

Design and Implementation of PID Control-based FSM Algorithm on Line Following Robot

Adnan Rafi Al Tahtawi¹, Yoyo Somantri², Erik Haritman³

¹Department of Computer Engineering, Politeknik Sukabumi

Jl. Babakan Sirna No. 25 Kota Sukabumi, Indonesia

^{2,3}Department of Electrical Engineering Education, Universitas Pendidikan Indonesia

Jl. Dr. Setiabudhi No. 207 Bandung, Indonesia

adnanrafi@polteksmi.ac.id

Abstract

Finite State Machine (FSM) is a control system methodology that describes system's behavior using three things, namely: state, event, and action. On a program, the system would be in one active state. The system can switch or move to another state if it gets a certain input or event. In this paper, FSM based on Proportional-Integral-Derivative (PID) controller algorithm will be implemented on line follower robot. PID controller is one of system control methods that many used recently. FSM based on PID controller is implemented to keep robot tracking the line trajectory as well. The test result shows that designed algorithm can work well and can be used as a based algorithm of this robot.

Keywords: algorithm, Finite State Machine (FSM), Proportional-Integral-Derivative (PID), robot, line follower

Abstrak

Finite State Machine (FSM) adalah sebuah metodologi perancangan sistem kontrol yang menggambarkan tingkah laku atau prinsip kerja sistem dengan menggunakan tiga hal, yaitu: state (keadaan), event (kejadian) dan action (aksi). Pada suatu program, sistem akan berada pada salah satu state yang aktif. Sistem dapat beralih atau bertransisi menuju state lain jika mendapatkan masukan atau event tertentu. Pada makalah ini, akan diimplementasikan algoritma FSM berbasis pengendali Proporsional-Integral-Derivatif (PID) pada robot pengikut garis. Pengendali PID merupakan salah satu metode kendali sistem yang banyak digunakan saat ini. Algoritma FSM berbasis pengendali PID diterapkan dengan tujuan agar robot mampu menjajaki lintasan dengan baik. Hasil pengujian menunjukkan bahwa algoritma yang dirancang mampu bekerja dengan baik dan dapat digunakan sebagai dasar algoritma dari robot ini.

Kata kunci: algoritma, Finite State Machine (FSM), Proporsional-Integral-Derivatif (PID), robot, line follower

I. INTRODUCTION

Line follower robot is one of autonomous robot which has designed for researches, industrial requirements, or robot competitions. Referring to its name, the task of this robot is to follow a line trajectory. There is needed a specified based algorithm for this robot to finish the task. One of the algorithms which can be implemented is Finite State Machine (FSM). This algorithm is built in three steps, i.e. state, event, and action. The third steps can be illustrated as follow: state is the robot conditions (such as moving forward, turn left, turn right, or turn back), event is the sensor's state when robot doing a task and action is DC motor action due to state and event conditions. Thus, this algorithm called in this paper as a based algorithm of line follower robot. In another hand, Proportional-Integral-Derivative (PID) controller

was promising in line follower robot control method. PID controller will minimize the error when robot moving on the line trajectory. Developments of FSM algorithm have been conducted they are self-adjusting FSM [1], decentralized Evolutionary Robotics (ER) based on FSM [2], and robot control teaching with state machine-based [3]. PID controller also implemented on this robot to reduce wobbling [4].

In this paper, FSM based on PID algorithm will be explained to give an overview and additional references for software design of line follower robot. After introduction in Section I, FSM algorithm as a based algorithm and PID controller overview are described in Section II. It is followed by Section III that explains the system designs of software and hardware. Section IV focus on the implementation, followed by system testing result

and analysis. Finally, Section V gives the conclusion of this paper.

II. SYSTEM ALGORITHM

A. FSM Algorithm

Finite State Machine (FSM) was a control system methodology that describes system's behavior using three things, i.e. state, event and action [5]. At one moment in a significant period time, the system will be at one active state and may be move to another state. These state transitions are generally also accompanied by actions which taken by the system when responding the input.

Figure 1 shows the FSM with two states, two inputs, and four different action outputs. As shown in figure 1, when the system is started, it will be at state0. In this condition, system will produce action1 if it gets event0, whereas if event1 occurs then action2 will be executed and the system will move forward to state2 and so on. FSM consist of two types, namely non-output FSM and output FSM [5]. Non-output FSM is used for language recognition on a computer. The input that has been entered will be processed and obtained the result. Meanwhile, output FSM is used for designing a machine or system. FSM has several advantages including: simple, predicted response, soft computing, relatively flexible, classic artificial intelligent that can be used for various systems, can converts from abstract to code easily. Several disadvantages, i.e. on game implementation not interested due to predictively and it is so difficult if implemented on the complex system.

B. PID Controller

Proportional-Integral-Derivative (PID) controller is often referred to as a 'three-term' controller [6]. This controller is currently one of the most frequently used controllers in industry. Each of controllers has a task as following description [6]:

1) *Proportional*: the error is multiplied by a gain K_p . A very high gain may cause instability, and a very low gain may cause the system to drift away.

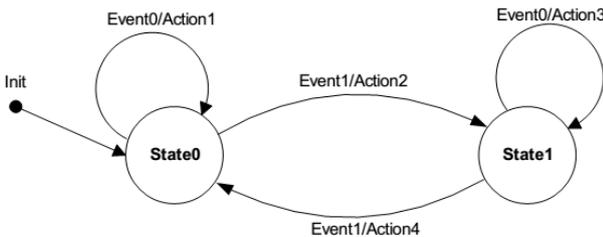


Figure 1. Simple state diagram example [5]

2) *Integral*: the integral of the error is taken and multiplied by a gain K_i . The gain can be adjusted to drive the error to zero in the required time. A too high gain may cause oscillations and a too low gain may result in a sluggish response.

3) *Derivative*: the derivative of the error is multiplied by a gain K_d . Again, if the gain is too high the system may oscillate and if the gain is too low the response may be sluggish.

Based on description above, it can be generalized that PID controller can improve system response according to desired response that determined. Input of this controller is error measurement and the output is control signal that given to the actuator. In continuous-time, block diagram of PID controller is depicted in Figure 2.

Referring to figure 2, the control signal of a PID controller in time-domain can be expressed as:

$$u(t) = K_p e(t) + K_i \int_0^t e(t) dt + K_d \frac{de(t)}{dt} \quad (1)$$

$$K_i = \frac{K_p}{T_i} \quad (2)$$

$$K_d = K_p T_d \quad (3)$$

where $u(t)$ is the output from the controller and $e(t) = r(t) - y(t)$, in which $r(t)$ is the desired set-point (reference input) and $y(t)$ is the plant output, T_i and T_d are known as the integral and derivative action time, respectively. For digital microcontroller implementation, PID controller should be in discrete-time domain. The control signal of PID controller given by (1) is transformed to discrete-time PID controller as follow:

$$u(kT) = K_p e(kT) + K_i T \sum_{k=1}^n e(kT) + K_d \frac{e(kT) - e(kT - T)}{T} \quad (4)$$

where $u(kT)$ and $e(kT)$ are control signal and error signal in discrete-time with T sampling time.

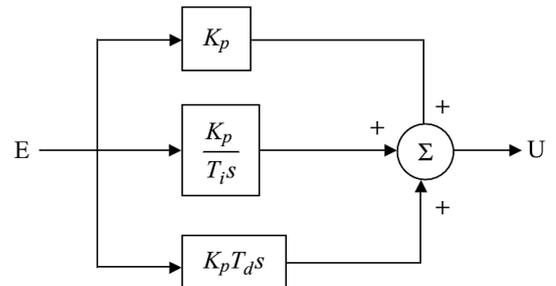


Figure 2. Continuous-time PID controller [6]

III. SYSTEM DESIGN

A. Hardware

FSM algorithm will be implemented on line follower robot. The robot is designed using microcontroller as main device. In addition, it is also used sensor as line detector and DC motor as actuator. Microcontroller that used in this study is ATmega328 which integrated on DFRduino Romeo v1.0 board, as depicted in Figure 3.

This board is used due to compatibility requirement for line follower robot. Its specifications based on [7] are: flash memory 32 KB, internal oscillator 16 MHz, 14 channels digital I/O, 6 PWM channels (Pin11, Pin10, Pin9, Pin6, Pin5, Pin3), 8 channels 10-bit analog I/O, USB connection, auto sensing/switching power input, ICSP header for directly program downloading, serial interface TTL level, support AREF, support header pin male and female, available socket for APC220 RF module and DF Bluetooth module, 5 units 12C interface pin sets, 2 units driver DC motor with maximum current 2A, 7 buttons input, DC supply 7-12 V, DC output 5V/3.3V and voltage output external, dimension 90 x 80 mm.

Hardware designs including electrical which are microcontroller, sensor, and DC motor. It also mechanical design as well as robot casing. Block diagram of electrical design is depicted in Figure 4 and mechanical design of robot presented in Figure 5.



Figure 3. DFRduino Romeo v1.0 board [7]

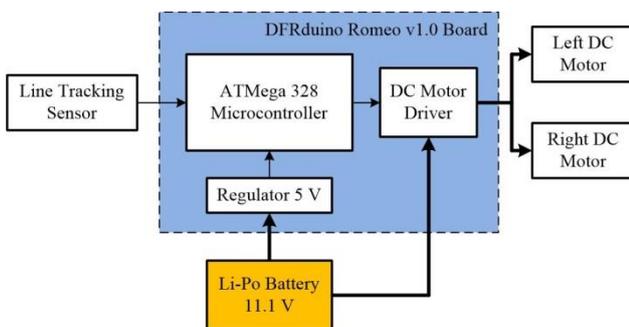


Figure 4. Hardware configuration

Block diagram in Figure 4 shows the connection between each electrical component in the robot. Solid line is powertrain on robot and thin line is data transfer.

Sensor that used in this robot is line tracker sensor in a form of module. This module uses photodiode as sensor to detect light reflection from infrared signal. Photodiode will give binary logic '1' to microcontroller if it receives signal from infrared, while binary logic '0' will be given if signal not received. Sensor modules used are six units and placed in series in front of robot.

This robot uses battery cell as power supply. Batteries that used are three cells of Lithium-Polymer (Li-Po) with nominal voltage of 3.7 V. It is also used for DC motor power supply through DC motor driver which has been integrated on DFRduino Romeo v1.0 module.

DC motor is used as a driving component. Microcontroller will receive data from sensor and send signal to DC motor according to sensor signal condition. Signals that send from microcontroller are Pulse Width Modulation (PWM) signal for speed control and binary logic signal for motor direction. PWM signal is digital switching signal in high frequency so it can modulate how amount the voltage that will be sent from battery to DC motor, while another signal is used for controlling motor direction with changes pole of battery so the robot can move forward, backward, turn right, turn left, turn back, and stop.

B. Design of PID-based FSM Algorithm

FSM algorithm designed is presented in Table 1 and Figure 6. There are two states when the robot is employed. The first state is robot moving forward and the second state is robot turn condition.

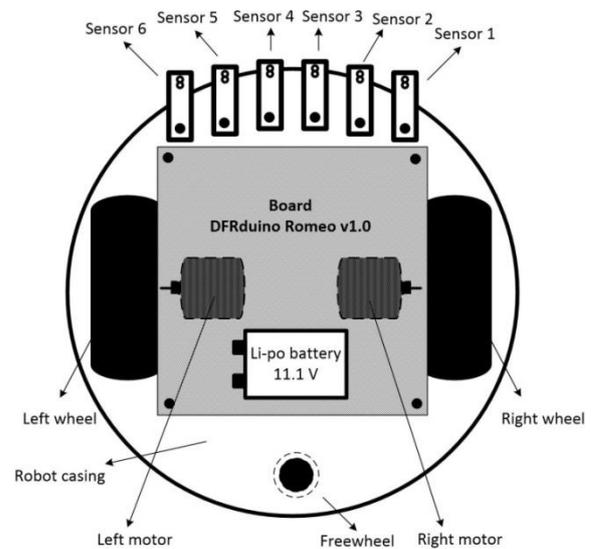


Figure 5. Mechanical robot design

In the first state, robot will receive the event from sensor's data and will generate an action through DC motor to keep its movement in ideal straight forward. This case has been done using PID control scheme. In the second state, robot will receive the event also from sensor's data and will generate an action through DC motor for following line trajectory as well. Both of states will switch as long as robot still working. Generally, we describe state as robot's state, event as robot's input, and action as robot's output.

Sensor's data logic is high or '1' if it detects white surface and it is low or '0' if detects black surface. Line trajectory that used has black color on top of white background with line width 3 cm and turn angle 90°.

Referring to FSM algorithm, robot's flowchart is divided into two conditions in accordance with robot's state, these are robot moving forward and robot turn condition. The aim of first flowchart is to keep robot moving forward perfectly. In this condition, PID control is used. PID will control both left and right motor speeds according to error measurement of sensor. This error is input for the controller and PWM signal is the output of the controller. PID constants (K_p , K_i , K_d) are obtained using Ziegler-Nichols tuning method. This method is used due to simplicity in terms of design and implementation. Ziegler-Nichols tuning method is presented in Table 2.

The second flowchart describes robot turn condition. This condition is also must work as well. PID controller effect which has been designed in

first flowchart will be observed in second flowchart. Turn conditions (left, right, and back) are conducted using different direction of both DC motor. This condition will be stopped when the robot's state is changed to the first state, and so on. Both of flowcharts are realized in microcontroller programming using C/C++ language and downloaded to DFRduino Romeo v1.0 module. These flowcharts are depicted in Figure 7 and Figure 8.

Table 1. FSM algorithm design

Finite State Machine (FSM) Algorithm		
State	Event	Action
Robot moving forward	011111	Rpwm < Lpwm
	001111	
	100111	
	110011	Rpwm = Lpwm
	111001	Rpwm > Lpwm
	111100	
	111110	
Robot turn conditions	100000	Rdir = LOW Ldir = HIGH (Turn right)
	110000	
	111000	
	000111	Rdir = HIGH Ldir = LOW (Turn left)
	000011	
	000001	
	000000	Stop

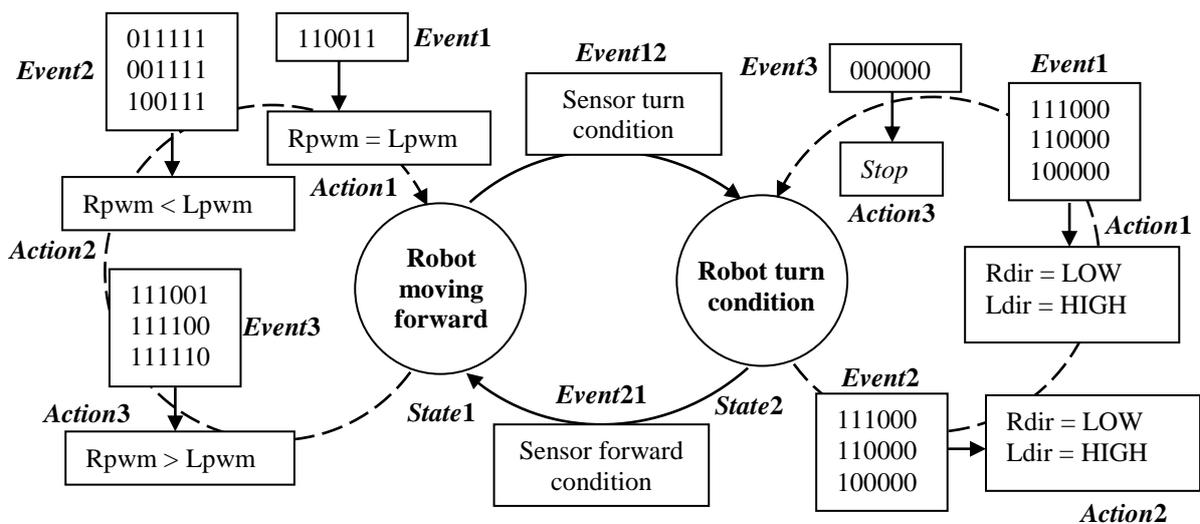


Figure 6. FSM diagram of line follower robot

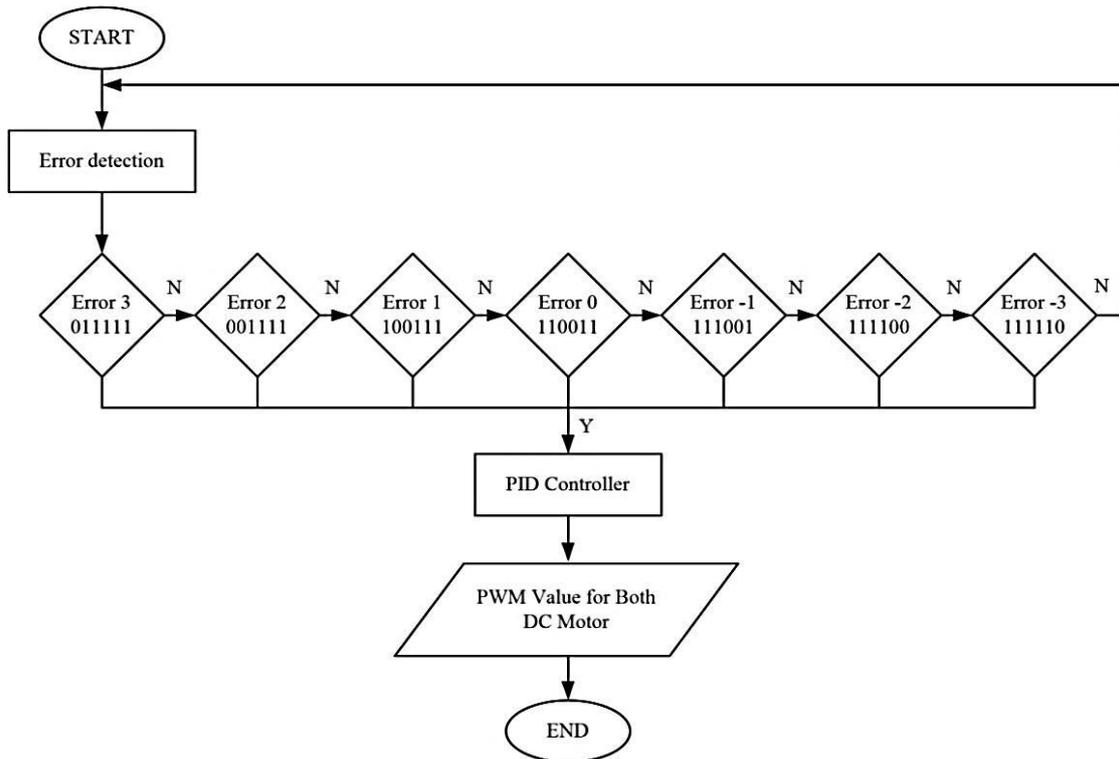


Figure 7. Flowchart of the first state

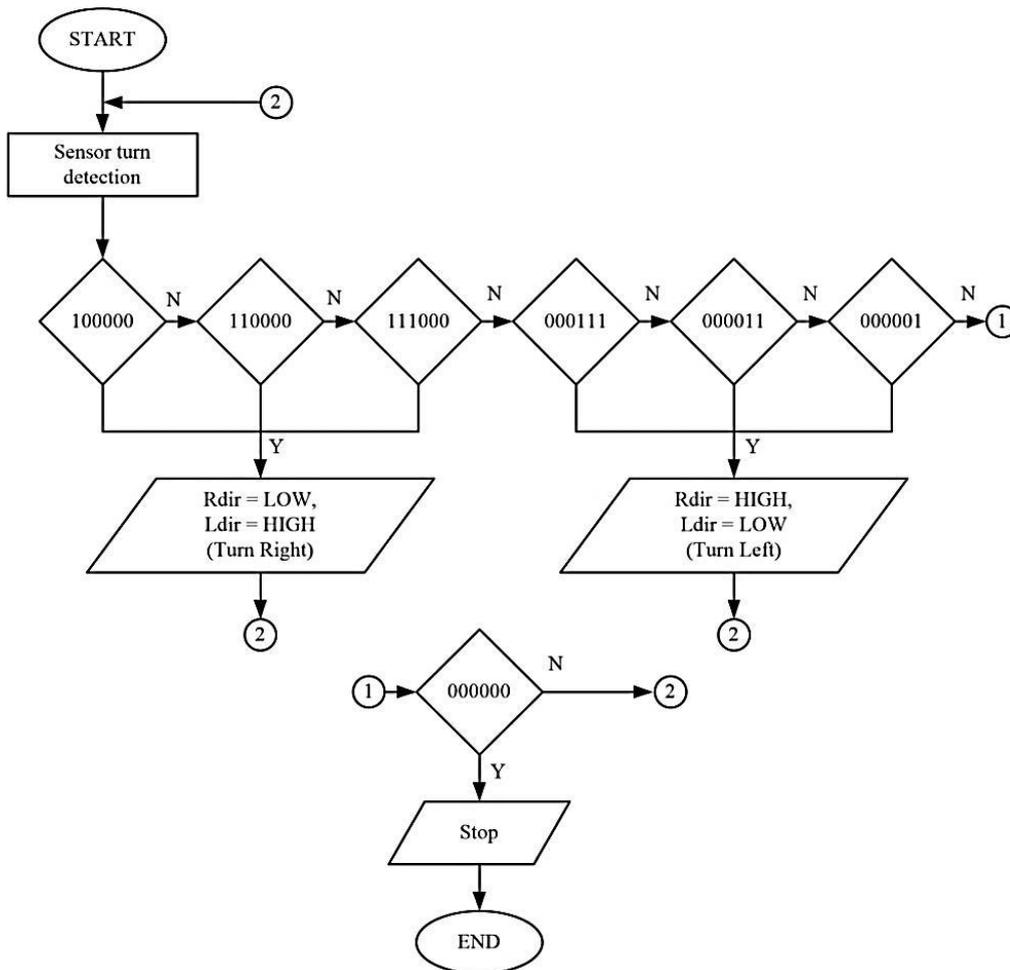


Figure 8. Flowchart of the second state

Table 2. Open-loop Ziegler-Nichols settings [6]

Controller	K_p	T_i	T_d
P	$\frac{T_I}{KT_D}$	-	-
PI	$\frac{0.9T_I}{KT_D}$	$3.3T_D$	-
PID	$\frac{1.2T_I}{KT_D}$	$2T_D$	$0.5T_D$

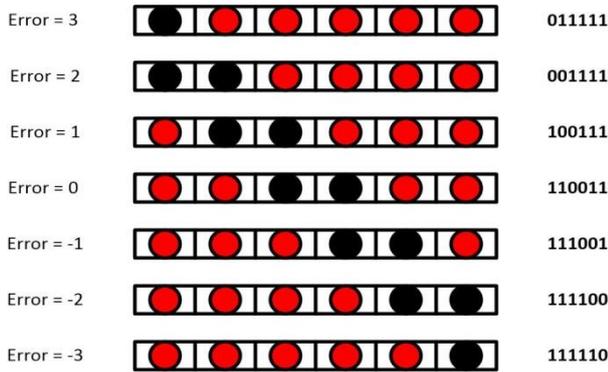


Figure 9. Ideal possibility sensor conditions in first state

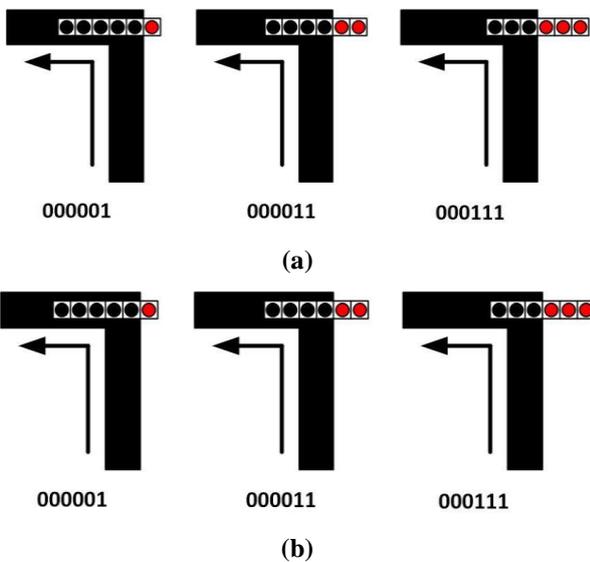


Figure 10. Ideal possibility sensor conditions in second state: (a) turn left (b) turn right

Sensor conditions that possible to happen in both first state and second state are illustrated in Figure 9 and Figure 10. However, this condition is ideal assumption due to simplicity in term of design the controller.

IV. IMPLEMENTATION

This section gives system implementation, testing results, and its analysis. Hardware implementation of this robot is shown in Figure 11.

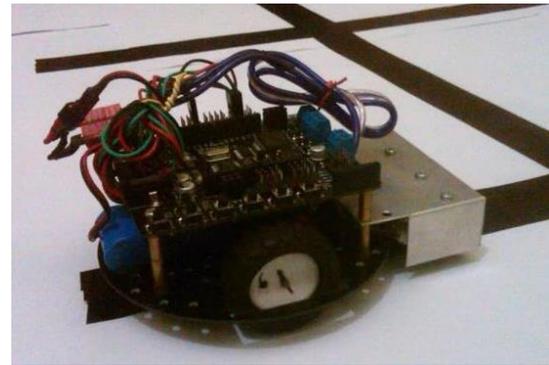


Figure 11. Line follower robot hardware

Table 3. PWM testing result

Experiment number	Bit PWM (0-255), DC	Driver output voltage (V)	
		Left motor	Right motor
1	0, 0%	0	0
2	64, 25%	2.6	2.7
3	127, 50%	5.4	5.5
4	192, 75%	7.8	7.9
5	255, 100%	10.7	10.8

There are two terms of system testing, first is PWM testing and second is FSM based on PID algorithm testing. PWM testing is done to make sure the robot's speed can be controlled, while algorithm testing is conducted to know the robot response when it doing a task.

A. PWM Testing

Testing of PWM is done using output voltage measurement from DC motor driver. Various duty cycles are given in form of 8-bit digital data (0-255) in microcontroller to generate PWM signal. Voltage measurement is placed in driver output with battery voltage nominal input 11.1V. Testing result is presented in Table 3.

From above PWM test result obtained that the amount of PWM duty cycle value is impact to the amount of output voltage from DC motor driver. The higher PWM duty cycle value the more high voltage given to DC motor. However, there is drop voltage in measurement. Maximum duty cycle doesn't generate maximum voltage. It is due to transistor semiconductor component inside DC motor driver.

B. Simulation Result

PID constants are determined firstly based on Ziegler-Nichols method. The constants on Ziegler-

Nichols method are K , T_I , and T_D . In this study, the gain constant is determined by $K = 3$ referring to maximum error of first state also based on open-loop model that depicted in Figure 12. This model obtained from experimental open-loop response that given to the robot using maximum error as an initial condition. From the test result obtained that the robot can track line trajectory by 4 seconds. Simulation of this response is then conducted using MATLAB/Simulink for designing the PID controller. Based on this response, to control this robot, integral and derivative action times are determined based on Table 2 with $T_I = 2\text{ s}$ and $T_D = 0.5\text{ s}$. Thus, there obtained $K_p = 1.6$, $K_i = 3$, and $K_d = 0.75$. Simulation result shows that the PID controller is eliminate the error faster than the open-loop response system (Figure 13). Robot can reach the line trajectory by 1.65 second although there is produce an overshoot.

C. Hardware Implementation

After PID controller simulation has been done, PID-based FSM algorithm hardware testing is conducted. This testing is overall system testing and main section of this study. Firstly, FSM algorithm is

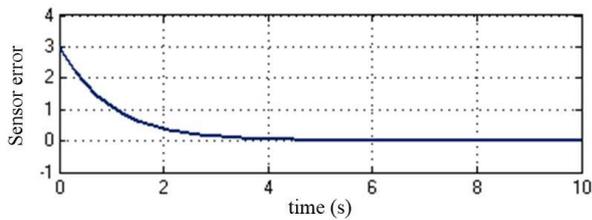


Figure 12. Open-loop system response

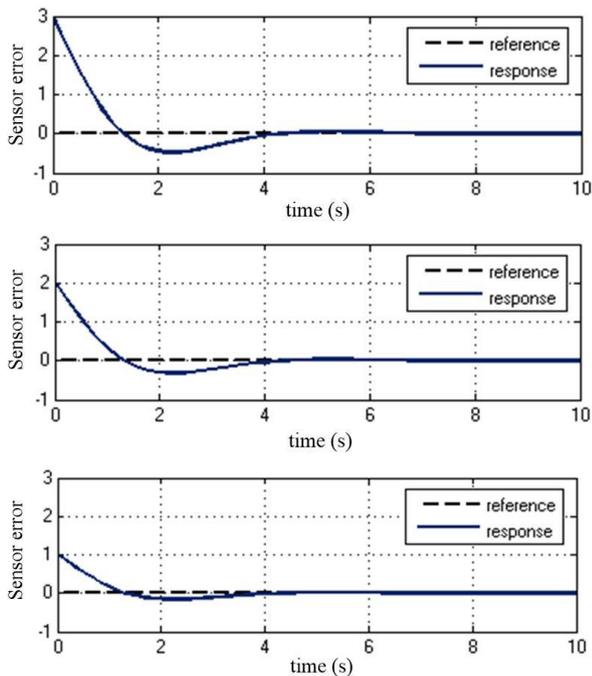


Figure 13. Closed-loop system response by error 3 (top), error 2 (middle), and error 1 (bottom)

realized in microcontroller programming according to system flowchart. In the first state, PID controller is placed. PID constant obtained is then implemented in microcontroller in accordance with (4). In the second state, the testing is employed in turn condition (right and left). PID controller response from the first state is observed in this state. Results of algorithm testing of both states first and second with error as the initial conditions are depicted in Figure 14 and Figure 15 below.

Overall algorithm testing result shows that FSM algorithm can work well according to system design. PID controller that implemented in first state is also effectively reducing robot error measurement. In the first state, robot can be moving forward on line trajectory as well. Error which given as initial condition is set to be zero by this controller. When robot in the second state, it shows that robot can track the turn line nicely. However, there is overshoot response when robot in this state. It is due to delay effect from sensor data transfer.

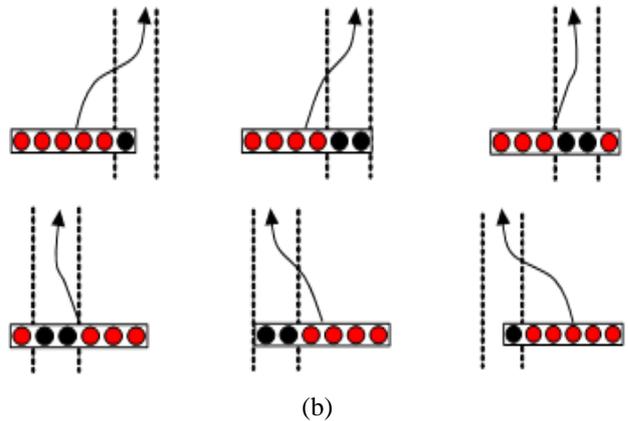


Figure 14. First state responses: $R_{pwm} < L_{pwm}$ (top), $R_{pwm} > L_{pwm}$ (bottom)

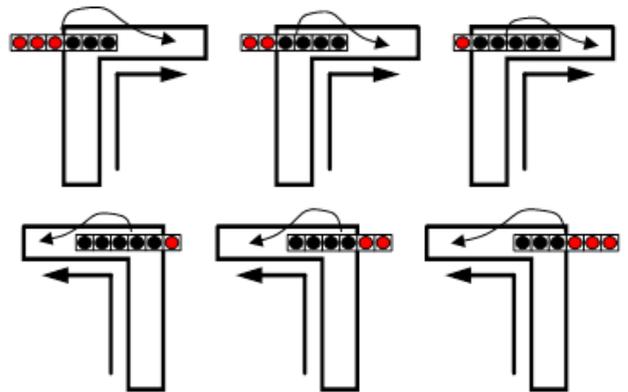


Figure 15. Second state responses: turn right (top), turn left (bottom)

V. CONCLUDING REMARKS

PID-based FSM algorithm is successfully designed and implemented on line follower robot. FSM is designed in two main states, named first state and second state. First state is robot moving forward, while second state is robot's turn condition. PID controller is designed and implemented in the first state on FSM algorithm to control robot movement. The test result shows that by combining both PID controller and FSM algorithm, it can be reach line follower robot response as well. These algorithms also can be used as based algorithm for this robot due to simplicity in terms of design and implementation.

REFERENCES

- [1] Schwartz, Scott, "Self-Adjusting Finite State Machines: an approach to Real-Time Autonomous Behavior in Robots", Computer Science Honors Theses, Paper 9, 2005
- [2] Konig, Lukas, *et al*, "Decentralized Evolution of Robotic Behavior Using Finite State Machines", *International Journal of Intelligent Computing and Cybernetics*, 2009
- [3] Levin, Ilya, *et al*, "Robot Control Teaching with a State Machine-based Design Method", *Int. Journal Engineering Education (IJEE)*, Vol. 20, No. 2, pp. 1-10, 2004
- [4] Nath, A.S. *et al*, "Implementation of PID Control to Reduce Wobbling in a Line Following Robot", *International Journal of Research in Engineering and Technology (IJRET)*, Vol. 02, Issue 10, 2013
- [5] Setiawan, I., "Design of Software Embedded System based on FSM", 2006
- [6] Ibrahim, Dogan, *Microcontroller Based Applied Digital Control*, Departement of Computer Engineering, Near East University Cyprus, Willey, 2006
- [7] Datasheet DFRduino RoMeo User Manual