan@gmail.com

Abstract. In the controlling three phase AC motor especially permanent magnet synchronous motor (PMSM), good controller is required to achieved the desired target of speed. This paper presents a comparison among different technique of speed and position estimation by using Proportional and Integral controller (PI) tune by using Heuristic, MATLAB PID auto tuner and Particle Swarm Optimization (PSO) method of sensorless control for PMSM. The reason of this study is to identify which method can produce better result. Simulation that have been conducted by using PSO optimization technique h produce better result than the conventional controller in terms of system parameter such as rise time (Tr), settling time (Ts), percent overshoot (%Os), percent undershoot (%Us) and root mean square error (RMSE) of PMSM.

Keywords: Permanent Magnet Synchronous Motor, PI controller, Particle Swarm Optimization, Artificial Neural Network, Model Reference Adaptive Control.

1. INTRODUCTION

Permanent Magnet Synchronous Motor (PMSM) has been widely used for various industries of technologies in robotics, electrical and automation. PMSM are well known because of its high power density, large torque to inertia ratio, high efficiency and simple structure. It consists of two primary parts which is permanent magnet that rotates called rotor and static core surrounded by three equally space winding called stator. There have two type of rotor. First is surface mounted permanent magnet where the magnets are attached at the surface of the rotor and second is interior mounted permanent magnet where the magnets are attached inside the rotor. Permanent magnets on the motor generate rotor flux. So the electrical rotor position is same with the rotor flux position [1]. For controlling the PMSM, it is necessary to have the information of speed and position of the rotor and normally encoder has been used to identify it [2]. Since the mechanic sensor is too expensive, increase the cost and decrease the stability, sensorless methods are implemented. It can save some space because installation of encoder requires bigger space [3].

Nowadays, sensorless technique for Permanent Magnet Synchronous Motor (PMSM) has various types. Generally, sensorless techniques have two methods which is back electromagnetic force (EMF) estimation and signal injection method [4], [5]. Back EMF method offer a good performance at high speed revolution per minute (RPM) for PMSM. But, at low speed and zero speed, the estimation accuracy will be decrease because the parameter reading is too small compare to the noise [6]. For signal injection method, their offer...
good performances at all state include zero speed. But, disadvantage of this method is requiring extra hardware and delay problem due to digital implementation of the digital filters and control algorithm [7].

The demand for variable speed drive in both low and high power applicants has resulted in great variety of products from different researcher, each offering a great variety of features [8]. A variety of techniques and strategies have been developed to overcome all this problem and weakness such as Sliding Mode Observer (SMO), Extended Kalman Filter (EKF), Artificial Neural Network (ANN) and Model Referenced Adaptive System (MRAS) [2], [9]–[11].

In this paper focus is on comparing the MRAS algorithm with same adaptive mechanism but different tuning method in order to identify which method can provide better result for speed and position tracking. Conventional method that have been propose using Proportional and Integral controller (PI)[6]. But, it was reported that low speed region still hunger with accurate controller. Therefore, this research is intend to improve the existing controller in term of decreasing the time to reach reference speed of PMSM with another type of tuning method. Particle Swarm Optimization (PSO) method was being prove the effectiveness for best parameter tuning in many application before.

Model reference adaptive system (MRAS) consist of two functional block which is reference model and adjustable model [12]–[15]. Main idea of this system is comparing the reference parameter (current $i_d$ & $i_q$ for this case) between actual parameter. Error of this signal will be use to estimate the speed and position of the PMSM. Then, the signal is fed into the adaptation algorithm to estimate the quantity of which is use to tune the adjustable model. This method is simple and required less computation [6].

I. PMSM MODEL

In this section, PMSM mathematical model are describe to understand the behavior. The PMSM can be modeled in stationary reference ($\alpha$, $\beta$) and rotor ($d$, $q$) as shown in Fig. 1.

The PMSM equivalent circuit as shown in fig. 2 are considered in order to determine the mathematical model of the PMSM.

Fig. 2 (a) and (b) is the equivalent circuit for voltage at $d$-axis and $q$-axis. From fig.2 the voltage equation can be derived as:
\[ V_d = R(i_d) + L_d \frac{di_d}{dt} - \omega \psi_q \]  
\[ V_q = R(i_q) + L_q \frac{di_q}{dt} - \omega \psi_d \]

where \( R \) is the phase resistor winding, \( \omega \) is the electrical angular velocity of the rotor and \( \psi_d, \psi_q \) is flux for \( d \) and \( q \). Flux equation be derived as:

\[ \psi_d = L_d(i_d) + \psi_m \]  
\[ \psi_q = L_q(i_q) \]

In the equation of flux above, \( \psi_m \) is the flux that generated by magnetic pole and stator.

2. POSITION AND ESTIMATION METHOD

A. Model Reference Adaptive System

Model reference adaptive control system (MRAS) scheme was first proposed by Professor Witark of the United States in 1958 [16]. The general view and idea MRAS is a closed loop system that contains reference model and adjustable model [12], [15]. This system will compare the output of plant (PMSM) with the desired response from reference model. The parameters from the plant will update into the system and then, the system will determine the error between reference models. If the parameters have an error, adjustable model will automatically tune and counter the error to match with reference model. The reference model is independent from the rotor speed, calculation of variable from the terminal voltage and current. Meanwhile, for the adaptive model is dependent on it.

The difference between these state variables is then used to drive an adaptation mechanism which generates the estimated speed (\( \hat{\omega} \)) [17]. Generalized of this system is shown in Fig. 2. Input for reference and adjustable model is same which is voltage in stationary form (\( u_d \) & \( u_q \)) that project from three phase (\( u_a, u_b \) & \( u_c \)).

B. Model Reference Adaptive System on PI Controller

Mainly, MRAS sensorless control is based on model of the motor at the rotating reference plane. In the rotor rotating reference frame, the model of the PMSM stator current describe as follows:
In order to simplify matrix in (8), assume:

\[
\frac{d}{dt}\left[ \begin{array}{c}
  i_d \\
  i_q \\
\end{array} \right] = \frac{R}{L_d} i_d + \frac{\omega_L}{L_d} i_q + \frac{V_d}{L_d}
\]

(5)

\[
\frac{d}{dt}\left[ \begin{array}{c}
  i_d \\
  i_q \\
\end{array} \right] = -\frac{\omega_L}{L_q} i_d - \frac{R}{L_q} i_q + \frac{\omega_L}{L_q} i_d
\]

(6)

where \(i_d, i_q\) is \(dq\)-axis stator current, \(u_d, u_q\) is \(dq\)-axis stator voltage, \(R\) is stator resistance, \(L_d, L_q\) is stator inductance and \(\psi_r\) is rotor permanent magnet flux. By estimating \(dq\)-axis current, estimated speed can be formulate as follows [16]:

\[
\hat{\omega} = \int_0^t \left[ k_1 i_d \dot{i}_d - i_d \dot{i}_d + \frac{\psi_f}{L} (i_q - \dot{i}_q) \right] d\tau
\]

\[
+ k_2 i_q \dot{i}_q - i_q \dot{i}_q - \frac{\psi_f}{L} (i_d - \dot{i}_d) \right] + \hat{\omega}(0)
\]

(7)

where \(\hat{\omega}\) is estimated rotor angular speed, \(\hat{\omega}, \dot{\omega}\) is \(dq\)-axis stator estimated current, \(k_1, k_2\) is PI regulator coefficients. The equation (2) and (3) will transform to matrix equation:

\[
\frac{d}{dt}\left[ \begin{array}{c}
  i_d + \frac{\psi_m}{L_d} \\
  i_q \\
\end{array} \right] = \left[ \begin{array}{c}
  \frac{-R}{L_d} \frac{L_d}{L_d} \frac{L_d}{L_d} \\
  -\frac{\omega_L}{L_d} \frac{L_d}{L_d} \frac{L_d}{L_d} \\
  \frac{1}{L_d} \\
  0 \\
\end{array} \right] \left[ \begin{array}{c}
  -R \frac{L_d}{L_d} \frac{L_d}{L_d} \\
  \frac{L_d}{L_d} \\
  0 \\
  \frac{1}{L_d} \\
\end{array} \right] \left[ \begin{array}{c}
  i_d \\
  i_q \\
  V_d + \frac{R \psi_m}{L_d} \\
  \psi_q \\
\end{array} \right] \]

(8)

In order to simplify matrix in (8), assume:

\[
\left[ \begin{array}{c}
  i'_d \\
  i'_q \\
\end{array} \right] = \left[ \begin{array}{c}
  i_d + \frac{\psi_m}{L_d} \\
  i_q \\
\end{array} \right]
\]

(9)

\[
V'_d = V_d + \frac{R \psi_m}{L_d}
\]

\[
V'_q = \psi_q
\]

Substitute equation (9) into (8) to simplify the matrix equation of (8):

\[
\frac{d}{dt}\left[ \begin{array}{c}
  i'_d \\
  i'_q \\
\end{array} \right] = -\frac{R}{L_d} i'_d + \frac{L_d}{L_d} i'_q + \frac{V'_d}{L_d}
\]

(10)

\[
\frac{d}{dt}\left[ \begin{array}{c}
  i'_q \\
\end{array} \right] = -\frac{\omega_L}{L_d} i'_d - \frac{R}{L_q} i'_q + \frac{V'_q}{L_q}
\]

(11)

Then, the value of \(i'_d, i'_q, V'_d\) and \(V'_q\) in the equation (10) and (11) must be expand back based on equation at (8):

\[
\frac{d}{dt}\left[ \begin{array}{c}
  i'_d \\
  i'_q \\
\end{array} \right] = -\frac{R}{L_d} i_d + \frac{\omega_L}{L_d} i_q + \frac{V_d}{L_d}
\]

(12)
\[ \frac{d}{dt}[i'_q] = -\frac{\omega L_q}{L_d} i_d + \frac{\omega L_g}{L_q} \psi_m - \frac{R}{L_q} i_q + \frac{V}{L_q} \quad (13) \]

While,

\[ \omega e = \frac{\omega L_q}{L_d} \quad (14) \]

Then, substitute (14) into (12) and (13):

\[ \frac{d}{dt}[i'_d] = -\frac{R}{L_d} i_d + \omega e(i_d) + \frac{V}{L_d} \quad (15) \]

\[ \frac{d}{dt}[i'_q] = -(\omega e) i'_q + \frac{\omega L_q}{L_d} \psi_m - \frac{R}{L_d} i_q + \frac{V}{L_q} \quad (16) \]

To generalized the error:

\[ e = i - \hat{i} \quad (17) \]

where \( e \) is the error between reference and adjustable model, \( i \) and \( \hat{i} \) is reference and adjustable current. The simple matrix can be form below:

\[ \frac{d}{dt} e = Ae - (\hat{A} - A)i \quad (18) \]

\[ \frac{d}{dt} e_d = \left[ \begin{array}{cc} \frac{-R}{L_d} & \frac{\omega L_q}{L_d} \\ -\omega L_d & -\frac{R}{L_d} \end{array} \right] e_d + \left[ \begin{array}{c} \omega \psi_m \\ -\omega \psi_m \end{array} \right] \]

From the previous formula:

\[ i = \left[ \begin{array}{c} i_d \\ i_q \end{array} \right], \hat{i} = \left[ \begin{array}{c} \hat{i}_d \\ \hat{i}_q \end{array} \right] \quad (19) \]

Substitute the formula in (19) into (17)

\[ e = \left[ \begin{array}{c} i_d - \hat{i}_d \\ i_q - \hat{i}_q \end{array} \right] \]

\[ J = \left[ \begin{array}{cc} 0 & -1 \\ 1 & 0 \end{array} \right] \]

\[ \frac{d}{dt} e = Ae - W \quad (20) \]

\[ W = (\hat{A} - A)\hat{x} \]

The adaptive law shown in (17) can be obtain by implement PI Controller.

\[ \dot{\omega} = \left[ k_p + k_i \right] \hat{x}_i \cdot i_s \]

\[ \hat{x}_i \cdot i_s = \zeta \omega \]

\[ \zeta \omega = e^t J' \cdot \hat{x}_i \cdot i_s \]
Substitute the equation (18) into (19):

\[
e = i'_d i_q - i'_d i_q - \frac{\psi_r}{L} (i'_q - i_q)
\]  

(21)

C. Particle Swarm Optimization

Particle Swarm Optimization (PSO) is an optimization technique developed by Dr. Eberhart and Dr. Kennedy in 1995 that inspired by social behavior of bird flocking or fish schooling. Due to many advantage can be found by using this technique including simplicity and easy implementation, this algorithm can be used widely in many field such as classification, neural network training, machine study, signal procession and etc. This algorithm is also appropriate for parameter optimization in continues search space in many dimensions [18].

PSO concept is using a number of agents (particles) that constitute a swarm moving around in the search space looking for the best solution. Each particles in search space adjust its ‘flying’ according to its own flying experience well as the flying experience of other particles and the best are called (pbest). Individual particles also have the best knowledge of values by the group (gbest). Each agent uses the information connecting to: current position (x,y), current velocity (v_x,v_y), the distance between the current position and gbest group to change the position. Velocity and position of each agent modified by equation (18) and (19) below.

\[
\hat{\omega} = [k_p + k_i] \left( i'_d i_q - i'_d i_q - \frac{\psi_r}{L} (i'_q - i_q) \right) + \omega_{\theta r}(0)
\]

(19)

\[
\hat{\theta} = \int_{0}^{t} \hat{\omega}
\]

\[
e = v^k_i + c_1 \cdot rand_1 \cdot (pbest_i - s^k_i) + c_2 \cdot rand_2 \cdot (gbest - s^k_i)
\]

(22)

\[
s^{k+1}_i = s^k_i + v^{k+1}_i
\]

(23)

Where:

- \( v^k_i \) = current velocity of agent i at iteration k
- \( v^{k+1}_i \) = new velocity of agent i at iteration k
- \( s^k_i \) = current position of agent i at iteration k
- \( s^{k+1}_i \) = denotes the position of agent i at the next iteration k+1
- pbest_i = personal best agent i
- gbest = global best of the population
- \( c_1 \) = adjustable cognitive acceleration constant (self-confidence)
- \( c_2 \) = adjustable cognitive acceleration constant (swarm confidence)
- \( rand_{1,2} \) = random number between 0 and 1

For this paper, PSO is used to optimize the value of PI controller that can minimum the error and overshoot of the PMSM sensorless control. There are four step for this algorithm optimize the value of PI that illustrate in Fig. 4 below.

![Flowchart for the steps of PSO](image-url)
Step 1: Generation of initial conditions of each agent
For the first stage, the initial input and speed of each agent will be choose randomly. Each agent will set the current search point to pbest and at the same time, gbest also will determine the best value of pbest. Then, the best value for the number agent is stored for evaluation process.

Step 2: Evaluation of searching point of each agent
Second stage is evaluation for the stored value of agent where output of the system will produce an error and overshoot. The data will be analyze and new agent will be replace to see the output of system behavior. If the agent is better than pbest at this time, it will replace current value. Also, if the value of pbest is better than gbest, it will replace current value that stored inside gbest. Then, the best agent will stored.

Step 3: Modification of each searching point
By yusing the equation (19)(20), the value of agent will be declared to find the point.

Step 4: Checking to exit condition
In this step, several test were carried out before the end of the process. If the system not reach the desired output, the process will repeated again from step 1 and if otherwise the process will stopped.

Table 1: Parameter of PSO

| Variable | Value | Description |
|----------|-------|-------------|
| n        | 25    | Size of the swarm “no. of birds” |
| Bird step | 10    | Max. no. of “bird steps” |
| dim      | 2     | Dimension of the problem |
| c1       | 1.5   | Velocity constant |
| c2       | 0.15  | Velocity constant |
| w        | 0.9   | Momentum of inertia |
| minKp    | 10    | Min. value of agent for Kp |
| maxKp    | 60    | Max. value of agent for Kp |
| minKi    | 0     | Min. value of agent for Ki |
| maxKi    | 5     | Max. value of agent for Ki |

In this research, controller that being use for adapation method for MRAS is Heuristic-PI, MATLAB PI and PSO-PI controllers.

Heuristic-PI Controller
Proportional Integral (PI) controller is a common feedback loop consist of direct and integrator gain that the gain value is set based on try and error (manually tune). This method generally used in industrial control system as shown in Fig. 5.

MATLAB-PI Controller
MATLAB-PI (M-PI) controller is the PI controller that tuned by using MATLAB tuner. It much simpler method where user only set the response time and transient behavior before the system automatically develop the gain value.
Figure 6: PID tuner that provide by MATLAB Simulink block

**PSO-PI controller**

PSO-PI controller is the PI controller that gain of the system is tuned by using Particle Swarm Optimization. Based on PSO-PI controller block diagram in Fig. 7, the simulation was carried out.

![Figure 7: PSO-PI Controller Block Diagram](image)

3. SIMULATION RESULT

First, the simulation results are using Heuristic-PI (H-PI) controller, then MATLAB Autotune PI (M-PI), and lastly is Particles Swarm Optimization PI (PSO-PI) controller. Simulation is perform in the MATLAB Simulink and time is kept 0.3sec. Result are based on system performance metric such as rise time (Tr), settling time (Ts), percent overshoot (%Os), percent undershoot (%Us) and root mean square error (RMSE) where its verified under four (4) different conditions.

1) Speed and load both are constant [speed is 1500r/min and load torque is 2Nm].
2) Speed is constant and load varied [speed is 1500r/min and load torque is 0-0.075sec (2Nm), 0.075-0.15sec (1Nm), 0.15-0.225sec (2Nm) and 0.225-0.3sec (1Nm)].
3) Speed is varied and load is constant.
4) Speed and load both are varied.

For condition 1-

For condition 1, PSO-PI produce better result where Tr is 0.026751s, Ts is 0.036654s, Os is 0.9431% and RMSE is 309.7595.

For condition 2-

For condition 2, PSO-PI produce better result where Tr is 0.026751s, Ts is 0.036654s, Os is 0.9431% and RMSE is 309.7595.

For condition 2- Speed is constant and load torque is varied [speed is 1500r/min and load torque is 0-0.075sec (2Nm), 0.075-0.15sec (1Nm), 0.15-0.225sec (2Nm) and 0.225-0.3sec (1Nm)].
For condition 2, PSO-PI produce better result where Tr is 0.026782s, Ts is 0.036713s, Os is 0.8816% and RMSE is 309.8039.

For condition 3, speed is varied and load torque is constant [speed is 0-0.1sec (0-1500r/min), 0.1-0.18sec (1500r/min), 0.18-0.25sec (800r/min) and 0.25-0.3sec (1500r/min) and load torque is 2Nm].

For condition 4, speed and load torque are varied [speed is 0-0.1sec (0-1500r/min), 0.1-0.18sec (1500r/min), 0.18-0.25sec (800r/min) and 0.25-0.3sec (1500r/min) and load torque is 0-0.075sec (2Nm), 0.075-0.15sec (1Nm), 0.15-0.225sec (2Nm) and 0.225-0.3sec (1Nm)].

Table 2: Result by using H-PI controller for condition 1 to 4

|        | Cond. 1 | Cond. 2 | Cond. 3 | Cond. 4 |
|--------|---------|---------|---------|---------|
| Tr     | 0.027678| 0.027509| 0.09316 | 0.092181|
| Ts     | 0.039906| 0.039693| 0.26455 | 0.262614|
| %Os    | 1.5386  | 1.4781  | 1.1247  | 1.9413  |
| RMSE   | 311.5337| 310.4529| 78.2006 | 77.1321 |

Table 3: Result by using M-PI controller for condition 1 to 4

|        | Cond. 1 | Cond. 2 | Cond. 3 | Cond. 4 |
|--------|---------|---------|---------|---------|
| Tr     | 0.03434 | 0.034835| 0.10394 | 0.104045|
| Ts     | 0.29863 | 0.298435| 0.264172| 0.263029|
Table 4: Result by using PSO-PI controller for condition 1 to 4

|       | Cond. 1 | Cond. 2 | Cond. 3 | Cond. 4 |
|-------|---------|---------|---------|---------|
| %Os   | 2.7949  | 2.7285  | 1.6554  | 1.9325  |
| RMSE  | 345.1328| 346.5095| 80.5208 | 89.2568 |

4. CONCLUSION

In this paper a new design method is proposed to determine stability of step response in term of rise time, settling time, maximum overshoot and root mean square error. Amongst this three controllers that were tested, PSO-PI was found to be producing better results compared to H-PI and M-PI. It shows that this method can improve the dynamic performance of the system in a better way.

ACKNOWLEDGMENT

This paper was supported by the Fundamental Research Grant Scheme (FRGS) (FRGS/1/2015/TK04/UITM/02/12) funded by the Ministry of Education and Faculty of Electrical Engineering Universiti Teknologi MARA.

REFERENCES

[1] Z. Zheng, Y. Li, and M. Fadel, “Sensorless Control of PMSM Based on Extended Kalman Filter,” *Eur. Conf. Power Electron. Appl.*, pp. 1–8, 2007.
[2] M. Elbuluk, “A sliding mode observer for sensorless control of permanent magnet synchronous motors,” *Conf. Rec. 2001 IEEE Ind. Appl. Conf. 36th IAS Annu. Meet. (Cat. No.01CH37248)*, vol. 2, no. C, pp. 1273–1278, 2001.
[3] Z. Qiao, T. Shi, Y. Wang, Y. Yan, C. Xia, and X. He, “New sliding-mode observer for position sensorless control of permanent-magnet synchronous motor,” *IEEE Trans. Ind. Electron.*, vol. 60, no. 2, pp. 710–719, 2013.
[4] E. M. Fernandes, A. C. Oliveira, M. a. Vitorino, E. C. Dos Santos, and W. R. N. Santos, “Speed sensorless PMSM motor drive system based on four-switch three-phase converter,” *Proceedings. IECION 2014 - 40th Annu. Conf. IEEE Ind. Electron. Soc.*, pp. 902–906, 2014.
[5] Z. Q. Zhu and L. M. Gong, “Investigation of effectiveness of sensorless operation in carrier-signal-injection-based sensorless-control methods,” *IEEE Trans. Ind. Electron.*, vol. 58, no. 8, pp. 3431–3439, 2011.
[6] A. Mishra, V. Mahajan, P. Agarwal, and S. P. Srivastava, “MRAS based estimation of speed in sensorless PMSM drive,” *2012 IEEE 5th Power India Conf. PICONF 2012*, 2012.
[7] F. Cupertino, A. Guagnano, A. Altomare, and G. Pellegrino, “Position estimation delays in signal injection-based sensorless PMSM drives,” *3rd IEEE Int. Symp. Sensorless Control Electr. Drives, SLED 2012*, pp. 1–6, 2012.
[8] P. Kumar, M. Kumar, and S. Dahiya, “Sensorless Speed Control of PMSM using SVPWM Technique Based on MRAS Method for Various Speed and Load Variations,” *Proc. World Congr. Eng. 2015*, vol. I, no. 1, pp. 1–6, 2015.
[9] D. Fodor, J. Vass, Z. Juhász, and K. Biró, “Extended Kalman Filter Based Speed Sensorless AC Motor Control,” p. 6.
[10] R. K. R. Kumar, R. a. Gupta, and A. K. Bansal, “Identification and Control of PMSM Using Artificial Neural Network,” *2007 IEEE Int. Symp. Ind. Electron.*, pp. 30–35, 2007.
[11] Z. Wang, Q. Teng, and C. Zhang, “Speed identification about PMSM with MRAS,” *2009 IEEE 6th Int. Power Electron. Motion Control Conf.*, IPEMC ‘09, vol. 3, pp. 1880–1884, 2009.
[12] G. Wu and X. Xiao, “Speed controller of servo system based on MRAS Method,” *Proc. IEEE Int. Conf. Ind. Technol.*, 2009.
[13] P. Vaclavek and P. Blaha, “Synchronous Machine Drive Observability Analysis and Sensorless Control Design,” *2008 IEEE 2nd Int. Power Energy Conf. Pecon, Vols 1-3*, no. PECon 08, pp. 265–270, 2008.
[14] H. M. Kojabadi and M. Ghribi, “MRAS-based adaptive speed estimator in PMSM drives,” *Int. Work. Adv. Motion Control. AMC*, vol. 2006, no. 1, pp. 569–572, 2006.
[15] K. Jinsong, Z. Xiangyun, W. Ying, and H. Dabing, “Study of position sensorless control of PMSM based on MRAS,” Proc. IEEE Int. Conf. Ind. Technol., no. 3, pp. 3–6, 2009.

[16] W. Guo and Z. Guo, “Speed sensorless control of PMSM using model reference adaptive system and RBFN,” J. Networks, vol. 8, no. 1, pp. 213–220, 2013.

[17] M. Dursun, A. F. Boz, M. Kale, and M. Karabacak, “Sensorless Speed Control of Permanent Magnet Synchronous Motor With Hybrid Speed Controller Using Model Reference Adaptive,” vol. 3, no. 1, pp. 24–37, 2014.

[18] W. F. Tarmizi, I. Elamvazuthi, N. Perumal, K. Nurhanim, M. K. A. A. Khan, S. Parasuraman, and A. V. Nandedkar, “A Particle Swarm Optimization-PID controller of a DC Servomotor for Multi-Fingered Robot Hand,” 2016 2nd IEEE Int. Symp. Robot. Manuf. Autom. ROMA 2016, pp. 1–6, 2017.