**Logo

Description automatically generated**

**San Francisco Bay University**

**CE450L Lab - Embedded Engineering Lab**

**2023 Summer Final Exam**

**Student Name: Melvin Divine Pritchard ID: 19857**

1. In feedback control system design, PID (Proportional-Integral-Derivative) is a loop feedback control mechanism. For example, in the temperature control system, when we setup *T=100 °C*, and measure the temperature < *100 °C* from the sensor, then the heating control signal will be generated, and vice versa. If we want the final temperature to be stable at *100 °C*, PID control method should be taken.

Assuming that if the setup temperature is , and measured value , then the error at . After one unit delay sampling, the measured value will be changed to , and the new error at , then the heating signal value at . Again after 1 more unit delay sampling, the measured value is , the error at . In terms of PID control algorithm, the incremental/decremental heating control signal ’s value can be calculated by the following equation.

where

and the new heating signal value at should be computed by

Based on the calculation in the above example, the general algorithm implementation can be shown as follows

Assuming that the sampling rate, this is, one unit delay is 0.2 second, and real measured temperature function vs time is following the function , please write the python program to find the heating control signal by using at each sampling step from until

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Time(Sec)** | **t = 0** | **t = 0.2** | **t = 0.4** | **t = 0.6** | **t = 0.8** | **t = 1.0** | **…** | **t = 3.8** | **t = 4** |
| **Meas.Temp(°C)** | 70 | 70.65 | 71.23 | ? | ? | ? | **…** | ? | ? |
| **Error** | **𝜀(0)** | **𝜀(1)** | **𝜀(2)** | **𝜀(3)** | **𝜀(4)** | **𝜀(5)** | **…** | **𝜀(19)** | **𝜀(20)** |
| 100-70  =30 | 100-70.65  =29.35 | 100-71.23  =28.77 | ? | ? | ? | **…** | ? | ? |
| **∆** |  |  | **∆** | **∆** | **∆** | **∆** | **…** | **∆** | **∆** |
|  |  | **139.32** | ? | ? | ? | **…** | ? | ? |
| **Ctrl Signal** |  |  |  |  |  |  | **…** |  |  |
|  | **120** | 120+139.32  =**259.32** | ? | ? | ? | **…** | ? | ? |

**Program**

import math

def calculate\_temperature(t):

tau = 10

return 100 \* (1 - math.exp(-(t + 12.06) / tau))

def calculate\_delta\_u(Kp, Ki, Kd, errors):

delta\_u\_values = []

for i in range(2, len(errors)):

delta\_u = Kp \* (errors[i] - errors[i - 1]) + Ki \* errors[i] + Kd \* (errors[i] - 2 \* errors[i - 1] + errors[i - 2])

delta\_u\_values.append(delta\_u)

return delta\_u\_values

def calculate\_control\_signal(initial\_u, delta\_u\_values):

control\_signals = [initial\_u]

for delta\_u in delta\_u\_values:

new\_u = control\_signals[-1] + delta\_u

control\_signals.append(new\_u)

return control\_signals

def main():

Kp = 8

Ki = 5

Kd = 1.6

initial\_u = 120

sampling\_rate = 0.2

num\_samples = 21

errors = [30, 29.35] # Given errors for t = 0 and t = 0.2

for i in range(2, num\_samples):

t = i \* sampling\_rate

measured\_temp = calculate\_temperature(t)

error = 100 - measured\_temp

errors.append(error)

delta\_u\_values = calculate\_delta\_u(Kp, Ki, Kd, errors)

control\_signals = calculate\_control\_signal(initial\_u, delta\_u\_values)

for i in range(num\_samples - 2):

t = (i + 2) \* sampling\_rate

print(f"t = {t:.1f}, Measured Temp = {calculate\_temperature(t):.2f} °C, Error = {errors[i + 2]:.2f}, ∆u = {delta\_u\_values[i]:.2f}, Control Signal u = {control\_signals[i + 1]:.2f}")

if \_\_name\_\_ == "\_\_main\_\_":

main()

1. Suppose *x* is an array of integers, and we have just executed this code:

*for(int i=0; i<5; i++)*

*x[i] = i+1;*

if *x[0]* is stored at address 0x4500, what is the value of each of the following expressions? Explain why.

(a) x : 0x4500; The expression x itself is a pointer to the first element of the array. Since x[0] is stored at address 0x4500, the value of the pointer x will be 0x4500, which is the address of the first element.

(b) *&x[0]: =* 0x4500; The expression &x[0] gives the address of the 0th index of the array. Given that x[0] is stored at address 0x4500, the address of x[0] is also 0x4500.

(c) *\*x:* \*x = 1; The expression \*x is equivalent to x[0], which holds the value 1. So, \*x will have the value 1.

(d) *x[1]: =* 2; The expression x[1] holds the value at the 1st index, which is 2. The loop assigned i + 1 to x[i], and for i = 1, x[1] is assigned the value 2.

(e) *&x[1]:* = 0x4504; Since x[0] is stored at address 0x4500, and each int element takes up 4 bytes (assuming the standard size of an int), the address of x[1] can be calculated as 0x4500 + 4 = 0x4504.

1. Write a program in any computer language to display prime number pair (difference is 2 between them, such as 5 7 and 17 19)

*Input from keyboard:*

*Enter a number less than 10000: 100*

*Output:*

*All pairs within input range:*

*3 5*

*5 7*

*11 13*

*17 19*

*29 31*

*41 43*

*59 61*

*71 73*

**Program**

# Online Python - IDE, Editor, Compiler, Interpreter

def is\_prime(num):

if num <= 1:

return False

if num <= 3:

return True

if num % 2 == 0 or num % 3 == 0:

return False

i = 5

while i \* i <= num:

if num % i == 0 or num % (i + 2) == 0:

return False

i += 6

return True

def find\_prime\_pairs(limit):

prime\_pairs = []

for num in range(3, limit - 1):

if is\_prime(num) and is\_prime(num + 2):

prime\_pairs.append((num, num + 2))

return prime\_pairs

if \_\_name\_\_ == "\_\_main\_\_":

limit = int(input("Enter a number less than 750: "))

prime\_pairs = find\_prime\_pairs(limit)

print("All pairs within input range:")

for pair in prime\_pairs:

print(f"{pair[0]} {pair[1]}")

Running result

A screenshot of a computer game

Description automatically generated