Determinación de plata

Diagrama esquemático:

|  |
| --- |
|  |

El modelo del siguiente péndulo se puede determinar como una ecuación de momentos, de la siguiente manera:

El rango de movimiento es de 30 de a -30 grados tomando de referencia a 0 grados la línea marcada como 90 grados en el esquema. Por lo que asumimos que: además de que la fuerza del resorte, no varía en función del ángulo, si no que siempre permanece tangencial al péndulo, quedándonos con lo siguiente, donde los valores que pueden ser obtenidos, son la masa del péndulo asimilada como la masa del motor (m), el largo (L) del péndulo, la gravedad (g) y el brazo del resorte (l\_k). Los desconocidos son la fricción (b) y la fuerza del resorte (F\_k).

El empuje provisto por el motor, está en función de la señal de voltaje PWM provisionada al mismo, la constante K\_m puede ser obtenida al variar este voltaje y medir la fuerza provisionada por el motor experimentalmente u obtenerlo del fabricante:

El resorte está fijado a la placa base del péndulo a cierta distancia del eje *y*, fijado como una referencia donde las coordenadas son de (0,0), así como también hay una distancia al punto donde rota el péndulo hasta el eje colocado. Este eje de referencia utilizado y distancias fueron utilizados para determinar el largo del resorte, ya que la fuerza del resorte varía en función de esta, y la distancia varía en función del ángulo. Sin embargo, la varianza del largo no es lineal, pero se puede hacer esta asimilación, solo tomando el máximo y el mínimo hasta donde se estira el resorte y obteniendo una función lineal en función del ángulo, donde las distancia y ángulo de referencia medido inicial son 16.2 cm y -34.1 grados y las finales son 32.6 cm y 34.1 grados. La constante K del resorte fue medida despejando de la ley de Hooke.

A partir de aquí tenemos el siguiente modelo ya lineal:

Podemos observar como la fuerza del resorte al estar en reposo el péndulo permanece una constante

Código de arduino:

#include <Servo.h>

#define tiempo\_muestreo 50

#define A0\_raw\_maximum  769

#define A0\_raw\_minimum  1007

#define maximum\_angle   68.2

#define minimum\_angle   0

#define max\_pwm 1500

#define kp\_pot A1

#define ki\_pot A2

const int input\_pin = A0;

const int output\_pin = 9;

Servo ESC;

int angulo;

double deltatiempo, tiempo\_anterior, integral, antes, pwm\_output;

double u\_t,kp, ki, kd, setpoint= -1;

double alfa, theta, potcorr;

unsigned long tiempo\_inicio;

void setup() {

  ESC.attach(9);

  ESC.writeMicroseconds(1000);

  //kp= 0.2940;

  //ki= 0.;

  kd= 0.0441;

  tiempo\_anterior= 0;

  Serial.begin(9600);

  Serial.setTimeout(10);

}

void loop() {

//Muestreo de tiempo

  if(millis() - tiempo\_inicio >= tiempo\_muestreo){

    tiempo\_inicio = millis();

    deltatiempo= (tiempo\_inicio - tiempo\_anterior)/1000.00;

    tiempo\_anterior = tiempo\_inicio;

    double sensor= angulos(input\_pin,alfa);

    double error= setpoint - sensor;

    pwm\_output= int(pid(error));

//Acomplamiento conversion de angulo a pwm

    //Establecimiento de setpoint y seguro para apagarlo

    if(Serial.available() >= 1)

    {

      setpoint = Serial.parseInt(); //Leer un entero por serial

    }

    if(setpoint == -1){

      pwm\_output= 1000;

    }

    //Modifica el pulso enviado al motor

    ESC.writeMicroseconds(pwm\_output);

    //Imprime datos

    Serial.print(tiempo\_inicio);

    Serial.print(" ");

    Serial.print(setpoint);

    Serial.print(" ");

    Serial.print(sensor);

    Serial.print(" ");

    Serial.print(error);

    Serial.print(" ");

    Serial.println(pwm\_output);

  }

}

double pid(double error){

  double proporcional = error;

  integral += error\*deltatiempo;

  double derivativo= (error - antes)/deltatiempo;

  antes = error;

  double kp = mapf(analogRead(kp\_pot), 0, 1023, 0.00, 2.00);

  double ki = mapf(analogRead(ki\_pot), 0, 1023, 0.00, 1.00);

  double u\_t= (kp\*proporcional) + (ki\*integral) + (kd\*derivativo);

  Serial.print(u\_t);

  Serial.print(" ");

  Serial.print(kp);

  Serial.print(" ");

  Serial.print(ki);

  Serial.print(" ");

  //double u = 2.1357\*u\_t + 1226.2;

  double u= 1226.2 + u\_t;

  if(u > 1500)

    u = 1226.2;

  return u;

}

//Filtro para valor de potenciometro con ruido

  float yn\_1 = 0;

float fiir(int input\_pin, float alfa){

  float xn, yn;

  alfa = 0.17;

  yn = alfa\*float(analogRead(input\_pin)) + (1-alfa)\*yn\_1;

  yn\_1 = yn;

  return yn;

}

//Funcion para calcular angulo theta

float angulos(int input\_pin, float alfa){

  //potcorr= fiir(input\_pin,alfa);

  theta = (maximum\_angle-(minimum\_angle))/(A0\_raw\_maximum-A0\_raw\_minimum)\*(analogRead(input\_pin) - A0\_raw\_minimum) + minimum\_angle;

  //Serial.print(theta);

  if(theta < 0){

    theta= 0;

  }

  return theta;

}

//Mod de kp

double mapf(double val, double in\_min, double in\_max, double out\_min, double out\_max) {

    return (val - in\_min) \* (out\_max - out\_min) / (in\_max - in\_min) + out\_min;

}