Motion Planning System on an Autonomous Differential Wheeled Robot Using PIC18F4550

Wong Weng Hong, Mohamad Fauzi Zakaria[1,*], Rosli Omar[1]

[1] Department of Mechatronics and Robotics Engineering
Faculty of Electrical and Electronic Engineering
Universiti Tun Hussein Onn Malaysia
86400 Parit Raja, Batu Pahat, Johor, MALAYSIA

Abstract: This paper presents design and development of a motion planning system for an autonomous mobile robot that would find the shortest travelling path to reach its goal. Motion planning system is a system used in robotics for the process of detailing a task into discrete motions. Since there are disadvantages by implement only one type of movement algorithm on an autonomous mobile robot, such the longer travel time taken by line follower, and inaccuracy movement result by dead reckoning; a motion planning system is designed in order to enhance the robot movement more effectively especially for a competition. An autonomous robot base of ROBOCON (ABU Robot Contest) was used as hardware testing. This project was carried on by designing a PID line follower algorithm and dead reckoning algorithm. Nevertheless, the motion planning system was designed to combine these two algorithms. As the outcomes of this project, there is a printed circuit board (PCB) of PIC18F4550 microcontroller designed to support the motion planning system. In addition, the motion planning system was successfully implemented on the autonomous mobile robot by combining the two types of movement algorithms.

Keywords: Motion Planning System, Autonomous Robot, PID Line Follower, Dead Reckoning

1. Introduction

This project is to design a motion planning system for autonomous robot that able to combine the two types of movement algorithm, including Proportional-Integral-Derivative (PID) line follower algorithm and dead reckoning algorithm. By navigating an autonomous robot on the game field to achieve a distance waypoint, the description of travelling path tasks will be obtained by motion planning system as input [1] [2]. This allows the autonomous robot understands its position on the game field and interpreted the speed and turning commands sent to the robot's wheels. Thus, the robot is able to switch between line follower algorithm and dead reckoning algorithm under a motion planning system to complete its travelling path.

There are disadvantages found by implemented only one type of movement algorithm on autonomous mobile robot due to the lack of feedback information. Addition, there is longer travelling time taken by using line follower algorithm, especially in dealing with turning task. By using dead reckoning algorithm only, it is shown that the inaccuracy movement result due to the sliding surface of floor.

In completing this project, a PCB of PIC18F4550 is designed to support the motion planning system, which is able to combine two types of movement algorithm, such as PID line follower algorithm and dead reckoning algorithm. Afterward, the motion planning system is implemented on a given autonomous mobile robot base.

This project is carried on by using a PIC18F4550 microcontroller as the main controller of the system. The digital compass, incremental rotary encoder and digital optic sensor are used as movement algorithms input sensors. Nevertheless, a pair of brushless direct-current (BLDC) motors are use as main driving force for mobile robot. The motion planning system is designed by using MPLAB IDE and C18 language.

2. Literature Review

2.1 Dead Reckoning Algorithm

The algorithm of estimating one's current position based upon a previously determined position. It can be advancing the position based upon known or estimated speeds over elapsed time, and course [3]. A disadvantage of dead reckoning is that since new position are calculated solely from previous positions, the errors of the process are cumulative, and so the error in the position fix grows with time.

For the differential drive dead reckoning automobile robot, the formulas for the coordinates (x and y) and heading (θ) for a differential drive robot with encoders on both drives are shown as (1), (2), and (3) respectively.

$$\Delta x = R_w \cos(\theta)(T_1 + T_2) \quad (1)$$

$$\Delta y = R_w \sin(\theta)(T_1 + T_2) \quad (2)$$

$$\Delta\theta = 2\pi \cdot (T_1 - T_2)/T_R \quad (3)$$

In Fig. 1, a dead reckoning algorithm travelling path is shown, where T_1 are the encoder ticks recorded on left drive; T_2 are the encoder ticks recorded on right drive; R_W is the radius of each drive wheel; T_R is the number of encoder ticks recorded in a full, in place rotation.

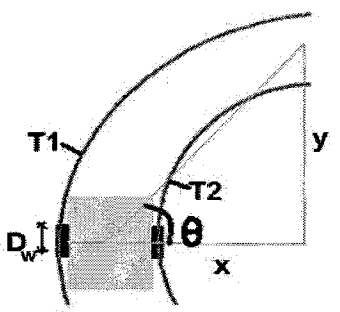

Fig. 1 Dead reckoning algorithm travelling path

2.2 PID Line Follower Algorithm

One idea in controlling a line following robot is using the PID control algorithm [4]. Most importantly, PID control uses continuous function to compute the motor speeds, so that the jerkiness of the previous condition can be replaced by a smooth response. The terms of Proportional-Integral-Derivative are described as three input values used in a simple formula to compute the speed of robot either in turning left or right. The PID line follower algorithm travelling path is shown in Fig. 2.

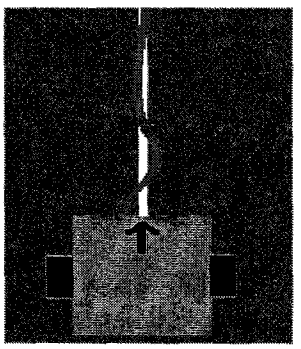

Fig. 2 PID line follower algorithm travelling path

The proportional, P, value is approximately proportional to automobile robot's position with respect to the line. If it is to the left of the line, the proportional term will be a positive value and vice verse of proportional term on right of the line.

The Integral, I, value records the history of automobile robot's motion. The derivative, D, is the rate of change of the proportional value, computed from the difference of the last two proportional values.

Initially, the algorithm should record all line sensors' data. Then, the program is able to determine the automobile robot either it is on centre of line, left to line or right to line. The proportional control term, P, is introduced first in computing algorithm to give a feedback to the robot to turn left or turn right. Then, derivative control term, D, starts in computing the motor speed is required to make a perfect turning for the robot. Lastly, integral control term, I, records all the history of robot for further computing progress.

3. Methodology

3.1 System Architecture

In Fig. 3, the system architecture block diagram based on a PIC18F4550 microcontroller designs is shown. PIC18F4550 is known as the main controller for this design. There are four main categories included for PIC18F4550 interfacing design, which are basic circuits for microcontroller, digital compass module interfacing, digital input interfacing and output from PIC.

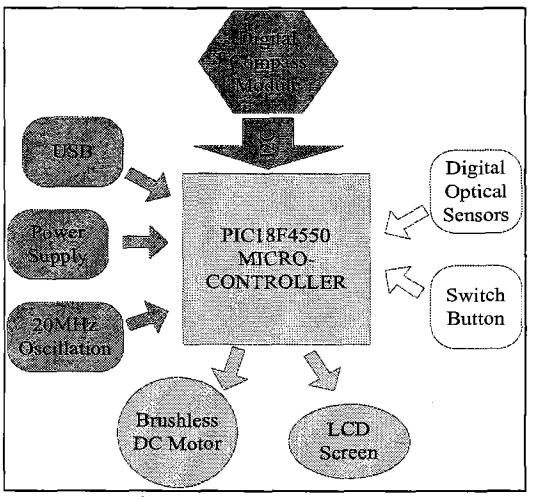

Fig. 3 System Architecture

3.2 Hardware Development

The top layer printed circuit board (PCB) is also called as main controller printed circuit board where the microcontroller functioning as 'brain' mounted on this circuit board. The overview of the components located in the top layer PCB or main circuit is shown in Fig. 4.

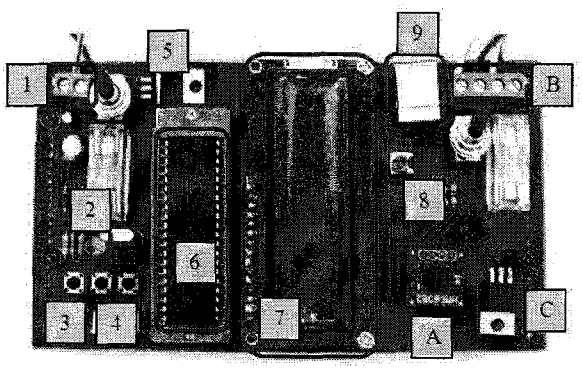

Fig. 4 Components on the top layer PCB

Those components labeled in Fig. 5, are described and summarized (see Table 1).

Table 1 Components on the Top of PCB

Label	Component's name and feature
1	12 voltage Li-Po Battery Supply
2	Top Layer PCB Indicator
3	Reset Button
4	Program Buttons
5	5 voltage Regulator (7805)
6	Microcontroller (PIC18F4550)
7	LCD screen
8	LCD Contrast Calibrator
9	USB port
A	Digital Compass Module (HMC6352)
B	24V Li-Po Battery Supply
C	3.3V Power Regulator (LM1117T)

The bottom layer as shown in Fig. 5 is an IO interfacing circuit board. This is because all inputs sensors or output controller pins are connected on this board. Since this bottom layer PCB is designed without any microcontroller on it, a top layer PCB is required in case.

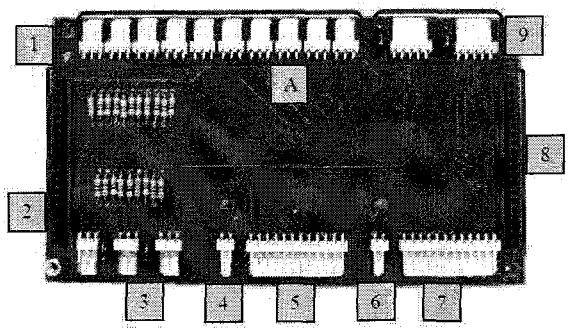

Fig. 5 Components on the bottom layer PCB

Those components labeled in Fig. 5, are described and summarized (see Table 2).

Table 2 Components on the Bottom of PCB

Label	Component's name and feature
1	Bottom Layer PCB Indicator
2	Left-sided Jumper Connector with Top Layer PCB
3	Encoder Ports
4	24 voltage supply to Brushless DC Motor 1
5	Controller Port to Brushless DC Motor 1
6	24 voltage supply to Brushless DC Motor 2
7	Controller Port to Brushless DC Motor 2
8	Right-sided Jumper Connector with Top Layer PCB
9	I2C Protocol Ports
A	Digital Input Ports (Digital Optical Sensor Ports)

3.3 Software Design and Development

3.3.1 Dead Reckoning Algorithm

A digital compass module HMC6352, which is selected as dead reckoning feedback, has been studied on its datasheet [5]. I2C protocol is introduced on communicating between a digital compass module as slave and a microcontroller as master. In designing a dead reckoning algorithm, the digital compass heading is used as the main feedback in guiding the robot's movement. The odometry data or encoder reading will be collected as reference. The block diagram for dead reckoning is shown in Fig. 6.

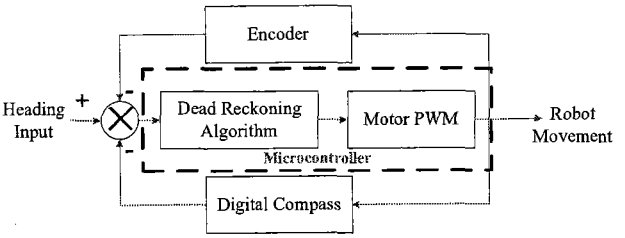

Fig. 6 Block Diagram for Dead Reckoning Algorithm

The concepts to design dead reckoning of straight line travelling and dead reckoning of curving travelling concept are shown in Fig. 7 and Fig. 8 respectively.

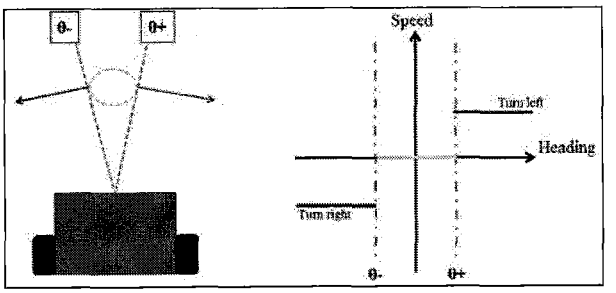

Fig. 7 Dead reckoning for straight line travelling concept

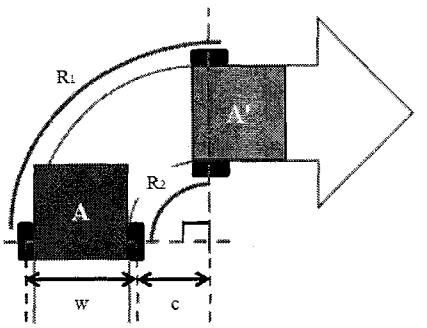

Fig. 8 Dead reckoning for curving travelling concept

3.3.2 PID Line Follower Algorithm

It is necessary to declare that the position error of robot on the white line of the game field in designing a PID line follower algorithm. Nevertheless, the digital sensors are used as the position error detectors in this project [6]. The arrangement of sensors position on the robot is shown in Fig. 9. Based on the arrangement of sensors, the errors can be predicted and described briefly by its condition (see Table 3).

In designing a PID line follower algorithm, there are seven digital sensors used as the position error feedback. The odometry data are collected as reference. The block diagram for PID line follower algorithm is shown in Fig. 10. In Fig. 11, the block diagram was expressed in term of PID controller. In derivation of PID controller algorithm, the full formula of PID controller is expressed as (4).

$$R_n = R_{n-1} + K_P \cdot (e_n - e_{n-1}) + K_I \cdot (e_n - e_{n-1})/2 + K_D \cdot (e_n - 2 \cdot e_{n-1} + e_{n-2}) \quad (4)$$

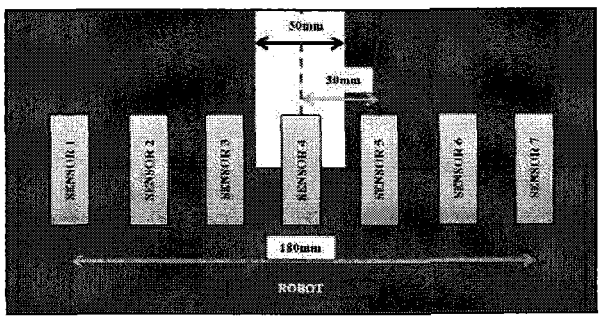

Fig. 9 The arrangement of sensors' position for PID line follower

Table 3 Predictable Digital Sensors Position and Errors

No	Sensor Condition (Sensor1-7)	Error	Detail
1	1000000	-6	Too far right
2	1100000	-5	Slightly too far right
3	0100000	-4	Far right
4	0110000	-3	Slightly far right
5	0010000	-2	To the right
6	0011000	-1	Slightly to the right
7	0001000	0	Middle on the line
8	0001100	1	Slightly to the left
9	0000100	2	To the left
10	0000110	3	Slightly far left
11	0000010	4	Far left
12	0000011	5	Slightly too far left
13	0000001	6	Too far left
14	0000000	-7	Lost from white line and offset to right side
15	0000000	7	Lost from white line and offset to left side
16	1111111	ignore	Robot meet the junction

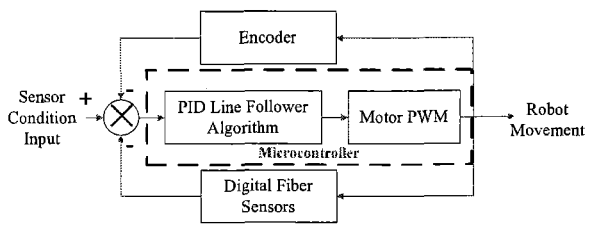

Fig. 1 Block diagram for PID line follower algorithm

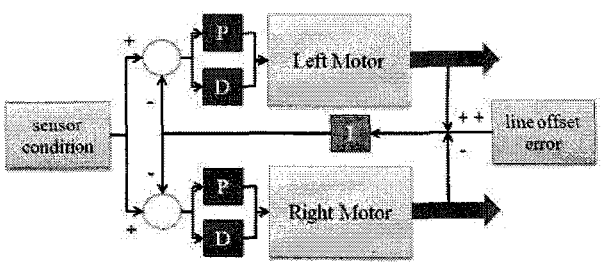

Fig. 2 Block diagram in term of PID algorithm

The method of tuning of the three PID parameters K_P, K_I, and K_D was an important issue, also known as *try and error turning method*. The following guidelines can be used for experimentally finding suitable values [7]:

i. *Select a typical operating setting for the desired speed turns off integral and derivative parts, and then the increase Kp to maximum or until oscillation occurs.*
ii. *If system oscillates, divide Kp by 2.*
iii. *Increase Kd and observe behavior when increasing/decreasing the desired speed by about 5%. Choose a value of Kd which gives a damped response.*
iv. *Slowly increase Ki until oscillation starts. Then divide Ki by 2 or 3.*
v. *Check whether overall controller performance is satisfactorily under typical system conditions.*

3.3.3 Motion Planning System Development

It is declared as the final designing stage in this project. The overview of motion planning system flowchart is shown in Fig. 12.

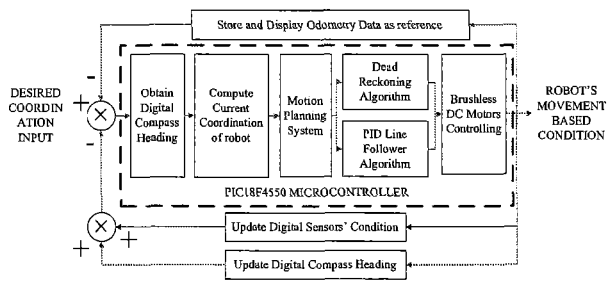

Fig. 3 Motion planning system block diagram

4. Results and Discussion

4.1 Dead Reckoning Algorithm Analysis

In the straight line travelling path observation, the autonomous mobile robot moves a straight line path, from the position A to the position A', with the heading feedback of digital compass. Then, the mobile robot stops for a desired distance which is measured in 1400mm. The travelling path is shown in Fig. 13.

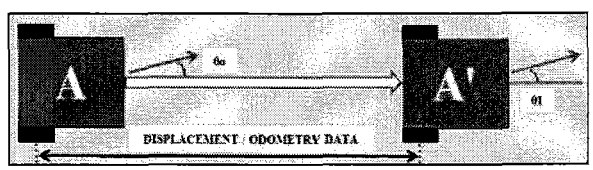

Fig. 4 Straight travelling path of autonomous robot

Afterwards, the odometry data of encoder are collected as the displacement travelled. Besides, the starting and the ending heading value of digital compass are collected in order to compare the efficiency of the dead reckoning algorithm.

The tolerance of the heading offset was approximately equal to ±0.4 degree (see Table 4). The result for offset error between the desired data and measured data would be computed. The maximum offset percentage is 0.6041 %; the minimum offset percentage is -0.5178% (see Table 5).

Table 4 Heading Results of Straight Travelling Path

Case	Starting Heading, θ_0 (°)	Ending Heading, θ_1 (°)	Heading Offset, $\theta_1 - \theta_0$ (°)
1	76.6	77.0	0.4
2	82.2	82.4	0.2
3	80.3	80.1	-0.2
4	80.5	80.7	0.2
5	77.9	77.6	-0.3

Table 5 Odometry Results for Straight Travelling

Case	Desired data, d_1 (pulses)	Measured data, d_2 (pulses)	Offset error, $d_1 - d_2$ (pulses)	Offset error percentage, (%)
1	3476	3482	-6	-0.1726
2	3476	3466	10	0.2876
3	3476	3494	-18	-0.5178
4	3476	3479	-3	-0.0863
5	3476	3455	21	0.6041

In curving travelling section, the robot movement is observed same as Fig. 14. The initial position of robot was located at position A, it starts to move and curve until it comes to the position A'. The mobile robot will stop when it is achieve the 90.0 degree turning to the left under the desired motor speed.

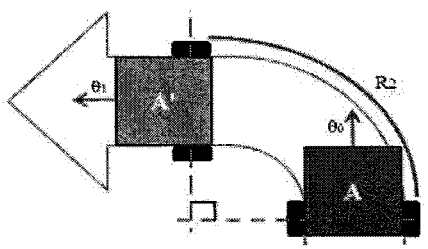

Fig.5 Curve Travelling Path of Autonomous Robot

As the collected result, the maximum heading offset was 0.5°; the minimum heading offset was 0.2° (see Table 6).

Table 6 Heading Results of Curve Travelling Path

Case	Starting Heading, θ_0 (°)	Desired Heading, θ_D (°)	Ending Heading, θ_1 (°)	Heading Offset, $\theta_D - \theta_1$ (°)
1	80.8	350.8	350.5	0.3
2	79.4	349.4	349.2	0.2
3	80.4	350.4	350.3	0.2
4	80.1	350.1	349.6	0.5
5	80.2	350.2	350.0	0.2

In the results analysis, the maximum offset percentage is 6.783%; the minimum offset percentage is 4.814% (see Table 7).

Table 7 Odometry Results for Curve Travelling

Case	Desired data, R_1 (pulses)	Measured data, R_2 (pulses)	Offset error, $R_1 - R_2$ (pulses)	Offset error percentage, (%)
1	5027	4868	341	6.783
2	5027	4774	253	5.033
3	5027	4785	242	4.814
4	5027	4777	250	4.973
5	5027	4782	245	4.874

4.2 PID Line Follower Algorithm Analysis

There were divided into four testing sections, including proportional (P) controller, proportional-integral (PI) controller, proportional-derivative (PD) controller, and proportional-integral-derivative (PID) controller. These PID controlling performances are observed. The method used to collect the PID follower algorithm result is done by capturing video. Afterward, the robot's movement video is converted into frames by using AVS video software. Each converted frame is observed and analyzed based on the predictable error table (see Table 3). Each frame is count in duration 0.1 second. A graph of error versus time will be plotted until the robot stop moving.

In P controller, the P constant parameter will be considered only, where I constant and D constant are set as zero value. For the experiment of this project, there are three values of P are tested, including P = 4, P = 5, and P = 6. The results of robot performances are observed and plotted in the graph of Fig. 15.

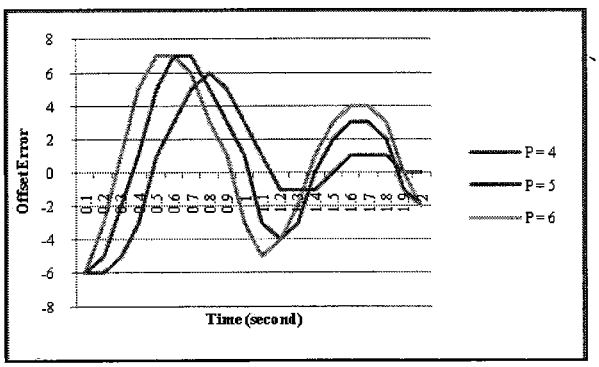

Fig. 6 Robot Responses for Proportional (P) Controller

By increasing the P constant, the robot's response is increased to adjust motor speed to achieve zero error in a shorter time. As well, the overshooting phenomenon of a system is increased too. Therefore, error of steady state is hard to be obtained. This P controller system is an unstable system yet.

The PI controller is implemented in testing the robot performance, where the P and I parameters are used and D parameter is set as zero value. In this project, the P parameter is chosen as value equal to four, since the P parameter does not lead the system to overshoot too much. Nevertheless, I parameters that used in this experiment are I = 0.7, I = 0.8 and I = 0.9. The robot responses for PI controller are shown in Fig. 16.

Since the P parameter in this experiment is set as constant value which is equal to value four. In increasing the Integral parameter, the system's response is slower down, regarding the rise time, but the steady state error has been eliminated. In this experiment, the P parameter and I parameter which is equal to value four ad value 0.9 respectively, is chosen to provide the better performance to the robot.

In Proportional-Derivative (PD) controller, there are only the P parameter and D parameter used to implement. I parameter is ignored in this experiment. Same as previous, the P parameter is set as value four.

Nevertheless, there are three values for D parameter used, including D = 12, D = 18, and D = 24. In this experiment, the robot responses for PD controller are observed and plotted in the graph, as shown in Fig. 17.

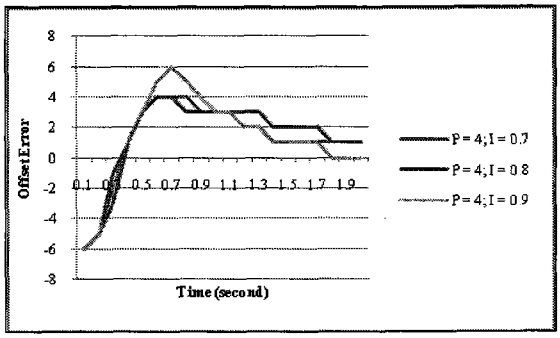

Fig. 16 Robot Responses for Proportional-Integral (PI) Controller

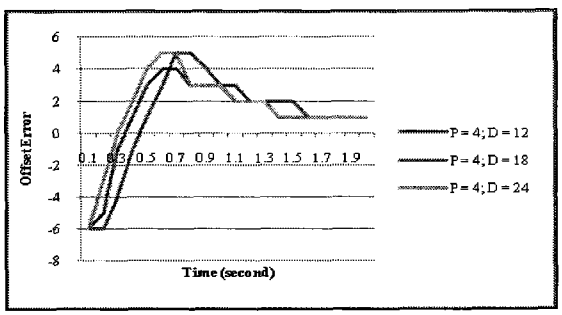

Fig. 7 Robot Responses for Proportional-Derivative (PD) Controller

There is a difference between PD controller and PI controller, the PD controller is able to reach the equilibrium faster than PI controller does; but the steady state error for PD controller system does not solve yet. In this experiment, the value of P and D parameters which provide a better performance to the system, are P = 4 and D = 18 respectively. The D parameter is chosen as 18 rather than 24 due to the lower overshoot of performance.

The full PID controller was used in proposes to combine all the advantages of PI controller and PD controller. There were three set of PID which would be implemented in robot, including SET1 (P = 3; I = 0.9; D = 32), SET2 (P = 4; I = 0.9; D=16), and SET3 (P = 5; I =0.8; D = 18) respectively. These three sets of PID controller on the robot performance are observed and plotted in the graph of Fig. 18.

From PID controller experiment, the SET2 parameters are recommended, since this combination of PID parameters able to provide a better response on a line follower robot. This is because the SET2 of PID controller is able to provide the line follower system a faster response, less overshooting, less oscillation and the zero behaviour of steady state error.

4.3 Motion Planning System Analysis

In Case A Analysis, the autonomous robot, implemented by motion planning system, would able to find out its initial starting coordination. Vividly, the starting coordination, in (x, y) form, was measured as (0, 0). Besides, the initial direction heading of the robot would be measured by using the digital compass module. The motion planning system was implied on the robot in finding the shortest distance to its goal. First, robot had been placed in the game field of coordination (0, 0). It would be prompt in a desired coordination by programming; the desired coordination was set as (-1, 3) in case. The position of robot and its coordination in the game field were demonstrated in Fig. 19. According to the motion planning system concept, the autonomous robot was able to travel in the shortest travelling path in order to reach to the desired coordination. The robot movement result was demonstrated in Fig. 20.

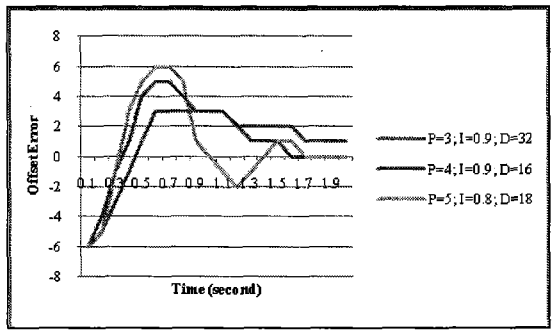

Fig. 188 Robot Responses for Proportional-Integral-Derivative (PID) Controller

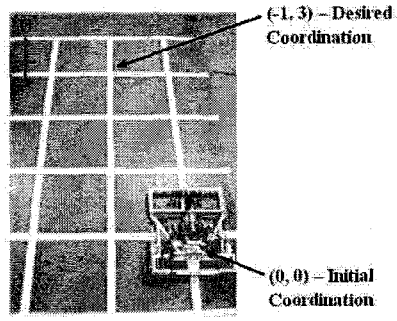

Fig. 9 Robot's Initial Coordination & Desired Coordination of Case A

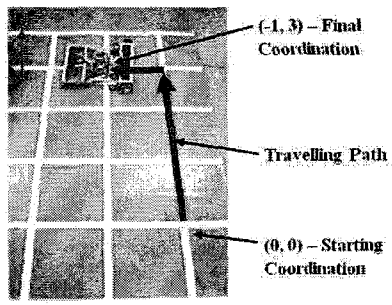

Fig. 20 Robot's Final Coordination & Its Travelling Path of Case A

In Case B, there was the same desired coordination prompt in the system of robot, where the robot would start to travel at the same initial coordination (0, 0). Nevertheless, the initial heading direction of robot had been changed from North to West in this case. There was the demonstration for the explanation shown in Fig.

21. Since the initial heading direction of Case B was differ in comparing with that of the Case A, the robot could navigate to find a new travelling path in order to reach it desired goal. The travelling path of the robot Case B would be demonstrated in Fig. 22.

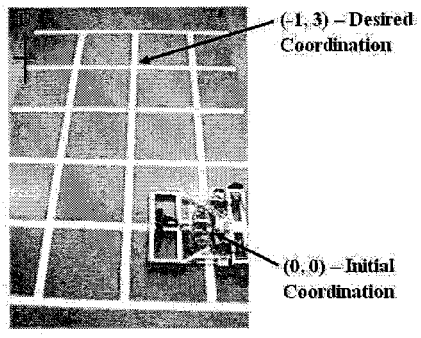

Fig. 21 Robot's Initial Coordination & Desired Coordination of Case B

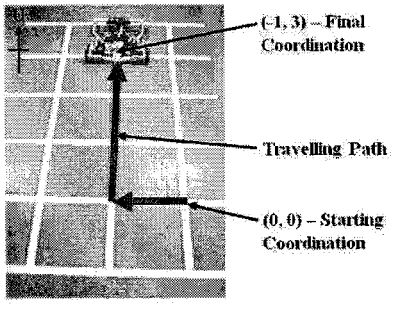

Fig. 10 Robot's Final Coordination & Its Travelling Path of Case B

In the Case C analysis, the motion planning system would be proved that the autonomous mobile robot able to travel by combining the two types of algorithms, including PID line follower algorithm and dead reckoning algorithm. Same as last previous two cases, the autonomous robot had been placed at its initial coordination (0, 0). For this situation, there was a new desired coordination, which was (-2, 4), had been prompt in the motion planning system by programming. The position of robot and its coordination in the game field was demonstrated in Fig. 23. Based on the given situation, the robot's travelling path of Case C was demonstrated in Fig. 24.

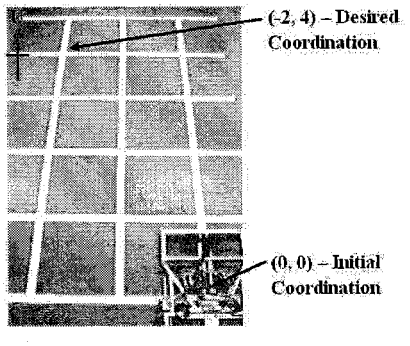

Fig. 11 Robot's Initial Coordination & Desired Coordination of Case C

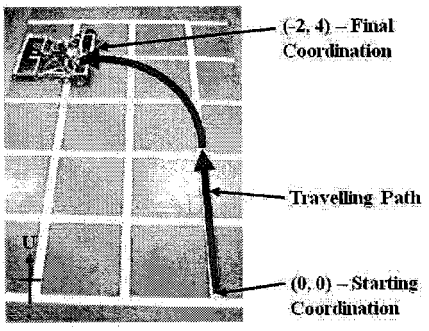

Fig. 24 Robot's Final Coordination & Its Travelling Path of Case C

5. Summary

This project has been carried successfully with all the objectives achieved. Nevertheless, two types of movement algorithms discussed for this project, including, the dead reckoning algorithm and PID line follower algorithm, have been combined under one motion planning system. Therefore, autonomous robot which has been implemented with this motion planning system, able to find and complete its travelling pathway with higher accuracy and less time taken. In other word, the autonomous mobile robot implemented with motion planning system is able to switch and select the suitable types of algorithm in order to reach the goal of target. The motion planning system was controlled by using the microcontroller PIC18F4550.

References

[1] Thomas Coffee. "Waypoint Following and Dead Reckoning", September 2006.
[2] Merl K.Miller, Nelson Winkless, & Joe Bosworth. "The Personal Robot: Navigator", *Conifer, Colorado: Robot Press*, 1998.
[3] Ikalogic.com. Ibrahim Kamal. "WFR, a Dead Reckoning Robot", April 2008.
[4] Qing-Guo Wang, Zhen Ye, Wen-Jian Cai, & Chang-Chieh Hang. "PID Control for Multivariable Processes", *Springer*, Singapore, 2008.
[5] http://www.ssec.honeywell.com/magnetic/hmc6352.html
[6] http://sunx.jp/en/products/fiber/fx-300/index.html
[7] Thomas Braunl. "Embedded Robotics: Mobile Robot Design and Applications with Embedded Systems", *Springer*, Perth, Australia, 2003.