Arduino-based PID control of humidity in closed space by pulse width modulation of AC voltage

SEIICHI NAKAMORI
Professor Emeritus, Faculty of Education
Kagoshima University
1-20-6, Korimoto, Kagoshima, 890-0065
JAPAN

Abstract: My previous work explored a method for constant temperature proportional-integral-derivative (PID) control in a closed space using an Arduino. Similarly, this paper proposes a method of PID control to keep the humidity constant in a closed space using Arduino. PID control by a pulse width modulation (PWM) signal with Arduino is used to keep the humidity constant due to the heat generated by an incandescent light bulb to which an AC voltage of 100 (V) is applied. Here, the humidity is measured using the temperature/humidity sensor DHT11. Specifically, the target value of humidity is set to 30 (%), and the experimental results of P, PD, PI, and PID controls are compared from the points of quick response and stationary performance. As an evaluation of stationary performance, the mean-square value of 1,000 data of the deviation (target value – measured humidity) is calculated, and the PID control is found to be the preferred control.

Key-Words: PID control, constant humidity control, Arduino, temperature/humidity sensor, zero cross-type SSR

Received: May 15, 2021. Revised: February 17, 2022. Accepted: March 12, 2022. Published: April 21, 2022.

1 Introduction

Temperature and humidity are controlled in various places such as egg incubators, baby incubators, rice husk dryers**, etc.** In Latif *et al.* [1], the temperature and humidity measured by the temperature/humidity sensor DHT11 in the baby incubator are controlled by the microcontroller ATmega8535 in the range of 33 (° C) to 35 (° C), humidity 40 (%) to 60 (%). A fan is used to control the humidity, and a heater is used to control the temperature. Matamoros *et al.* [2] use proportional-integral-derivative (PID) control of temperature and humidity in an open-atmosphere system for bioprinting. The PID controller stabilizes the temperature and humidity in the open-atmosphere system at 311 (s) and 65 (s), respectively. In Nugroho *et al.* [3], PID control of chamber temperature is used to obtain test conditions for isolators that separate conductors in power system transmission. The temperature is controlled by an Arduino R3 microcontroller, and the PID controller drives a thermoelectric cooled actuator (TEC12706) based on the temperature measured by the DHT22 sensor. The experimental results show that the chamber temperature is maintained at the target value with proportional coefficient $k_p = 200$, integral coefficient $k_i = 10$ and derivative coefficient $k_d = 5$. The objective of Okpagu *et al.* [4] is to model, design and develop an egg incubator system. This system uses temperature sensor LM35 to measure the condition of the incubator and automatically change the condition to be suitable for the eggs. In this study, light bulbs are used to regulate the temperature of the eggs and water, and control fans are used to ensure humidity and ventilation. Mathematical models of the incubator, actuator and PID controller are developed. The controller design based on the model is validated by Matlab Simulink simulation, and the Zeigler-Nichol tuning method is employed to adjust the controller for temperature control. The parameters of the PID controller are tuned to obtain the desired transient response to the unit step input. Desirable overshoot, rise time, peak time, and settling time are obtained to improve the robustness and stability of the system. In Rahman *et al.* [5], a PID controller is suggested to control the temperature and humidity of an egg incubator. The controller parameters can be adjusted from a PC and an implemented panel (keyboard). The system consists of an Arduino Mega board, humidity sensor, heat sensor, temperature sensor, heat source, humidity source, servo motor, LCD module, and keyboard with ATmega1280 microcontroller. In Veldscholte *et al.* [6], a PID controller-based humidistat is proposed using a microcontroller Arduino Uno, a small proportional solenoid valve, a gas flush bottle and a humidity sensor. The settling time is about 30 (s) and the relative humidity range of 10-95% can be achieved. In Nakamori [3], a PID constant temperature control is proposed to maintain a constant temperature in a closed space of a styrofoam box using Arduino. An incandescent light bulb with an applied voltage of 100 (V) AC is pulse-width modulated by Arduino, and the temperature in the closed space is controlled by PID to be constant. An IC temperature sensor LM35DZ is used to measure the temperature.

Similar to Nakamori [3], in this study, we propose a method of PID control to maintain a constant humidity in the closed space of the styrofoam box. PID control is applied so that the humidity due to heat emitted from an incandescent lamp bulb to which an AC voltage of 100 (V) is applied becomes a constant value with a pulse width modulation (PWM) signal from the Arduino. Here, the humidity is measured using the temperature/humidity sensor DHT11. Specifically, the target value of humidity is set to 30 (%), and the experimental results of P, PD, PI, and PID controls are compared. From the experimental results, the P control ($k_p = 100$), PD control ($k_p = 5$, $k_d = 5$), PD control ($k_p = 5$, $k_d = 1$), PID control ($k_p = 100$, $k_i = 0.8$, $k_d = 5$), and PI control ($k_p = 100$, $k_i = 0.8$) are preferred in order of quick response. As an evaluation of the steady-state characteristics, the average of the squares of 1,000 data of the deviation (target value - measured humidity) is calculated. As a result, the performance of steady-state characteristics is preferable in the order of PID control ($k_p = 100$, $k_i = 0.8$, $k_d = 5$), PI control ($k_p = 100$, $k_i = 0.8$), P control ($k_p = 100$), PID control ($k_p = 100$, $k_i = 5$, $k_d = 5$), PD control ($k_p = 5$, $k_d = 5$), and PD control ($k_p = 5$, $k_d = 1$). Here, k_p, k_i and k_d represent the PID parameters as shown in (1).

2 Block Diagram of Feedback Control for Constant Humidity Adjustment

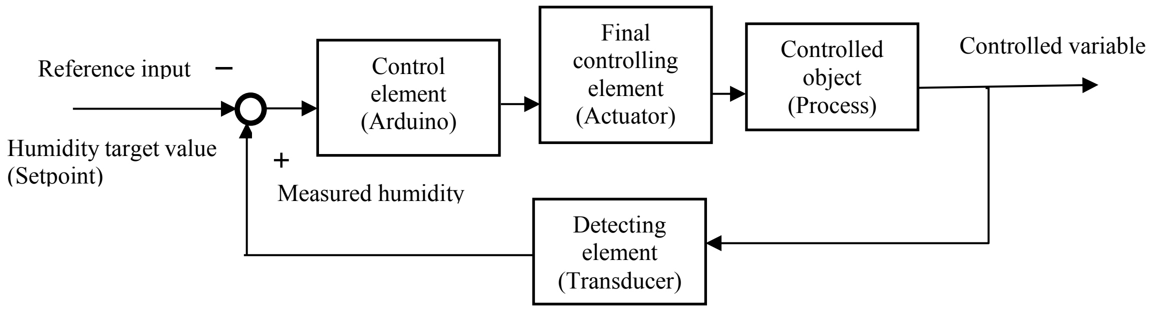

Fig.1 Block diagram of the constant humidity feedback control system.

The purpose of the control in this paper is to perform PID control so that the humidity in the closed space is as close as possible to the target value of humidity. The proportional, integral, and derivative coefficients of the PID controller are adjusted appropriately based on experiments. The overall control function is given by

$$\mathrm{u(t)} = k_p\left(e(t) + \frac{1}{T_i}\int_0^t e(t)dt + T_d\frac{de(t)}{dt}\right).$$

k_p: proportional gain; T_i: integral time;

T_d: derivative time;

u(t): control signal;

$$e(t) = -\big(Refrence\ input - Measured\ humidity(t)\big) = -Deviation(t).$$

In constant control of temperature in the closed space, the temperature increases when an incandescent light bulb is heated. Usually, deviation is given by “target value - measured temperature(t)”. In the case of humidity control, when the incandescent light bulb is heated, the humidity in the closed space decreases. For this reason, the amount of – deviation(t) is input to the control element.

Section 3 presents PID control experiments for constant humidity.

3 PID Control Experiments for Constant Humidity Adjustments

Fig.2 illustrates the circuit diagram of the constant humidity control using the temperature/humidity sensor DHT11 with an incandescent light bulb as the heat source. The outline of the operations is as follows.

(1) PWM pulses are output from port 9 of the Arduino to the Base pin of transistor NPN 2SC1815. During the PWM control signal is 5 [V], electric current flows through the Phototriac and a trigger pulse is generated from the Phototriac.

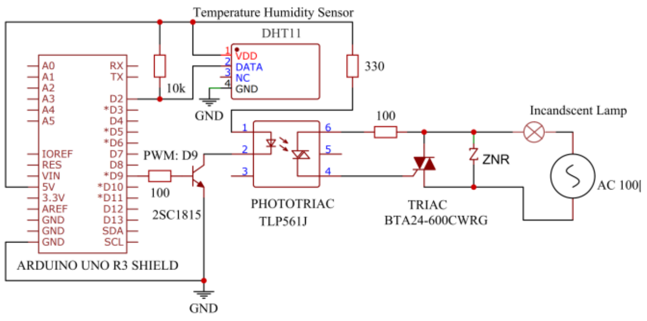

Fig.2 Circuit diagram for constant humidity control.

(2) The DHT11 temperature/humidity sensor converts heat from a 40 (W) incandescent light bulb into a DC voltage. The sensitivity of DHT11 at 25(°C) is ±5(%) (RH) for humidity and ±2(°C) for temperature.

(3) The Phototriac and Triac function as zero-crossing solid-state relay (SSR). Following zero-cross switching of the SSR, an AC voltage with a zero-crossing waveform is supplied to incandescent light bulb.

(4) The humidity value measured by the temperature/humidity sensor is compared with the reference input level, and the Arduino operates the P, PD, PI, and PID controllers so that the humidity value approaches the reference input level. If the PWM value is negative, it is set to 0; if the value exceeds 255, it is set to 255.

Fig.3 shows the inside of the closed space of a styrofoam box for constant humidity control. There are DHT11 temperature/humidity sensor and a 40(W) incandescent light bulb. The depth, width, height, and thickness inside the styrofoam box are 145 (mm), 350 (mm), 280 (mm), and 25 (mm). Styrofoam belongs to the insulation category. A material whose thermal conductivity does not exceed 0.05 (W/m-K) is called Here, $\mathrm{e(k)} = \mathrm{M(k)} - \mathrm{R(k)}$ (see the block diagram of Fig.1), R(k): the desired humidity at discrete time $t = \mathrm{k}\Delta$, $\Delta t = \Delta$, M(k): the measured humidity at

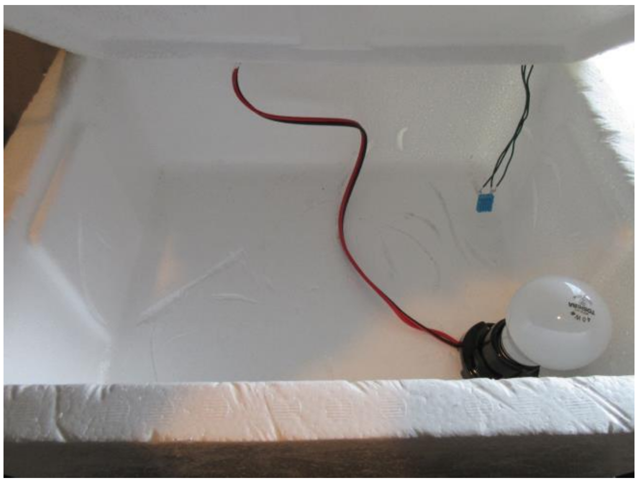

Fig. 3 Internal layout of a closed space in a constant humidity control experiment. There are DHT11 temperature/humidity sensor and a 40(W) incandescent light bulb.

a heat insulator. The constant value PID control program for humidity with the temperature/humidity sensor DHT11 is shown later. In this program, the control input is of the form

$$\mathrm{u(k)} = k_p e(k) + k_i \sum_{i=1}^{k} e(i)\Delta t + k_d \frac{e(k) - e(k-1)}{\Delta t}. \quad (1)$$

discrete time $t = \mathrm{k}\Delta$. Usually, the deviation is given by $R(k) - M(k)$, but in this paper e(k) is set to $-(R(k) - M(k))$, as shown in the block diagram. The justification for this can be explained as follows. In the constant temperature control by Nakamori [3], increasing the PWM value of the Arduino causes the temperature of the incandescent light bulb to rise and the temperature in the closed space to increase. In the constant humidity control, increasing the PWM value of the Arduino increases the temperature of the incandescent light bulb, while decreasing the humidity in the closed space.

In this experiment, the PID parameters k_p, k_i, and k_d are adjusted based on the deviation. The constant value PID control program for humidity is shown in Program listing for PID constant control for humidity. In the case of PID control of temperature in Nakamori [3], the more the incandescent light bulb is heat, the higher the temperature of the closed space. In the case of PID control of humidity, the more the incandescent light bulb is heated, the lower the humidity. Thus, in the humidity PID control program, the line of italicized bold face is different from the temperature PID control program. In the PID program, dt represents Δ. Its value is 1.01 (s). the value of dt is calculated in the program by "dt=(micros()-pretime)/1000000;".

Program listing for PID constant control for humidity.

```
#include <dht.h> //inclusion of header file “dht.h”
dht DHT;
#define DHT11_PIN 2
const float moku=30; //desired humidity 30 [%]
// kp=1; kp=5; kp=100
const float kp=100;
// ki=0.8; ki=5
const float ki=0.8;
// kd=1; kd=5
const float kd=5.0;
float dt;
float d;
float pretime;
float P,I,D;
float PreP = 0;
int co;
int pre;
float PID_error = 0;
long PID_p = 0;
int i;
void setup() {
Serial.begin(9600);
pinMode(9,OUTPUT);
co=10;
}
void loop() {
int chk = DHT.read11(DHT11_PIN);
analogWrite(9,co);
d=float(DHT.humidity);
dt=(micros()-pretime)/1000000;
//Serial.print("dt=");
//Serial.println(dt);
pretime=micros();
PID_error=d-moku;
I+=PID_error*dt;D=(PID_error-PreP)/dt;
PreP=PID_error;
// PID controller:
co = kp*PID_error+kd*D+ki*I;
// P controller:
//co = kp*PID_error;
// PD controller:
//co = kp*PID_error+kd*D;
// PI controller:
//co = kp*PID_error+ki*I;
// I controller:
//co = ki*I;
delay(500);
if (co <0) {
co=0;
}
else if (co >255) {
co=255;
}
else {
co=co;
}
Serial.print("PWM value=");
Serial.println(co);
Serial.print("humidity=");
Serial.println(d);
analogWrite(9,co);
delay(500);
}
```

Fig.4 illustrates the humidity (%) vs. discrete time (k) with the P controller for $k_p = 100$. After $k = 88$ the humidity repeats the values of 30 (%) and 31 (%). Fig.5 illustrates the humidity (%) vs. k with the P controller for $k_p = 1$. After $k = 294$ the humidity repeats the values of 30 (%) and 31 (%). In view of quick response, P control for $k_p = 100$ is superior to that for $k_p = 1$. Fig.6 illustrates the PWM control signal vs. k by the P controller for $k_p = 1$ and $k_p = 100$. In Fig.6, the PWM control signal takes the value of 100 or 0 after $k = 100$ when $k_p = 1$. For $k_p =$

100, the PWM control signal has a value of 9 at 0 (s) and converges to 1 after 500 (s). Fig.7 illustrates the humidity (%) with the PD control vs. k when $k_p = 5$ and $k_d = 1$. At $k = 117$, the humidity is 31 (%), followed by repeating values of 31 (%) and 30 (%). Fig.8 illustrates the PWM control signal vs. k by the PD controller for $k_p = 5$ and $k_d = 1$. At $k = 118$, the PWM value is 5 and then repeats between 5 and 0. Fig.9 illustrates the humidity (%) vs. k by the PD controller for $k_p = 5$ and $k_d = 5$. At $k = 111$, the humidity is 31 (%), and then the values of 31 (%) and 30 (%) are repeated. Fig.10 illustrates the PWM control signal vs. k by the PD controller for $k_p = 5$ and $k_d = 5$. At $k = 95$, the PWM value is 5, and then 5, 0, and 9 repeat as the main values. Fig.11 illustrates the humidity (%) vs. k by the PI controller for $k_p = 100$, $k_i = 0.8$ and $k_p = 100$, $k_i = 5$. For $k_p = 100$ and $k_i = 5$, the humidity values vary widely up and down from 30 (%) and does not converge near the target value 30 (%) even after a period of time. For $k_p = 100$ and $k_i = 0.8$, at $k = 894$, the humidity is 30 (%) from 31 (%) one second earlier, and then it takes values of 30 (%), 31 (%), and 29 (%) randomly. Fig.12 illustrates the PWM control signal vs. k by the PI controller for $k_p = 100$, $k_i = 0.8$ and $k_p = 100$, $k_i = 5$. For $k_p = 100$ and $k_i = 5$, the PWM signal randomly takes the values 0 and 255. The maximum value of PWM for $k_p = 100$, $k_i = 0.8$ is 113 at $k = 877$ and the minimum value is 0. The PWM value changes randomly with time. Fig.13 illustrates the humidity (%) vs. k with the PID controller for $k_p = 100$, $k_i = 0.8$, $k_d = 5$ and $k_p = 100$, $k_i = 5$, $k_d = 5$. For $k_p = 100$, $k_i = 0.8$, $k_d = 5$, the humidity is 31 (%) at $k = 548$ and changes randomly to 31 (%), 30 (%), and 29 (%) thereafter. For $k_p = 100$, $k_i = 5$, $k_d = 5$, the humidity swing is almost as large as for $k_p = 100$, $k_i = 5$ in the PI control. Fig.14 illustrates the PWM control signal vs. k by the PID controller for $k_p = 100$, $k_i = 0.8$, $k_d = 5$ and $k_p = 100$, $k_i = 5$, $k_d = 5$. The PWM control signal waveform for $k_p = 100$, $k_i = 0.8$, $k_d = 5$ is almost the same as that for PI control for $k_p = 100$, $k_i = 0.8$. The PWM control signal waveform for $k_p = 100$, $k_i = 5$, $k_d = 5$ is similar to that of the PI control for $k_p = 100$, $k_i = 5$. From the above experimental results, the P control ($k_p = 100$), PD control ($k_p = 5$, $k_d = 5$), PD control ($k_p = 5$, $k_d = 1$), PID control ($k_p = 100$, $k_i = 0.8$, $k_d = 5$), and PI control ($k_p = 100$, $k_i = 0.8$) are preferred in order of quick response.

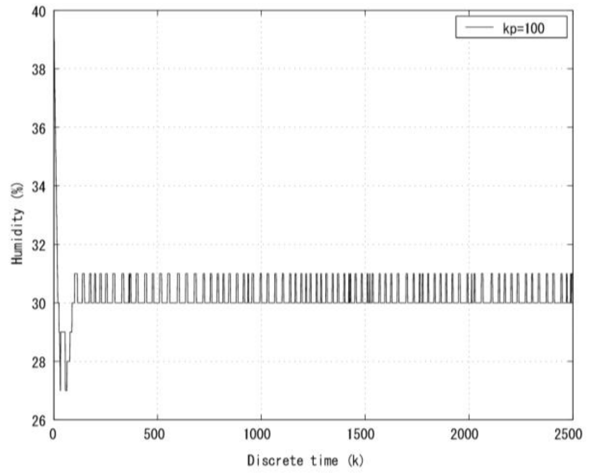

Fig.4 Humidity (%) vs. discrete time (k) by P controller for $k_p = 100$.

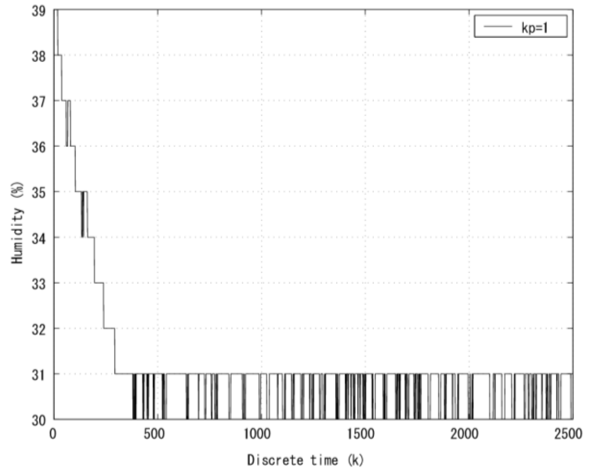

Fig.5 Humidity (%) vs. discrete time (k) by P controller for $k_p = 1$.

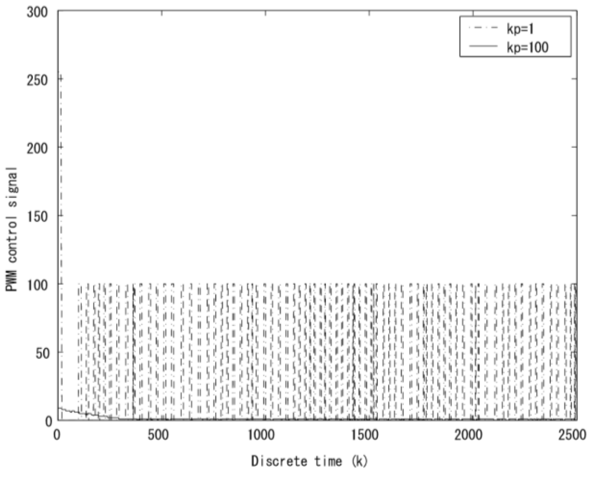

Fig.6 PWM control signal vs. discrete time (k) by P controller for $k_p = 100$ and $k_p = 1$.

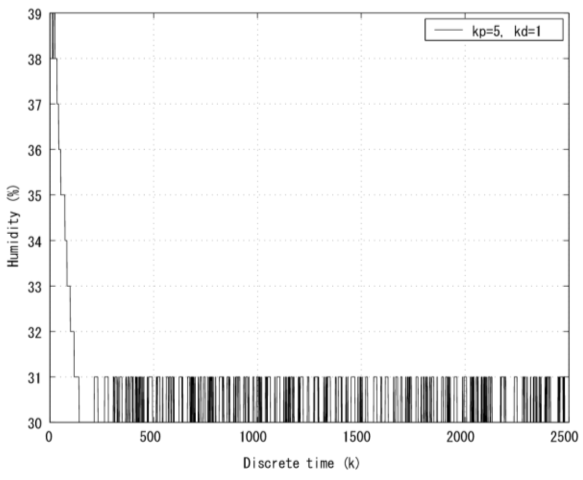

Fig.7 Humidity (%) vs. discrete time (k) by PD controller for $k_p = 5$ and $k_d = 1$.

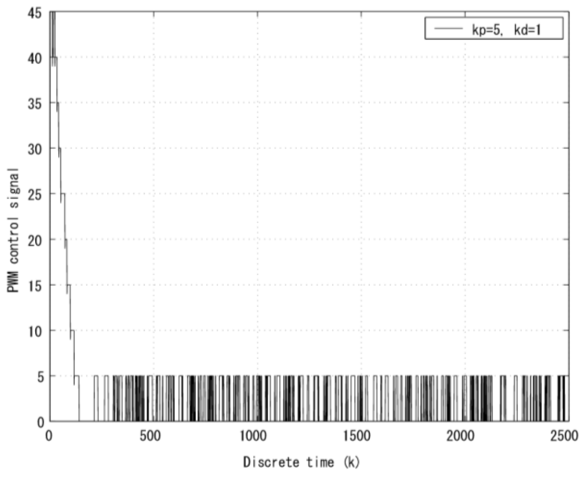

Fig.8 PWM control signal vs. discrete time (k) by PD controller for $k_p = 5$ and $k_d = 1$.

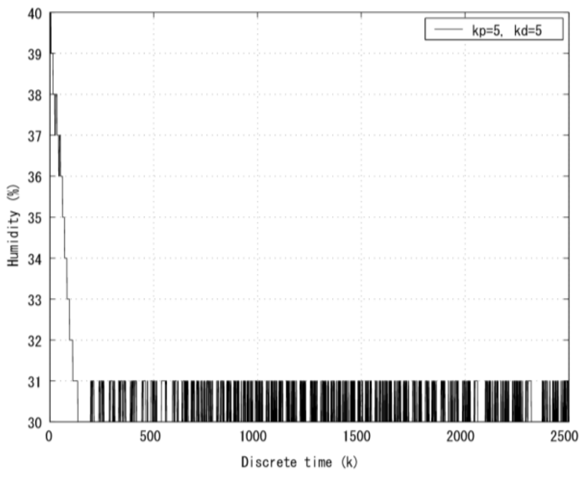

Fig.9 Humidity (%) vs. discrete time (k) by PD controller for $k_p = 5$ and $k_d = 5$.

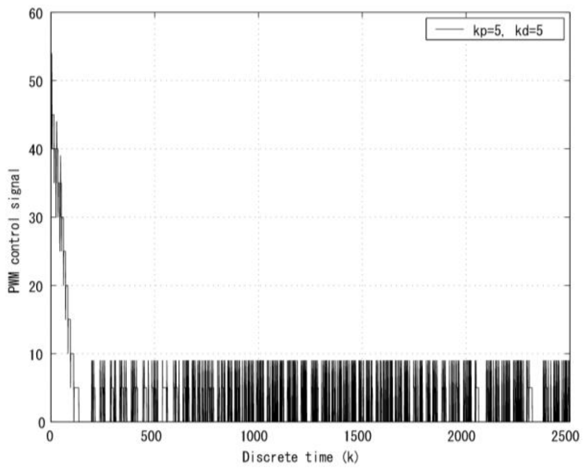

Fig.10 PWM control signal vs. discrete time (k) by PD controller for $k_p = 5$ and $k_d = 5$.

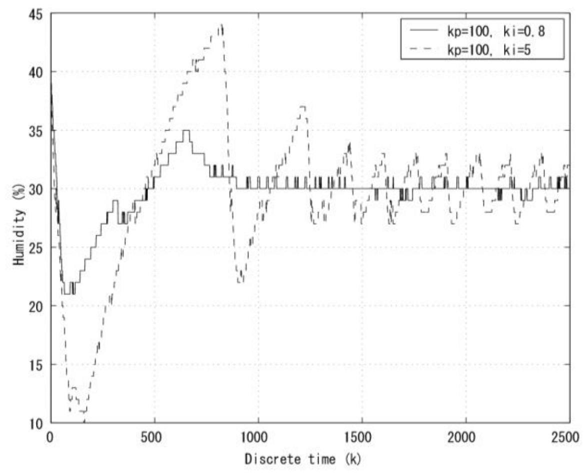

Fig.11 Humidity (%) vs. discrete time (k) by PI controller for $k_p = 100$, $k_i = 0.8$ and $k_p = 100$, $k_i = 5$.

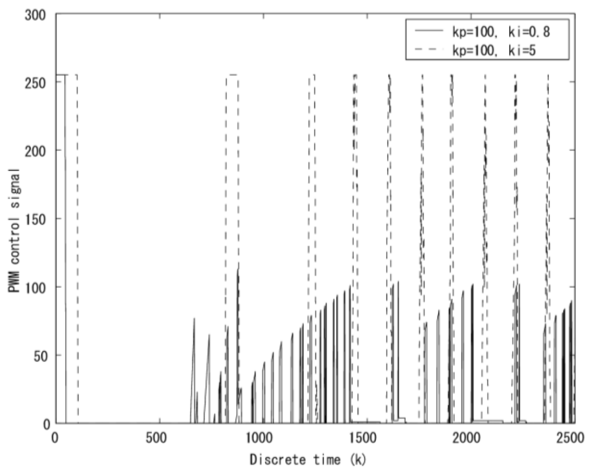

Fig.12 PWM control signal vs. discrete time (k) by PI controller for $k_p = 100$, $k_i = 0.8$ and $k_p = 100$, $k_i = 5$.

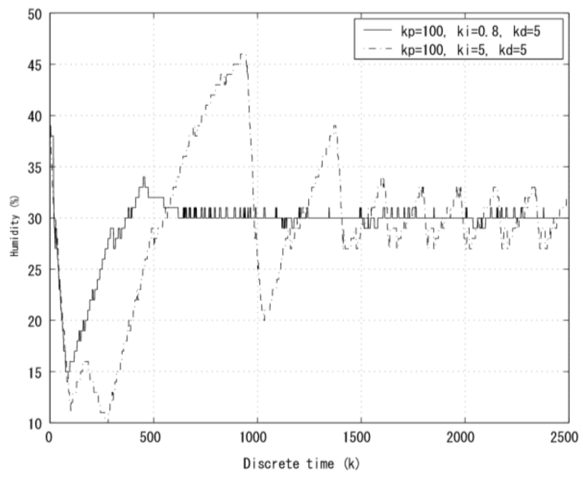

Fig.13 Humidity (%) vs. discrete time (k) by PID controller for $k_p = 100, k_i = 0.8, k_d = 5$ and $k_p = 100, k_i = 5, k_d = 5$.

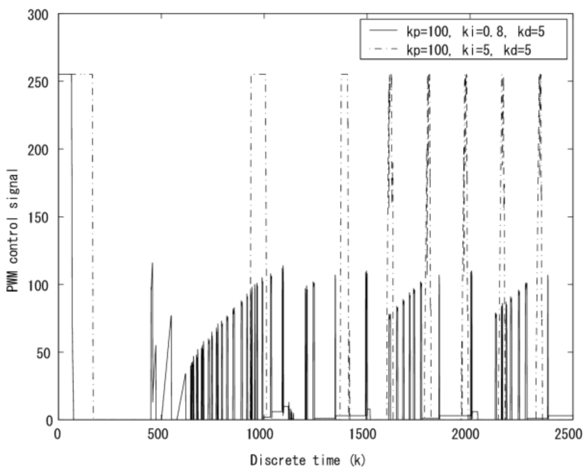

Fig.14 PWM control signal vs. discrete time (k) by PID controller for $k_p = 100, k_i = 0.8, k_d = 5$ and $k_p = 100, k_i = 5, k_d = 5$.

Table 1 shows the mean-square values (MSVs) of deviations for P, PD, PI and PID controllers. The MSV value is calculated by $\frac{1}{1000}\sum_{k=1501}^{2500}(Measured\ humidity(k) - Humidity\ target\ value)^2$. Here, $Humidity\ target\ value = 30$ (%). To investigate the steady-state performance of the PID control, the MSV is calculated using the humidity data from discrete time $k = 1501$ to $k = 2500$. From the MSVs, the performance of steady-state characteristics is preferable in the order of PID control ($k_p = 100$, $k_i = 0.8$, $k_d = 5$), PI control ($k_p = 100$, $k_i = 0.8$), P control ($k_p = 100$), PID control ($k_p = 100$, $k_i = 5$, $k_d = 5$), PD control ($k_p = 5, k_d = 5$), and PD control ($k_p = 5, k_d = 1$).

Table 1 Mean-square values of deviations for P, PD, PI and PID controllers. Mean-square value is calculated by $\frac{1}{1000}\sum_{k=1501}^{2500}(Measured\ humidity(k) - Humidity\ target\ value)^2$, $Humidity\ target\ value =$ 30 (%).

Types of PID control	PID parameters	Mean-square values of deviations
P control	$k_p = 100$	0.2070
	$k_p = 1$	0.7660
PD control	$k_p = 5, k_d = 1$	0.3320
	$k_p = 5, k_d = 5$	0.3180
PI control	$k_p = 100, k_i = 0.8$	0.1990
	$k_p = 100, k_i = 5$	3.2090
PID control	$k_p = 100, k_i = 0.8, k_d = 5$	0.1620
	$k_p = 100, k_i = 5, k_d = 5$	3.1530

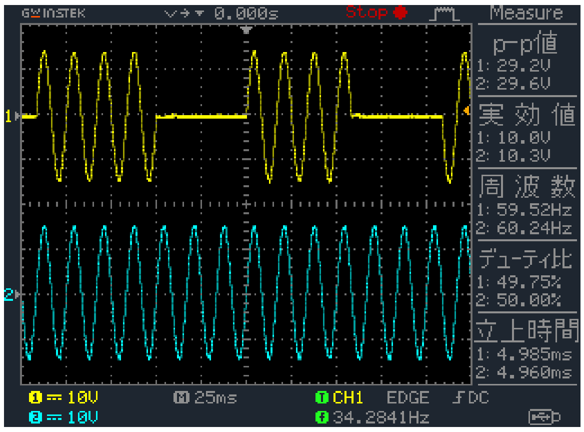

Fig.15 Pulse width modulated waveform (upper figure), for the PWM value 128, of the AC voltage waveform (lower figure) [3].

Fig.15 shows the PWM waveform of the AC voltage by the zero-crossing SSR when the PWM value is 128. Fig.15 shows the root-mean-square (RMS) and peak-to-peak 1/10 values of the AC voltage measured with the channel 1 and 2 probes, respectively.

4 Conclusions

This paper proposed an Arduino-based constant-humidity PID control method with the incandescent light bulb as a heat source in the closed space of the styrofoam box, using the humidity measured by a temperature/humidity sensor DHT11. In PID constant value control of humidity, the PWM signal output from the Arduino is adjusted to the AC voltage applied to the incandescent light bulb through the

NPN transistor and zero-crossing type SSR so that the humidity approaches the target value. From the above experimental results, P control ($k_p = 100$), PD control ($k_p = 5$, $k_d = 5$), PD control ($k_p = 5$, $k_d = 1$), PID control ($k_p = 100$, $k_i = 0.8$, $k_d = 5$), and PI control ($k_p = 100$, $k_i = 0.8$) are preferred in order of quick response. From the MSV, the performance of steady-state characteristics is preferable in the order of PID control ($k_p = 100$, $k_i = 0.8$, $k_d = 5$), PI control ($k_p = 100$, $k_i = 0.8$), P control ($k_p = 100$), PID control ($k_p = 100$, $k_i = 5$, $k_d = 5$), PD control ($k_p = 5$, $k_d = 5$), and PD control ($k_p = 5$, $k_d = 1$).

References:

[1] A. Latif, H. A. Widodo, R. A. Atmoko, T. N. Phong, and E. T.Helmy, Temperature and Humidity Controlling System for Baby Incubator, *Journal of Robotics and Control (JRC)*, Vol.2, No.3, 2021, pp. 190-193. doi: 10.18196/jrc.2376.

[2] M. Matamoros *et al.*, Temperature and Humidity PID Controller for a Bioprinter Atmospheric Enclosure system, *Micromachines*, Vol.11, No.11, 2020, pp. 1-15. doi: 10.3390/mi11110999.

[3] S. Nakamori, Arduino-based PID Control of Temperature in Closed Space by Pulse Width Modulation of AC Voltage, *International Journal of Computer and Systems Engineering*, Vol.1, No.2, 2021, pp. 1–18.

[4] A. Nugroho, T. Andromeda, and M. A. Riyadi, PID Controller Implementation for Temperature Control in Leakage Current Test Chamber 20kv Insulator, *International Journal of Advanced Engineering Research and Science*, Vol.8, No.2, 2021, pp.106–112. doi: 10.22161/ijaers.82.14.

[5] P. E. Okpagu and A. W. Nwosu, Development and Temperature Control of Smart Egg Incubator System for Various Types of Egg, *European Journal of Engineering and Technology*, Vol. 4, No. 2, pp. 13–21, 2016.

[6] Z.-A. S. A. Rahman and F. S. A. Hussain, Smart Incubator based on PID Controller, *International Research Journal of Engineering and Technology*, Vol.4, No.3, 2017, pp. 2501–2509.

[7] L. B. Veldscholte, R. J. Horst, and S. de Beer, Design, Construction, and Testing of an Accurate Low-Cost Humidistat for Laboratory-Scale Applications, *Eur. Phys. J. E*, Vol.44, No.4, 2021, Article no.48, pp.1-8. doi: 10.1140/epje/s10189-021-00062-5.

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

The author contributed in the present research, at all stages from the formulation of the problem to the final findings and solution.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

No funding was received for conducting this study.

Conflict of Interest

The author has no conflict of interest to declare that is relevant to the content of this article.