Lucas de Aguilar Junqueira Campos
Matheus Barcelos de Oliveira

Trabalho Prático

Belo Horizonte
04 de dezembro de 2015

Sumário

Lista de ilustrações 2

1 INTRODUÇÃO 3

2 METODOLOGIA 4
2.1 Análise do Circuito 4
2.2 Função de Transferência 4
2.3 Projeto de Controlador por Alocação de Polos 4
2.4 Função de Transferência da Malha Fechada 5
2.5 Projeto controlador PI 5
2.6 Projeto controlador Dahlin 6

3 RESULTADOS 7
3.1 Resposta do sistema à uma entrada em degrau 7
3.2 Resposta pulsada do sistema ao controlador por alocação de polos 8
3.3 Controlador PI 9
3.4 Controlador Dahlin 11

APÊNDICE A – CONTROLADOR PROPORCIONAL 12

APÊNDICE B – CONTROLADOR PROPORCIONAL INTEGRATIVO 14

APÊNDICE C – CONTROLADOR DAHLIN 16

Lista de ilustrações

Figura 1 – Circuito objetivo 3
Figura 2 – Resultado da simulação do sistema em malha aberta 7
Figura 3 – Comparação resultado experimental e teórico 8
Figura 4 – Comparação resultado experimental e teórico para controlador por alocação de polos digital 9
Figura 5 – Resultado da simulação do controlador PI x demais sistemas 10
Figura 6 – Comparação entre resultado teórico e experimental para o controlador PI 10
Figura 7 – Resultado da simulação do sistema utilizando o controlador de Dahlin 11

1 Introdução

O presente trabalho tem como objetivo aplicar teorias de controle desenvolvidas na disciplina de Controle Digital de Sistemas Dinâmicos para analisar o determinado circuito:

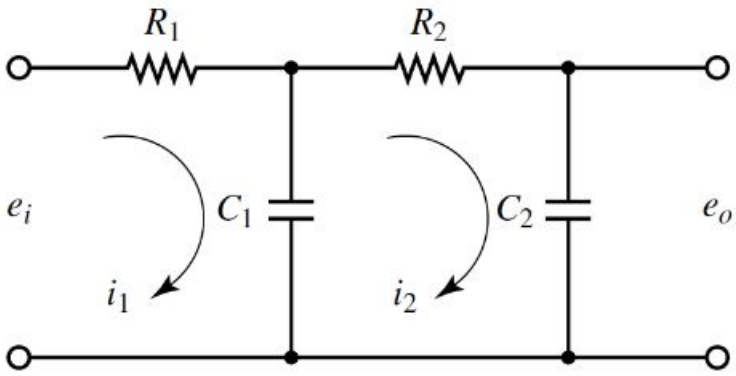

Figura 1 – Circuito objetivo

No circuito acima foram utilizados capacitores de 220 nF e resistências de 100KΩ . Além disso são implementados três diferentes controladores, um controlador Proporcional, um controlador Proporcional Integrativo e um controlador Dahlin.

2 Metodologia

2.1 Análise do Circuito

A partir do circuito da figura 1, foram obtidos as seguintes equações diferenciais através das leis dos nós e das malhas de Kirchhoff:

$$\frac{dV_{C_1}}{dt} = -\frac{R_1 + R_2}{C_1 R_1^2} V_{C_1} + \frac{1}{C_1 R_1} V_{C_2} + \frac{R_2}{C_1 R_1^2} e_i \tag{2.1}$$

$$\frac{dV_{C_2}}{dt} = \frac{1}{C_2 R_2} V_{C_1} - \frac{1}{C_2 R_2} V_{C_2} \tag{2.2}$$

$$e_o = V_{C_2} \tag{2.3}$$

A partir das equações foi obtido a seguinte representação em espaço de estados:

$$\begin{bmatrix} \dot{V_{C_1}} \\ \dot{V_{C_2}} \end{bmatrix} = \begin{bmatrix} -\frac{R_1+R_2}{C_1 R_1^2} & \frac{1}{C_1 R_1} \\ \frac{1}{C_2 R_2} & -\frac{1}{C_2 R_2} \end{bmatrix} \begin{bmatrix} V_{C_1} \\ V_{C_2} \end{bmatrix} + \begin{bmatrix} \frac{R_2}{C_1 R_1^2} \\ 0 \end{bmatrix} e_i \tag{2.4}$$

$$e_o = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} V_{C_1} \\ V_{C_2} \end{bmatrix} + [0]\, e_i \tag{2.5}$$

2.2 Função de Transferência

A partir da representação em espaço de estados, utilizando o MATLAB, chegou-se a seguinte equação:

$$G(s) = \frac{2066}{s^2 + 136,4s + 2066} \tag{2.6}$$

2.3 Projeto de Controlador por Alocação de Polos

A função de transferência da malha fechada foi obtida utilizando a seguinte definição:

$$T(s) = \frac{D(s)G(s)}{1 + D(s)G(s)} \tag{2.7}$$

Sendo: T(s): função de transferência da malha fechada; G(s): função de transferência da malha aberta; D(s): função do controlador.

Assim, definindo

$$D(s) = K_p \tag{2.8}$$

obteve-se a seguinte função de transferência da malha fechada:

$$T(s) = \frac{2066K_p}{s^2 + 136,4s + (K_p + 1)2066} \tag{2.9}$$

A partir da equação temos que:

$$\zeta = \frac{68.2}{w_s} \tag{2.10}$$

Para que o sistema se torne oscilatório, K_p deve respeitar a seguinte inequação:

$$0 < \frac{68.2}{w_s} < 1 \tag{2.11}$$

Isso implica que:

$$K > 1.2513 \tag{2.12}$$

Para se escolher uma constante de tempo para a discretização deve-se escolher uma constante menor que:

$$T < 2/w_s \tag{2.13}$$

$$T < 0.0147s \tag{2.14}$$

2.4 Função de Transferência da Malha Fechada

Utilizando K_p como 10, chegamos à equação:

$$T(s) = \frac{20660}{s^2 + 136.4s + 22726} \tag{2.15}$$

Discretizando a função de transferência contínua anterior e constante de tempo T = 0.0018 s, a T(z) foi definida como:

$$T(z) = \frac{0.0307z + 0.02829}{z^2 - 1.717z + 0.7823} \tag{2.16}$$

Para projetar os controladores a seguir será utilizada como planta as funções de transferência acima.

2.5 Projeto controlador PI

O controlador Proporcional e Integral (PI) foi modelado seguindo as definições a seguir:

$$D(z) = K_p + K_i \frac{1}{z - 1} \tag{2.17}$$

$$K_i = \frac{K_p T}{T_i} \tag{2.18}$$

Em que: Kp: constante da ação proporcional; Ki: constante da ação integral; T: tempo de amostragem; Ti: tempo de integração.

Com o objetivo de eliminar as oscilações do sistema, eliminar o erro em estado estacionário e possibilitar que o sistema responda mais rapidamente foram definidos os valores de K_p e K_i, que foram:

$$K_p = 0,023404 \tag{2.19}$$

$$T_i = 0,0009 \tag{2.20}$$

O que leva a seguinte ação de controle:

$$D_{PI}(z) = 0.023404 + 0.0468 \frac{1}{z - 1} \tag{2.21}$$

Assim, a função de transferência da malha fechada obtida foi:

$$T_{PI}(z) = \frac{0,0007186(z + 1)(z + 0,9213)}{(z - 0,9509)(z^2 - 1,766z + 0,822)} \tag{2.22}$$

2.6 Projeto controlador Dahlin

O controlador de Dahlin foi modelado seguindo as seguintes definições:

$$\tau = 0,01 \tag{2.23}$$

$$k = 1 \tag{2.24}$$

$$q = e^{-\frac{T}{\tau}} \tag{2.25}$$

$$D_{Dahlin}(z) = \frac{1}{G(z)} \frac{(1-q)z^{-k}}{1 - qz^{-1} - (1-q)z^{-k}} \tag{2.26}$$

Assim, após a realização dos cálculos, com τ igual a 0,01 s e k=1 devido ao atraso intrínseco, a função do controlador Dahlin foi definida como:

$$D_{Dahlin}(z) = \frac{5,365(z^2 - 1,717z + 0,7823)}{(z + 0.9213)(z - 1)} \tag{2.27}$$

Fechando a malha, obteve-se a seguinte função de transferência pulsada para malha fechada:

$$T_{Dahlin}(z) = \frac{0,16473}{z - 0.8353} \tag{2.28}$$

3 Resultados

3.1 Resposta do sistema à uma entrada em degrau

Simulando o sistema discretizado, obteve-se o seguinte resultado:

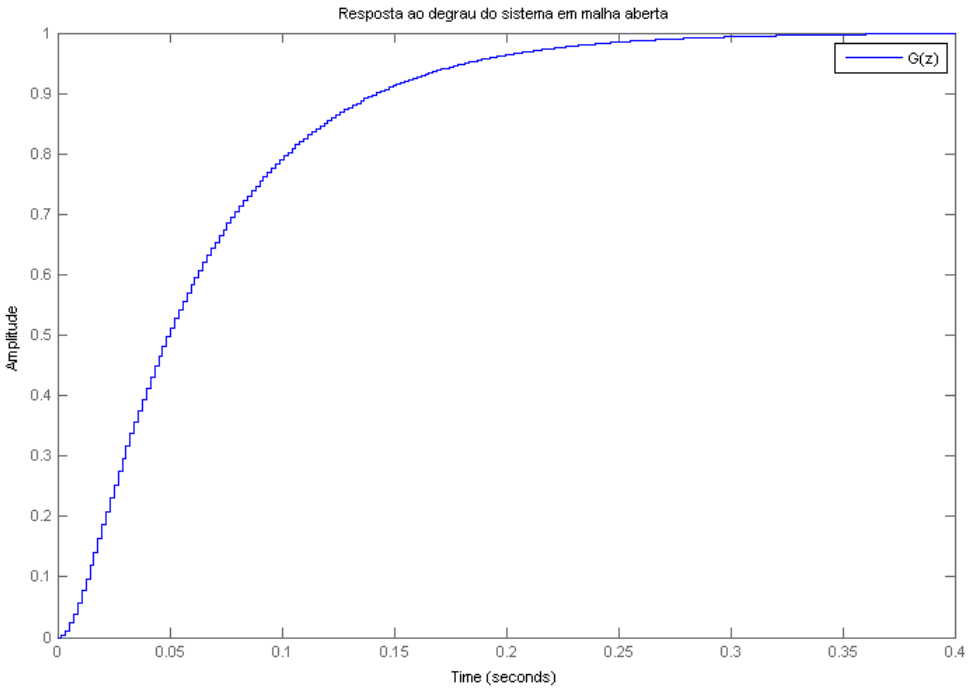

Figura 2 – Resultado da simulação do sistema em malha aberta

Após a medição com auxílio do osciloscópio foi criado o seguinte gráfico comparativo:

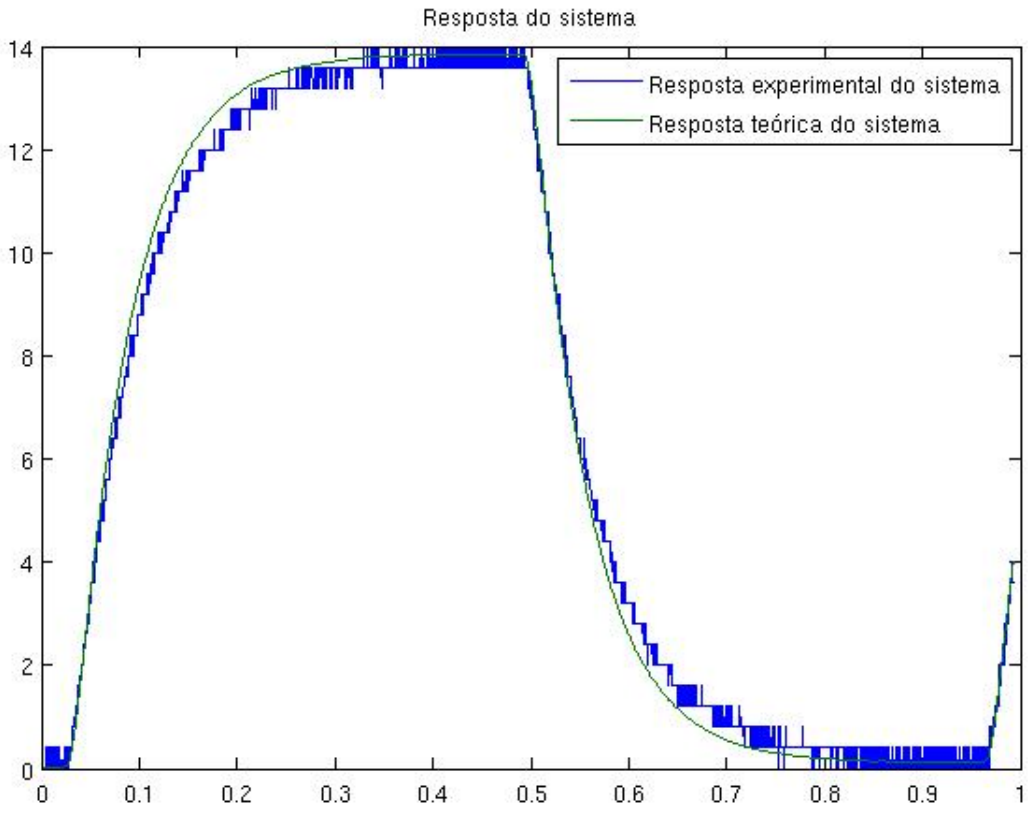

Figura 3 – Comparação resultado experimental e teórico

3.2 Resposta pulsada do sistema ao controlador por alocação de polos

Comparando o resultado da simulação com o experimental, observa-se a seguinte similiaridade entre as duas respostas:

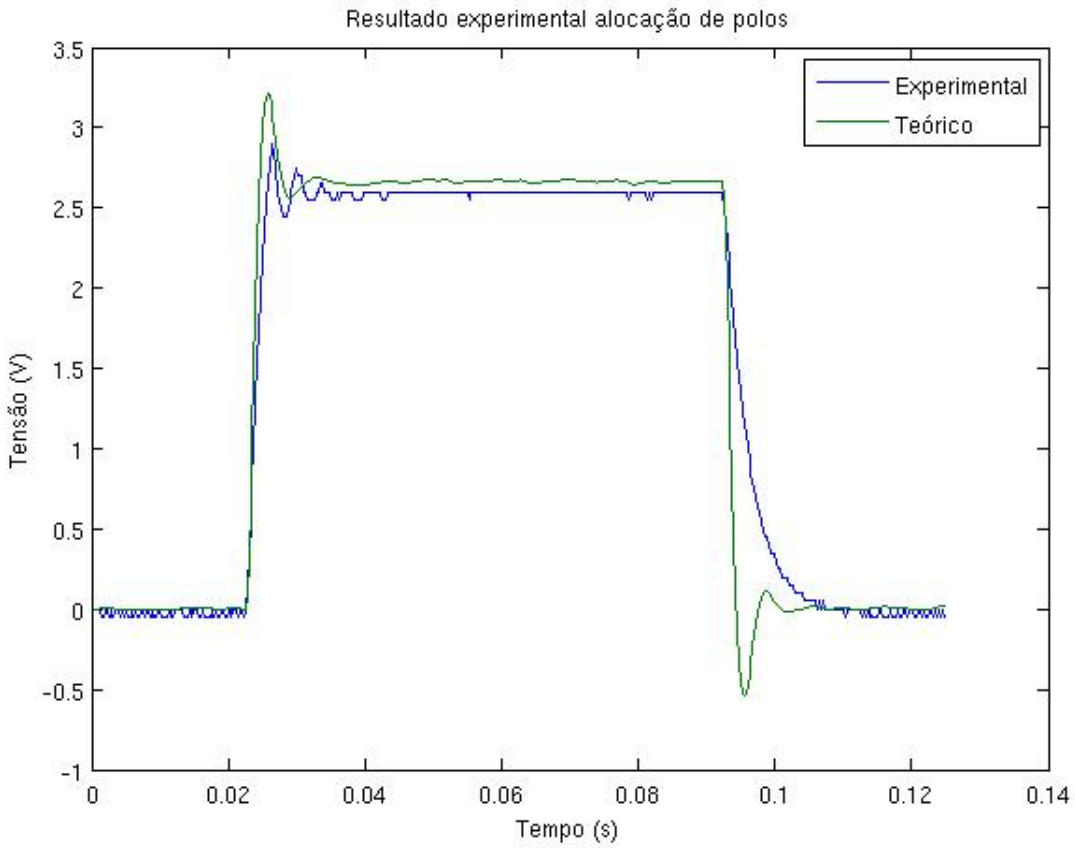

Figura 4 – Comparação resultado experimental e teórico para controlador por alocação de polos digital

3.3 Controlador PI

Após a modelagem do controlador PI, simulou-se o sistema afim de verificar a modelagem desenvolvida e comparar o resultado com a simulação do sistema com o controlador por alocação de polos. Com o auxílio do MATLAB, a simulação obteve o seguinte resultado:

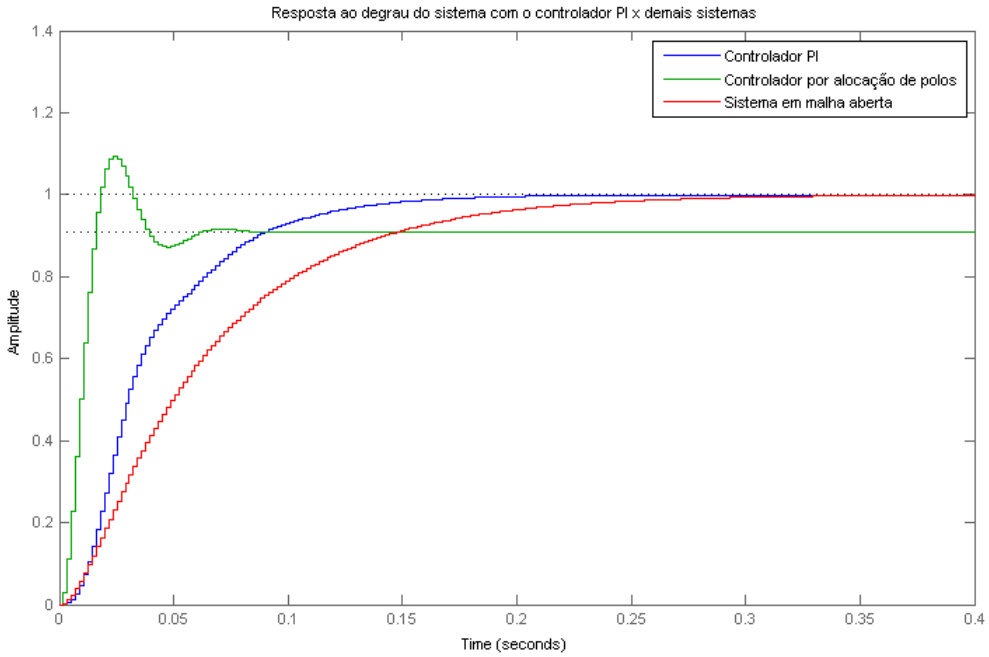

Figura 5 – Resultado da simulação do controlador PI x demais sistemas

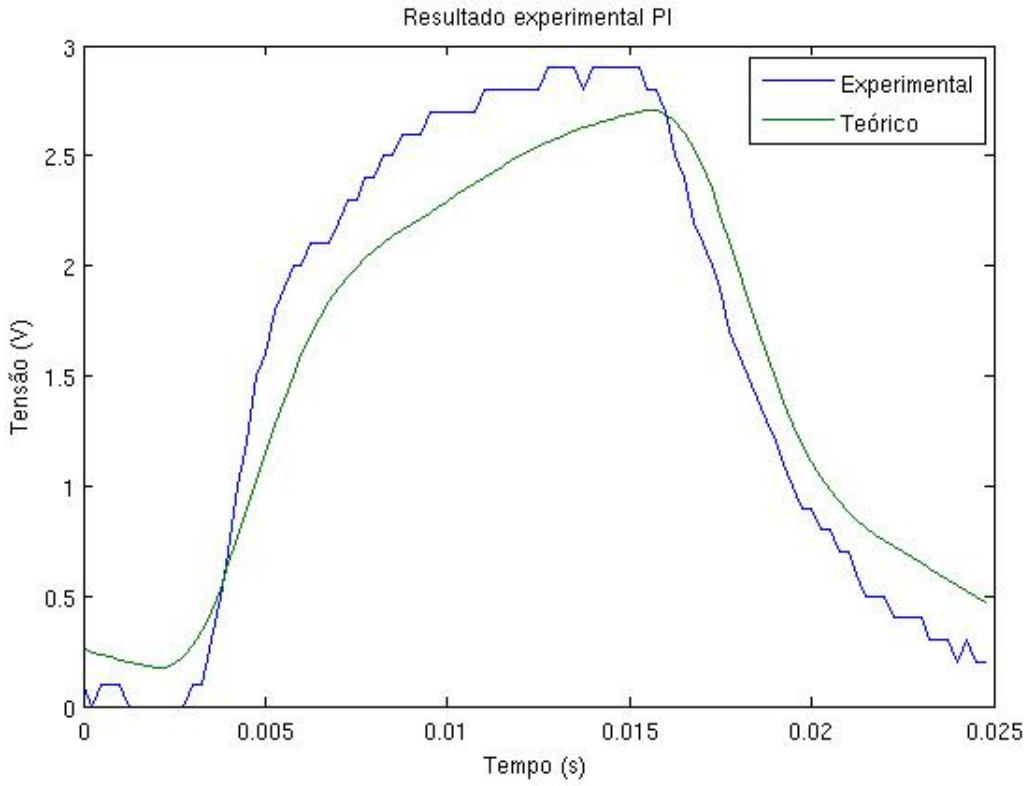

Figura 6 – Comparação entre resultado teórico e experimental para o controlador PI

Analisando o gráfico, pode-se constatar que o sistema utilizando o controlador PI eliminou o erro em estado estacionário, em relação ao controlador por alocação de polos, e obteve resposta mais rápida em relação ao

sistema em malha aberta, utilizando os devidos valores de Kp e Ki calculados.

3.4 Controlador Dahlin

Comparou-se as respostas do sistema com um controlador de ganho proporcional igual a 10, com um controlador de Dahlin e a resposta do sistema em malha aberta, resultando no seguinte gráfico comparativo:

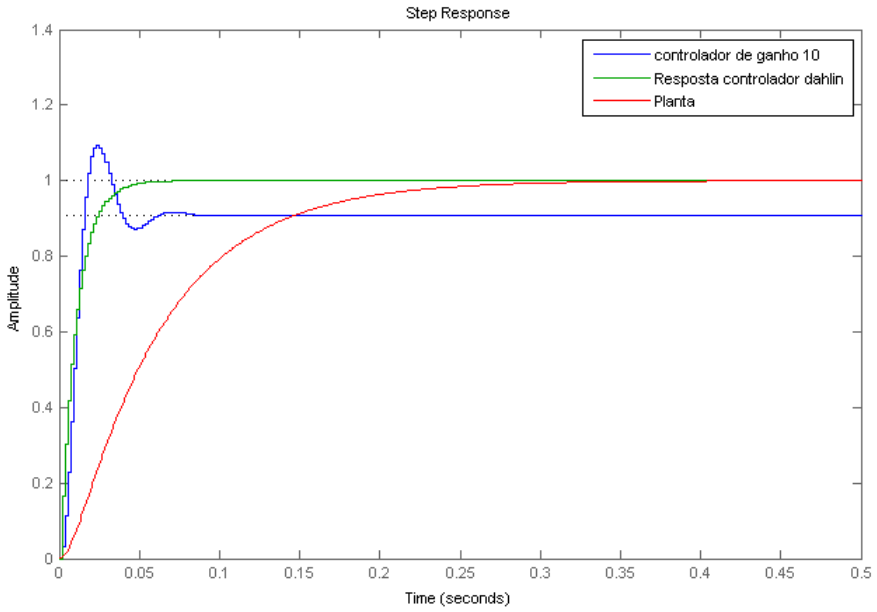

Figura 7 – Resultado da simulação do sistema utilizando o controlador de Dahlin

Observa-se que o controlador de Dahlin respondeu mais rapidamente que os outros, eliminando o erro de estado estacionário do controlador proporcional.

APÊNDICE A – Controlador Proporcional

```
/* ------ Controlador proporcional para um circuito RC ------ */

const int Referencia = 0; // O sinal de referencia sera lido a partir do pino de
                          // entrada analogica A0

const int Sensor = 1; // O sensor sera conectado ao pino de entrada analogica A1

const int Atuador = 3; // O sinal de comando "analogico" (saida do controlador)
                       // sera transmitido pelo pino de saida digital 5 (PWM)

int Valor_Referencia=0; // Variavel que armazenara o valor do sinal de referencia

int Valor_Sensor; // Variavel que armazenara o valor da saida (tensao no capacitor)

int Valor_Atuador; // Variavel que armazenara o valor da
                   // saida do controlador (acao de controle)

int Erro; // Variavel que armazenara o sinal de erro (Valor_referencia - Valor_Sensor)

const float Kp = 10; // Parametro do controlador proporcional
const float T = 1800;
float anterior=0, agora;
float ct = 0;

void setup(){

pinMode(Sensor, INPUT); // Define o pino do sensor como uma entrada

pinMode(Referencia, INPUT); // Define o pino da referencia como uma entrada

pinMode(Atuador, OUTPUT); // Define o pino do atuador como uma saida

Serial.begin(9600); // Especifique a velocidade da comunicacao serial
pinMode(5,OUTPUT);
}



void loop(){
  agora = micros();
  if(agora-anterior>=T){
```

```
  anterior = agora;
  Valor_Referencia = analogRead(Referencia); // Converte o valor de tensao de referen
  // numa palavra binaria de 10 bits (0V a 5V <---> 0 a 1023)

  Valor_Sensor = analogRead(Sensor); // Converte o valor de tensao do capacitor (saida
  // numa palavra binaria de 10 bits (0V a 5V <---> 0 a 1023)

  Erro = Valor_Referencia - Valor_Sensor; // Erro pode assumir valores entre -1023 e

  Erro = map(Erro,-512,511,-128,127); // Mapeia o intervalo [-1023, 1023] no intervalo
  // [-255, 255], visto que o saida do controlador, que e funcao so sinal de Erro,
  //sera um sinal PWM de 8 bits

  Valor_Atuador = constrain(Kp*Erro,0,255); // Restringe o valor do sinal de atuacao
                                            // a faixa de 0V a 5V (0 a 255)

  analogWrite(Atuador, Valor_Atuador); // Escreve no pino 5 (Atuador), que simulara
                                       // uma saida analogica via PWM
  Serial.print(Erro);
  Serial.print("\t\t");
  Serial.println(Valor_Atuador);
  ct++;
  if(ct==100){
     ct=0;
     PORTD ^= 0b00100000;
  }
  }


  //delay(100);

}
```

APÊNDICE B – Controlador Proporcional Integrativo

```
/* ------ Controlador proporcional para um circuito RC ------ */


const int Referencia = 0; // O sinal de referencia sera lido a partir do pino de
                          // entrada analogica A0

const int Sensor = 1; // O sensor sera conectado ao pino de entrada analogica A1

const int Atuador = 3; // O sinal de comando "analogico" (saida do controlador)
                       // sera transmitido pelo pino de saida digital 5 (PWM)

int Valor_Referencia=0; // Variavel que armazenara o valor do sinal de referencia

int Valor_Sensor; // Variavel que armazenara o valor da saida (tensao no capacitor)

int Valor_Atuador; // Variavel que armazenara o valor da
                   // saida do controlador (acao de controle)

float Erro; // Variavel que armazenara o sinal de erro (Valor_referencia - Valor_Sens
float SumErro;
float Erro2;

float P=0;
float I=0;

const float K = 10.; // Parametro do controlador proporcional

const float Kp = 0.0234; // Parametro do controlador proporcional
const float Ki = 0.0468; // Parametro do controlador proporcional
const float T = 1800;
float anterior=0, agora;
float ct = 0;

void setup(){

pinMode(Sensor, INPUT); // Define o pino do sensor como uma entrada

pinMode(Referencia, INPUT); // Define o pino da referencia como uma entrada

pinMode(Atuador, OUTPUT); // Define o pino do atuador como uma saida

//Serial.begin(9600); // Especifique a velocidade da comunicacao serial
```

```
pinMode(5,OUTPUT);
}



void loop(){
  agora = micros();
  if(agora-anterior>=T){
  anterior = agora;
  Valor_Referencia = analogRead(Referencia); // Converte o valor de tensao de referen
  // numa palavra binaria de 10 bits (0V a 5V <---> 0 a 1023)

  Valor_Sensor = analogRead(Sensor); // Converte o valor de tensao do capacitor (saida
  // numa palavra binaria de 10 bits (0V a 5V <---> 0 a 1023)

  Erro = Valor_Referencia - Valor_Sensor; // Erro pode assumir valores entre -1023 e


  SumErro += Erro;
  P = Kp*Erro;
  I = Ki*SumErro;

  Erro2 = K*((P+I)-Valor_Sensor);
  Erro2 = map(Erro2,-512,511,-128,127);

  Valor_Atuador = constrain(Erro2,0,255); // Restringe o valor do sinal de atuacao
                                          // a faixa de 0V a 5V (0 a 255)

  analogWrite(Atuador, Valor_Atuador); // Escreve no pino 5 (Atuador), que simulara
                                       // uma saida analogica via PWM
  //Serial.print(Erro);
  //Serial.print("\t\t");
  //Serial.print(Erro2);
  //Serial.print("\t\t");
  //Serial.println(Valor_Atuador);
  ct++;
  if(ct==100){
     ct=0;
     PORTD ^= 0b00100000;
  }
  }


  //delay(100);

}
```

APÊNDICE C – Controlador Dahlin

```
/* ------ Controlador proporcional para um circuito RC ------ */


const int Referencia = 0; // O sinal de referencia sera lido a partir do pino de
                          // entrada analogica A0

const int Sensor = 1; // O sensor sera conectado ao pino de entrada analogica A1

const int Atuador = 3; // O sinal de comando "analogico" (saida do controlador)
                       // sera transmitido pelo pino de saida digital 5 (PWM)

int Valor_Referencia=0; // Variavel que armazenara o valor do sinal de referencia

int Valor_Sensor; // Variavel que armazenara o valor da saida (tensao no capacitor)

int Valor_Atuador; // Variavel que armazenara o valor da
                   // saida do controlador (acao de controle)

float Erro; // Variavel que armazenara o sinal de erro (Valor_referencia - Valor_Sens

float Erro2;

float e[3] = {0,0,0};
float c[3] {0,0,0};

const float K = 10.; // Parametro do controlador proporcional

const float T = 1800;
float anterior=0, agora;
float ct = 0;

void setup(){

pinMode(Sensor, INPUT); // Define o pino do sensor como uma entrada

pinMode(Referencia, INPUT); // Define o pino da referencia como uma entrada

pinMode(Atuador, OUTPUT); // Define o pino do atuador como uma saida

Serial.begin(9600); // Especifique a velocidade da comunicacao serial
pinMode(5,OUTPUT);
}
```

```
void loop(){
  agora = micros();
  if(agora-anterior>=T){
  anterior = agora;
  Valor_Referencia = analogRead(Referencia); // Converte o valor de tensao de referen
  // numa palavra binaria de 10 bits (0V a 5V <---> 0 a 1023)

  Valor_Sensor = analogRead(Sensor); // Converte o valor de tensao do capacitor (saida
  // numa palavra binaria de 10 bits (0V a 5V <---> 0 a 1023)

  Erro = Valor_Referencia - Valor_Sensor; // Erro pode assumir valores entre -1023 e

  e[2] = e[1];
  e[1] = e[0];
  e[0] = Erro;

  c[2] = c[1];
  c[1] = c[0];
  c[0] = 5.365*e[0] + 0.0787*c[1] - 0.9213*c[2] - 9.212*e[1] + 4.197*e[2];


  Erro2 = K*(c[0]-Valor_Sensor);
  Erro2 = map(Erro2,-188477.52,109767.9,-255,255);

  Valor_Atuador = constrain(Erro2,0,255); // Restringe o valor do sinal de atuacao
                                          // a faixa de 0V a 5V (0 a 255)

  analogWrite(Atuador, Valor_Atuador); // Escreve no pino 5 (Atuador), que simulara
                                       // uma saida analogica via PWM
  Serial.print(Erro);
  Serial.print("\t\t");
  Serial.print(Erro2);
  Serial.print("\t\t");
  Serial.println(Valor_Atuador);
  ct++;
  if(ct==100){
     ct=0;
     PORTD ^= 0b00100000;
  }
  }
}
```