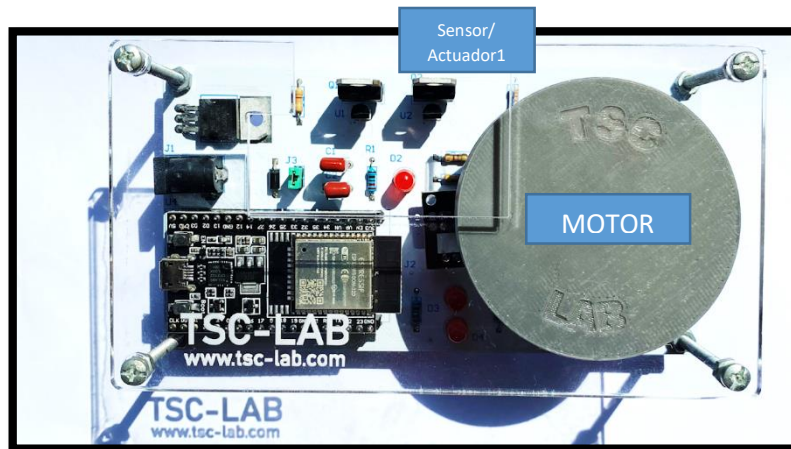


Nombre: Denisse Fiallos Cañarte

Proyecto de sistemas de control Embebidos

1. Descripción de la planta seleccionada (incluir fotografías), así como las variables involucradas

La planta del proyecto describe la relación de bajar temperatura al momento que se activa el motor, dicho valor es de la temperatura 1 que envía el sensor 1 por tal motivo comienza a descender debido al viento ejercido por el motor. Esta acción evita que el sensor1 y su actuador1 eleven su lectura de temperatura.

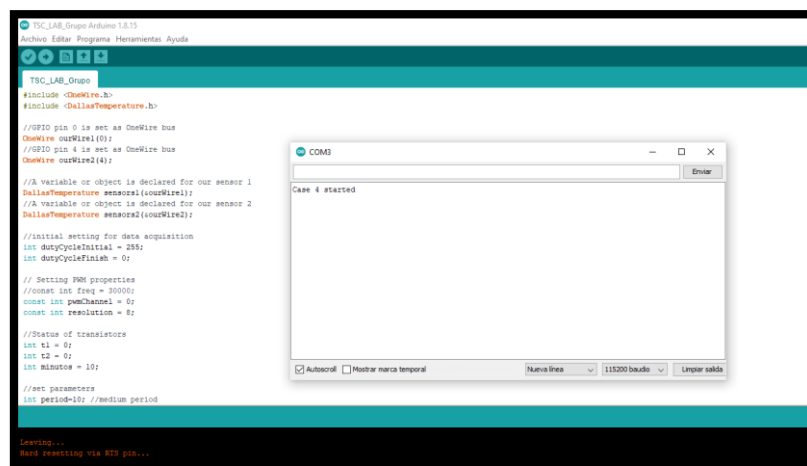


Las variables en el proyecto son las siguientes:

- Los pulsos ejercidos a la salida del motor es la señal de entrada
- La lectura del sensor1 es la salida de la planta

2. Metodología, donde se deberán describir claramente los puntos del 1 al 6 (descritos arriba).

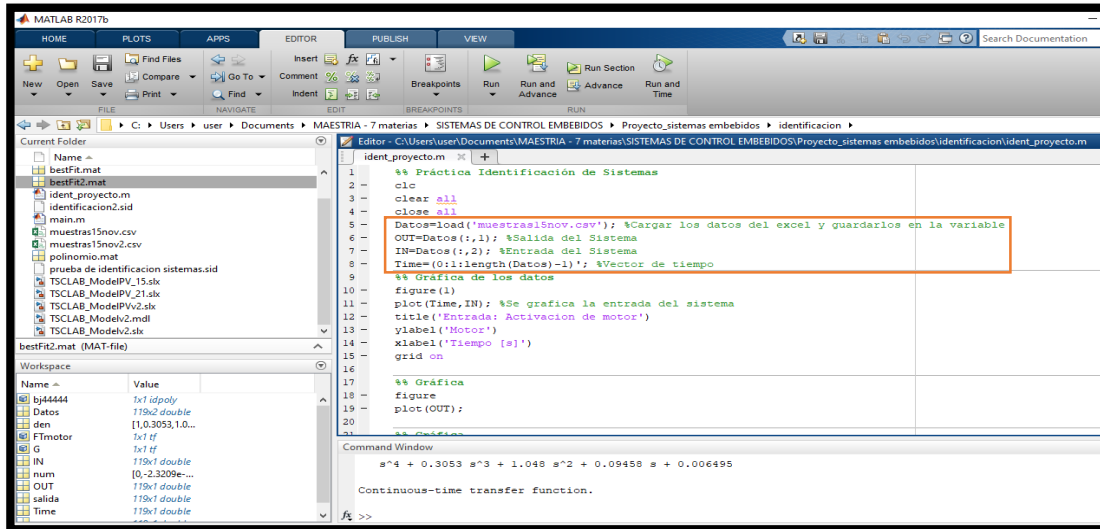
2.1. Se procede a ejecutar el programa de arduino y se abre el monitor serial para observar los valores de temperatura y velocidad durante aproximadamente unas cuatro horas. En las siguientes figuras se observa el procedimiento indicado:



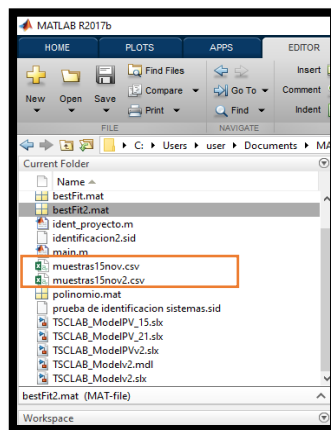
Nombre: Denisse Fiallos Cañarte

Proyecto de sistemas de control Embebidos

Luego se guardan estos datos en un archivo .csv, dicho archivo será utilizado en matlab para realizar el análisis correspondiente.



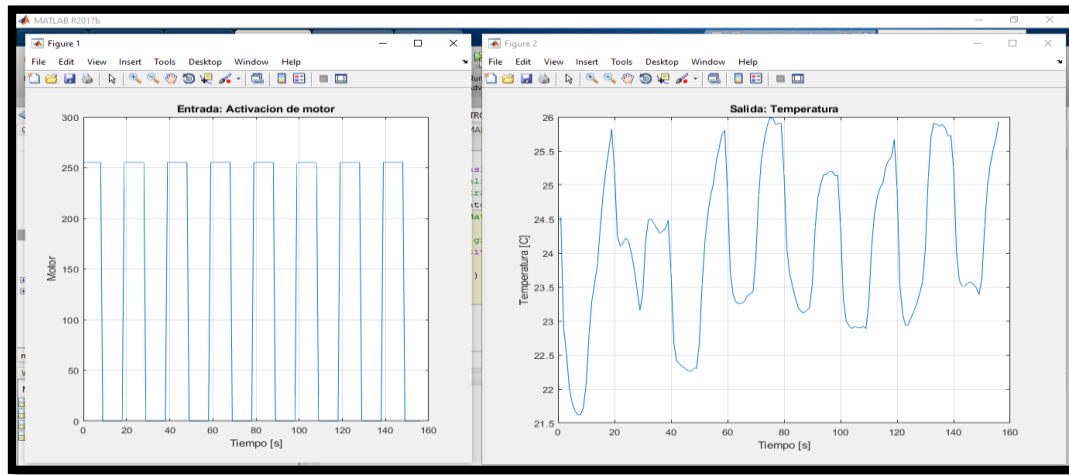
Se realizaron dos muestreos, el cual permitió generar dos archivos para realizar su posterior análisis de cada uno.



Se realiza la gráfica de la tabla de datos del archivo: muestras15nov2.csv. La grafica se muestra en las siguientes figuras:

Nombre: Denisse Fiallos Cañarte

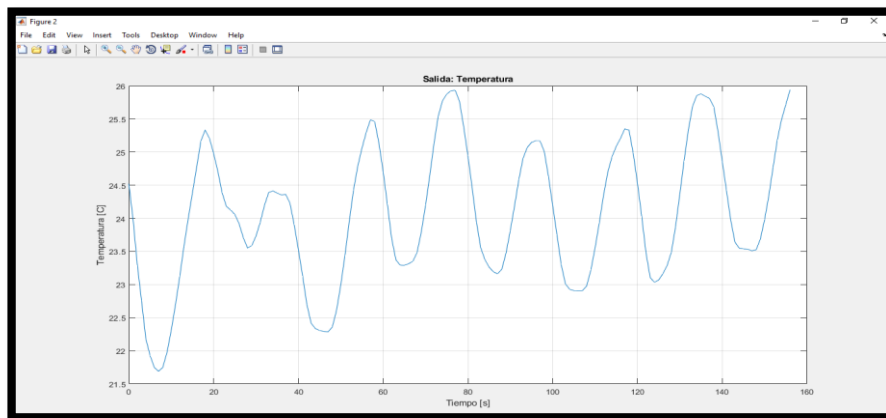
Proyecto de sistemas de control Embebidos



Utilizando el filtro (smooth), y lo guardamos en la variable ‘salida’:

```
figure  
salida=smooth(OUT);
```

Logramos eliminar ciertas perturbaciones en la gráfica como se observa en la siguiente grafica

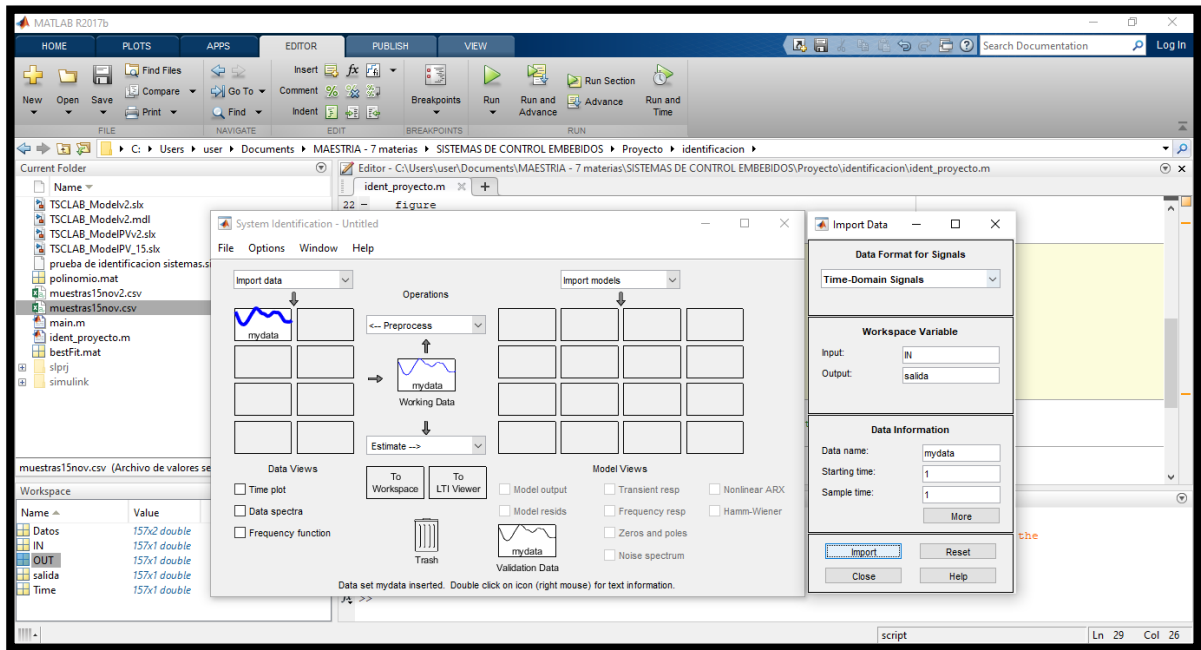


2.2. Incluir la identificación de la planta y la comparación en simulink entre el modelo identificado y la planta real.

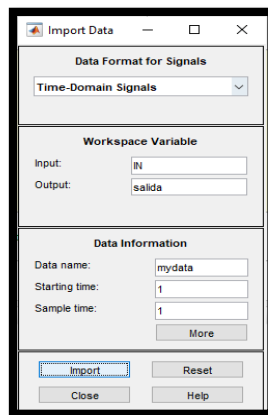
Procedemos a llamar al tool de matlab **SYSTEM IDENTIFICATION** y seleccionamos **TIME DOMAIN DATA** esta opción permite cargar los datos que matlab a extraído del archivo hacia el tool y poder realizar el procedimiento de tratamiento de datos para la identificación del sistema.

Nombre: Denisse Fiallos Cañarte

Proyecto de sistemas de control Embebidos



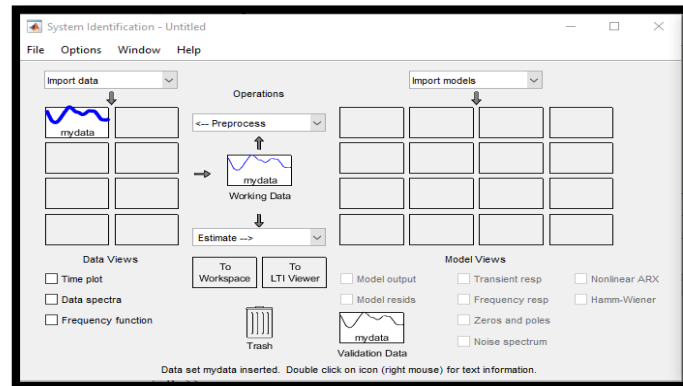
En **INPUT** ingresamos el nombre que se creó a la columna de entrada en este caso (IN), y en **Output** se ingresa el nombre creado con el filtro smooth (en este caso el nombre es: salida). Como se observa en la siguiente figura:



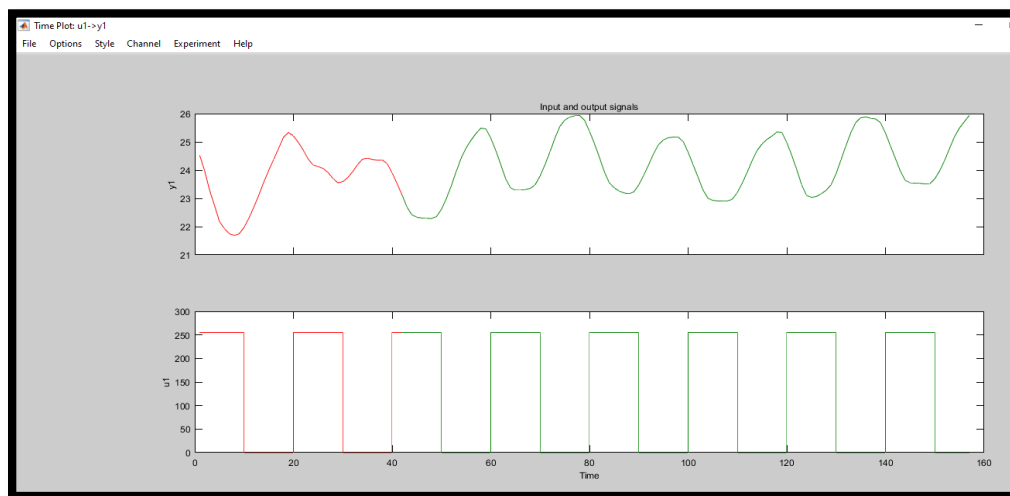
Al presionar el botón **IMPORT** en la ventana principal del tool de identificación aparece la gráfica de los datos. El cual hay que preprocesarla seleccionando los rangos de validation (30%) y working data (70%).

Nombre: Denisse Fiallos Cañarte

Proyecto de sistemas de control Embebidos



Al seleccionar los rangos nos aparece la gráfica con dos colores diferentes indicando la selección realizada, como se observa en la siguiente figura:

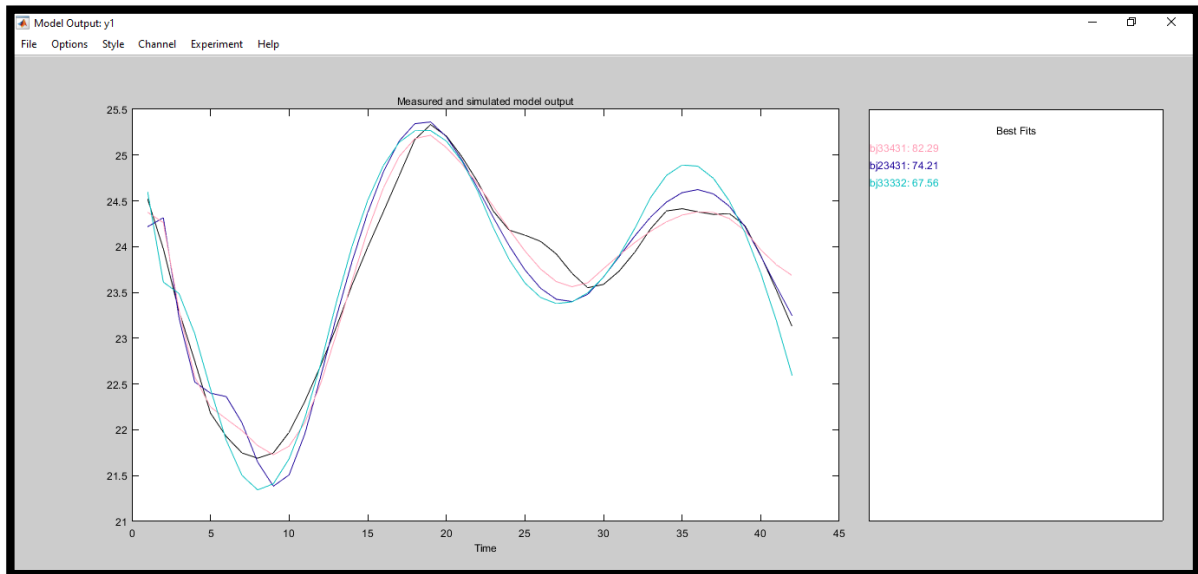


Procedemos a arrastrar el 70% seleccionado de la gráfica al recuadro donde dice WORKING DATA, y el 30% llevarlo al recuadro de VALIDATION.

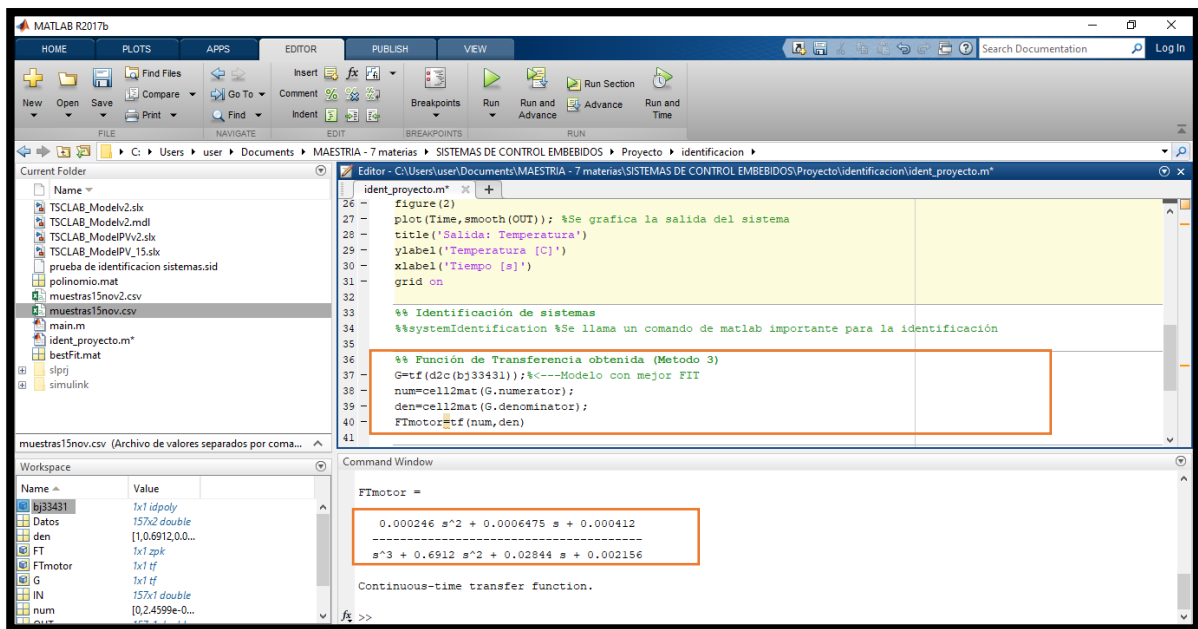
Y en ESTIMATE, seleccionamos polynomial models y BOOT JEKINS y comenzamos a realizar cambios en sus valores de polos y ceros. Para cada caso aparece una gráfica diferente el cual permite ser observado desde MODEL OUTPUT, como se observa en la siguiente figura:

Nombre: Denisse Fiallos Cañarte

Proyecto de sistemas de control Embebidos



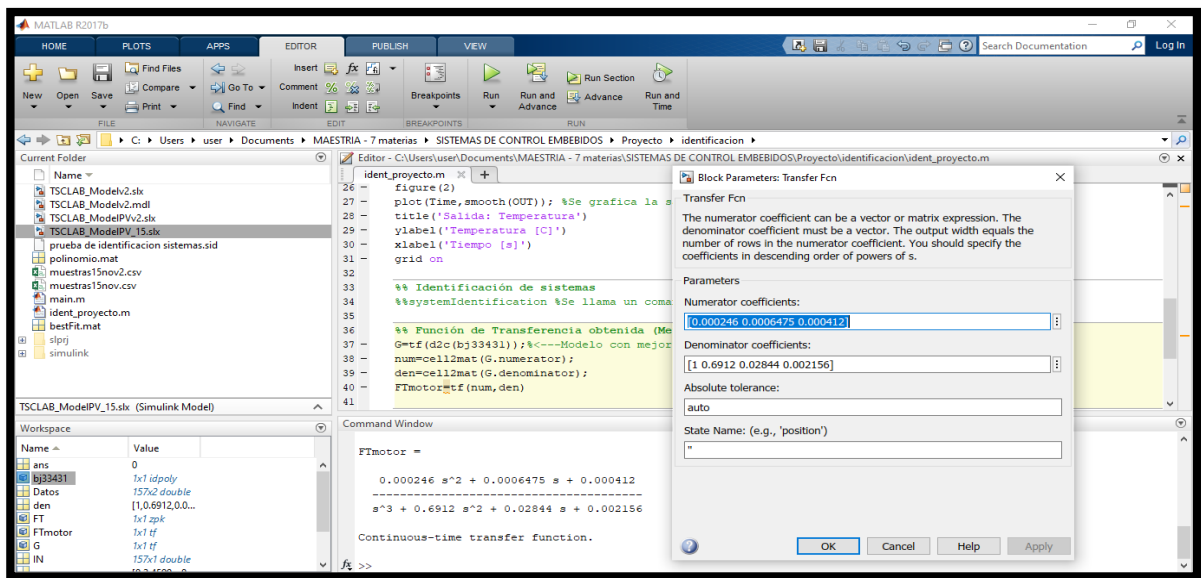
El siguiente paso es seleccionar el mejor FIT, en este caso es el 82,29% modelo BJ33431. Se procede a llevar a este modelo al workspace de matlab y se programa para sacar de este modelo la función de transferencia, como se observa en la siguiente figura:



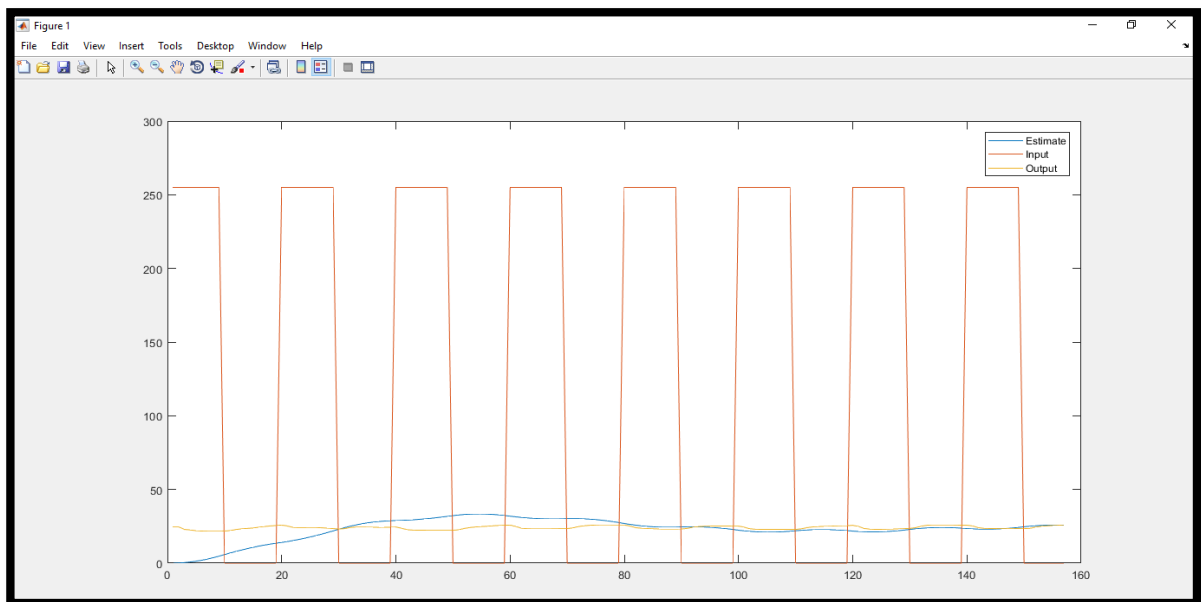
Luego se procede a ingresar los valores de la función de transferencia en simulink. Debido a problemas de versión del simulink que tengo instalado, no me reconoce ciertos bloques y me sale error al ejecutar por ello se realizó la gráfica directamente en el workspace de matlab.

Nombre: Denisse Fiallos Cañarte

Proyecto de sistemas de control Embebidos



Al ejecutar la programación con la función de transferencia aparece la siguiente figura:



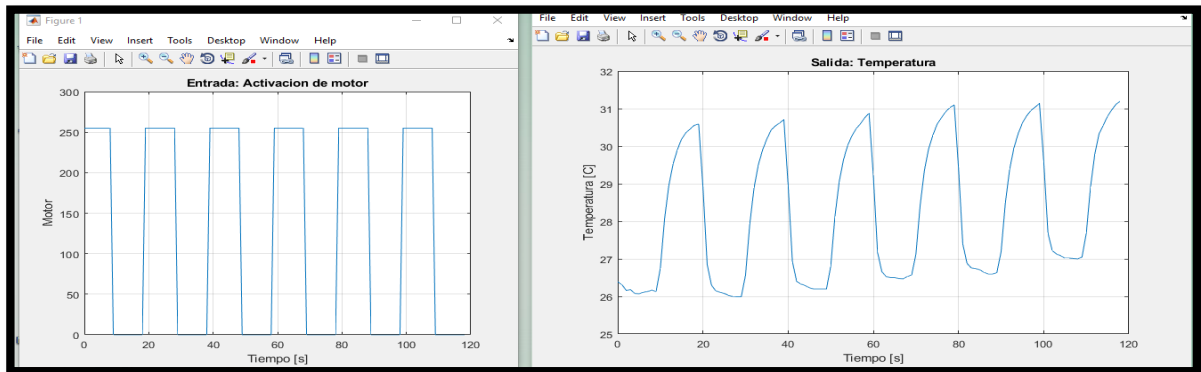
Nombre: Denisse Fiallos Cañarte

Proyecto de sistemas de control Embebidos

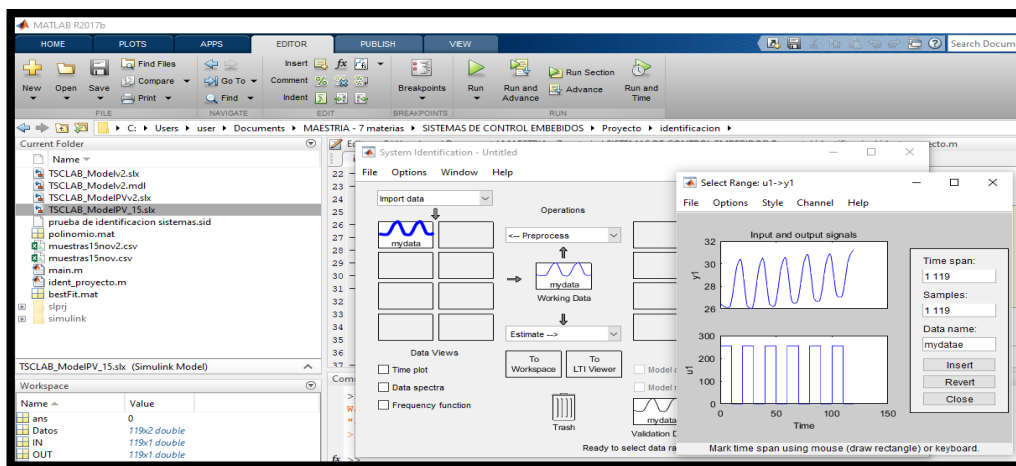
Opción 2

Luego se realiza el mismo procedimiento con el otro archivo **muestras15nov.csv** obtenido del muestreo de datos con la planta conectada.

Al realizar la carga de los datos en matlab, guardando en una variable la columna de entrada y en otra variable la columna de salida. Se realiza la gráfica de los datos en el workspace:



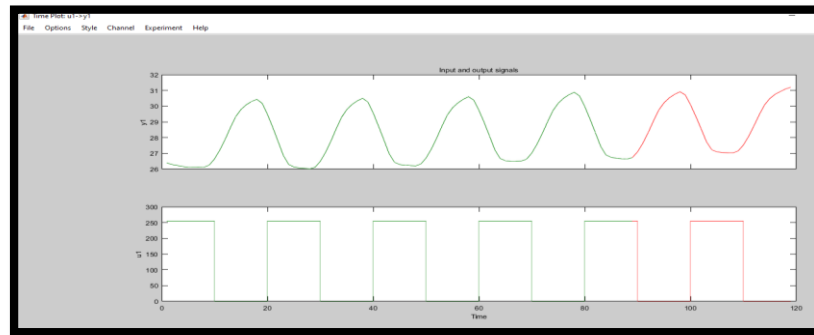
Posteriormente se cargan los datos en SYSTEM IDENTIFICATION, seleccionamos TIME DOMAIN DATA y se cargan las variables de la columna IN y la columna salida.



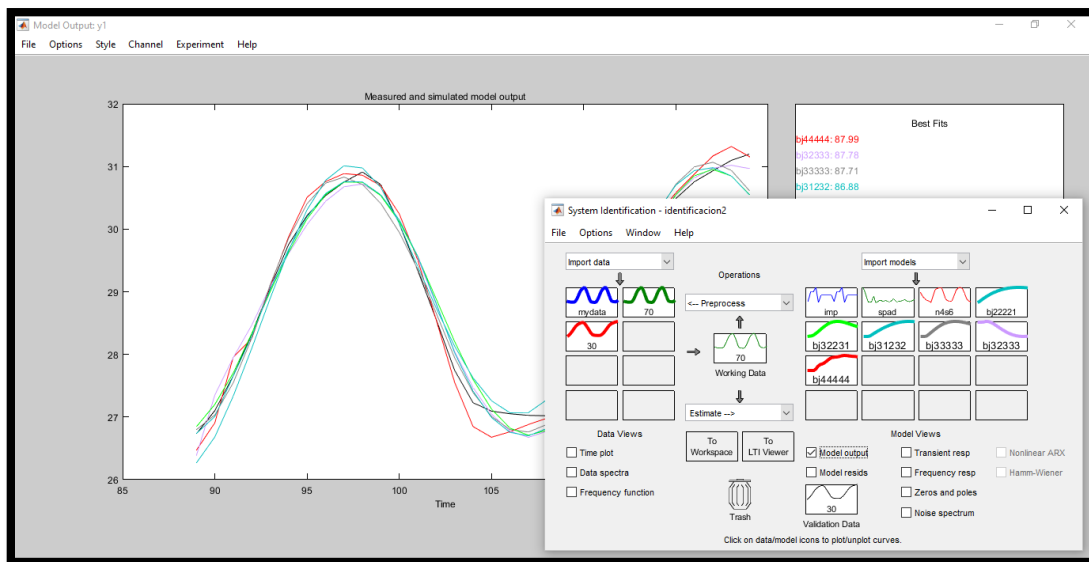
En preprocess seleccionamos el rango de la curva de datos con la que se trabajara: 70% Working Data, 30% Validation como podemos visualizar en la siguiente gráfica:

Nombre: Denisse Fiallos Cañarte

Proyecto de sistemas de control Embebidos



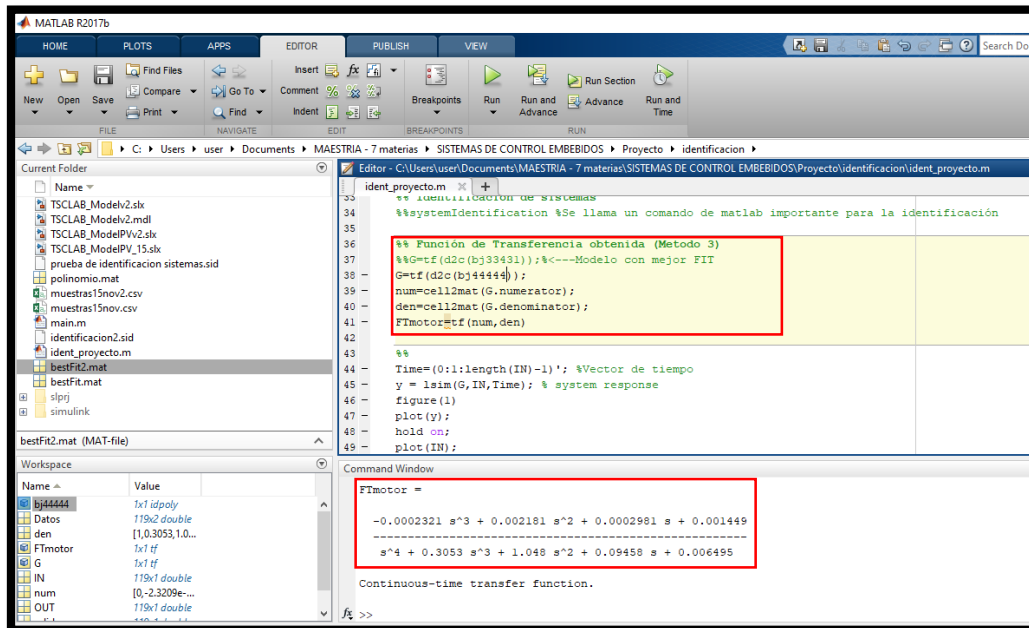
Luego arrastramos el 70% en WORKING DATA y el 30% en Validation Data. En Estimate seleccionamos modelo polinomial y procedemos a evaluar los distintos modelos como podemos observar en la siguiente figura:



Seleccionamos el mejor FIT que es 87.99 que es el modelo BJ44444 y lo arrastramos hasta el recuadro **TO WORKSPACE**. Procedemos a insertar el modelo seleccionado en la programación y ejecutamos, con ello obtenemos la función de transferencia:

Nombre: Denisse Fiallos Cañarte

Proyecto de sistemas de control Embebidos



```
ident_proyecto.m
%% AGREGACION DE SISTEMAS
%%systemIdentification %Se llama un comando de matlab importante para la identificación

%% Función de Transferencia obtenida (Metodo 3)
%%G=tf(d2c(bj33431)); %<---Modelo con mejor FIT
G=tf(d2c(bj44444));
num=cell2mat(G.numerator);
den=cell2mat(G.denominator);
FImotor=tf(num,den)

%%
Time=(0:1:length(IN)-1)'; %Vector de tiempo
y = lsim(G,IN,Time); % system response
figure(1);
plot(y);
hold on;
plot(IN);
```

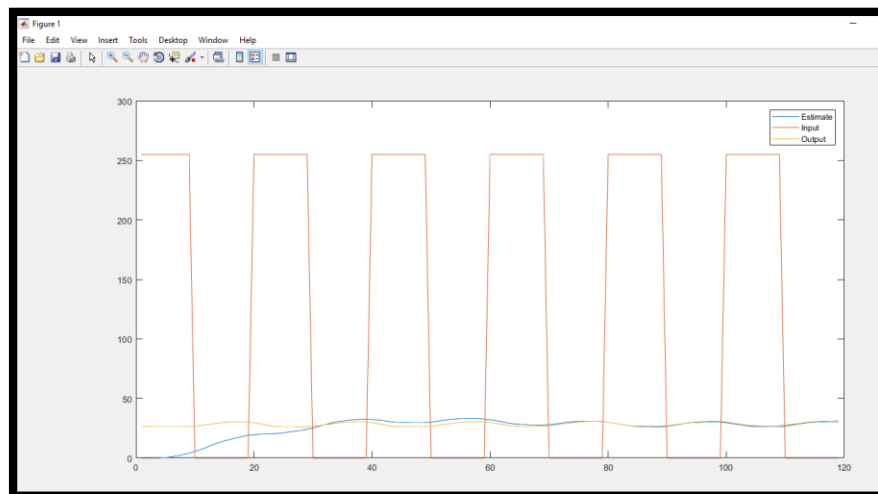
Workspace

Name	Value
bj44444	1x1 idpoly
Datos	119x2 double
den	[1,0.3053,1.0...
FImotor	1x1 tf
G	119x1 double
IN	[0,-2.3209e...
num	119x1 double
OUT	119x1 double

Command Window

```
FImotor =
-0.0002321 s^3 + 0.002181 s^2 + 0.0002981 s + 0.001449
-----
s^4 + 0.3053 s^3 + 1.048 s^2 + 0.09458 s + 0.006495
Continuous-time transfer function.
```

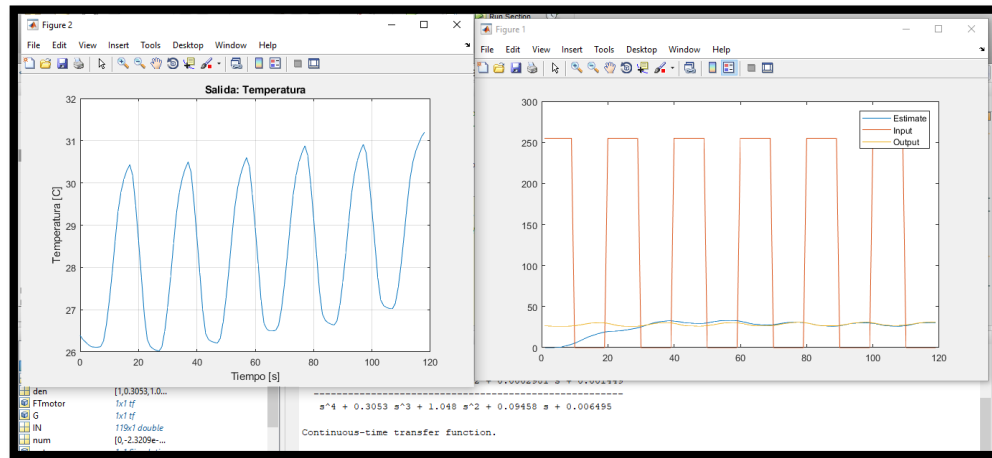
Con la programación realizada y la ecuación de transferencia obtenida, el siguiente paso es ingresar la ecuación en simulink, debido al problema de que no me reconoce ciertos bloques el simulink instalado en mí máquina. Procedo a ejecutar desde el workspace, dándome la siguiente gráfica:



Observando que la gráfica obtenida es más estable, es bastante parecida a la estimada. Puedo concluir que el modelo BJ44444 es el mejor por su comportamiento

Nombre: Denisse Fiallos Cañarte

Proyecto de sistemas de control Embebidos



3. El diseño del controlador PID es obligatorio

Se procede a llamar al tool PID TUNER, el cual arroja unos valores inmediatamente pero necesitamos evaluar la función:

```
Command Window

ans =

    Kp = 1

P-only controller.

>> pidTuner
>> Gc = pidtune(FTmotor,'PID')

Gc =

      1
Kp + Ki * ---- + Kd * s
      s

with Kp = 9.52, Ki = 0.4, Kd = 36.4

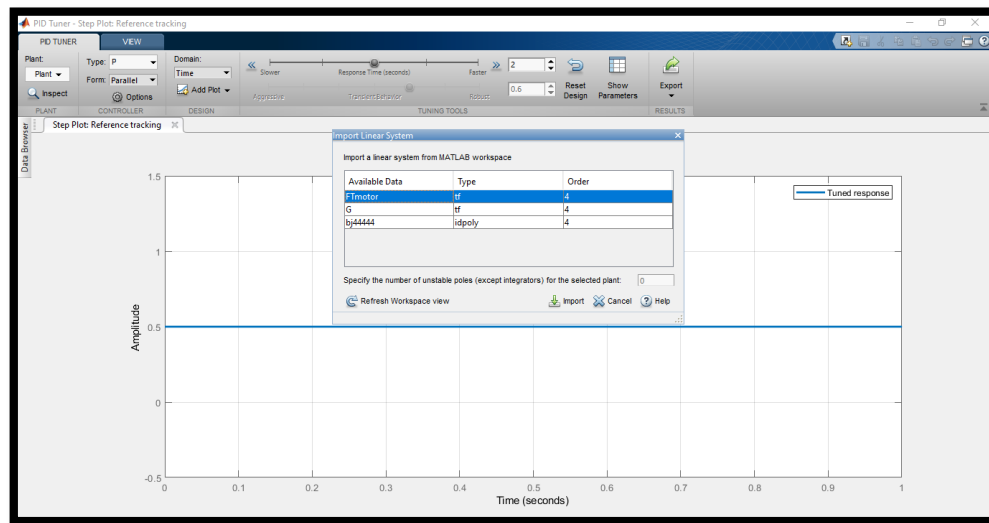
Continuous-time PID controller in parallel form.

>> pidTuner(FTmotor,Gc)
```

Al abrir **PID Tuner**, procedemos a importar la función de transferencia obtenida en este caso es FTmotor. Se observa en la siguiente figura:

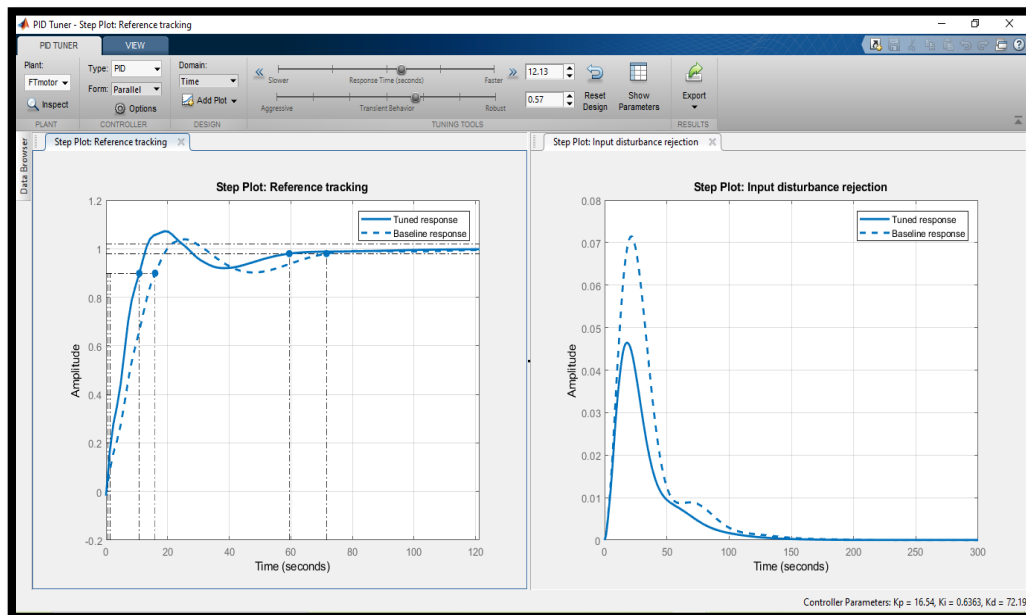
Nombre: Denisse Fiallos Cañarte

Proyecto de sistemas de control Embebidos



Al importar la variable se observa la gráfica de la función y añadimos otra grafica **Input disturbance rejection**.

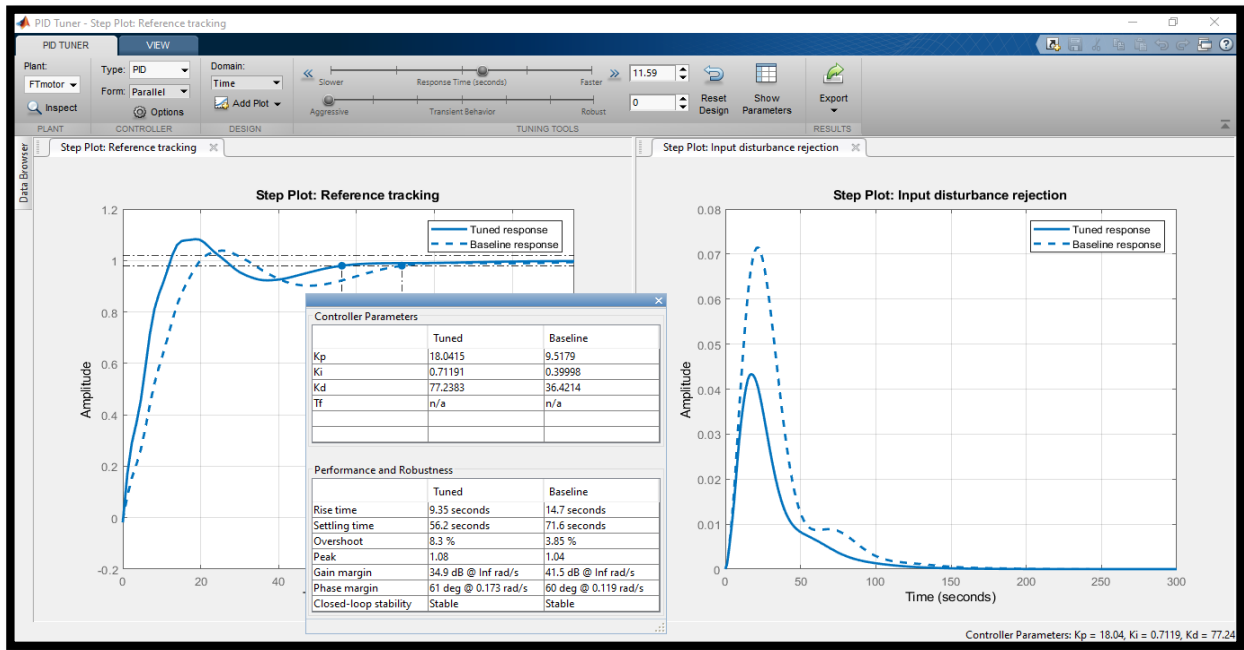
Al deslizar la barra **Response Time** observamos que las gráficas se desplazan y sus valores cambian según su posición. La principal idea es que la curva de **Input disturbance rejection** no sea ni muy grande ni muy pequeña. Y también es necesario mejorar **Rise time** incluyendo **Settling time** en la gráfica:



En la siguiente figura podemos observar la tabla de datos obtenida con el criterio mencionado

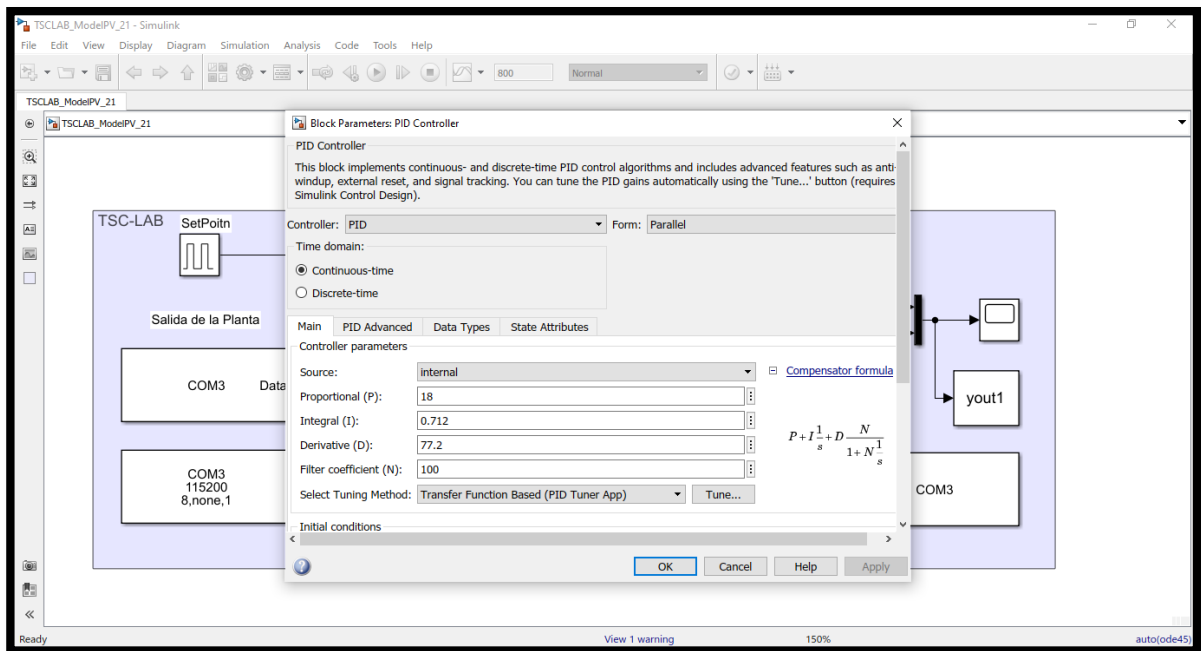
Nombre: Denisse Fiallos Cañarte

Proyecto de sistemas de control Embebidos



4. El controlador en lazo cerrado deberá ser comprobado con Matlab

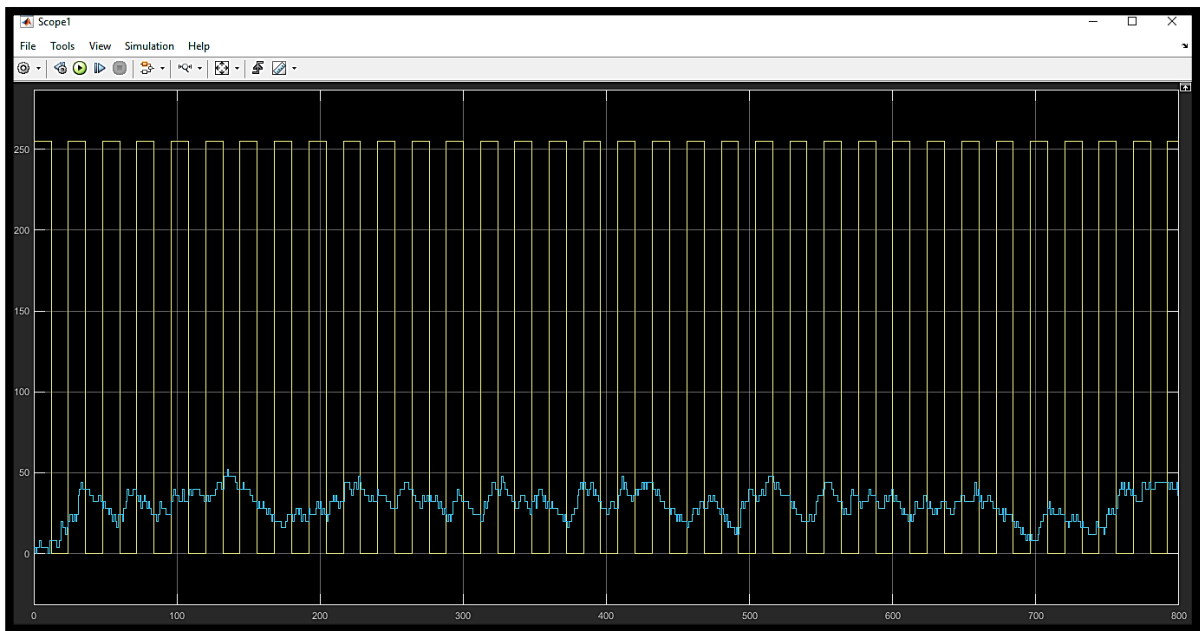
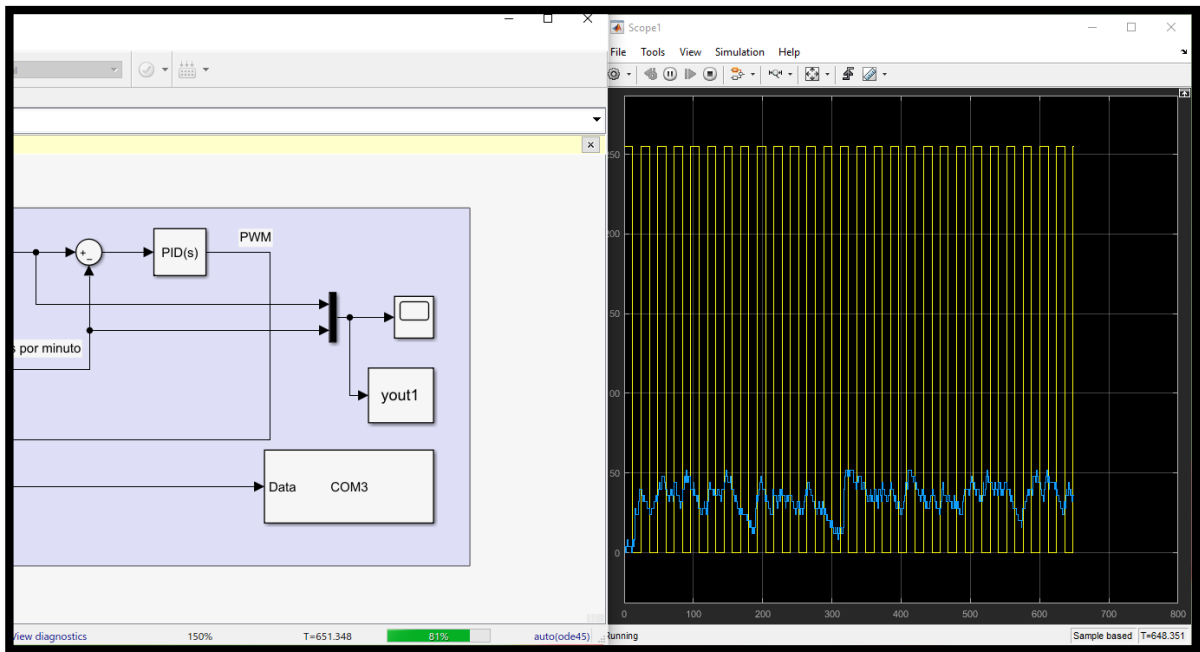
Con las constantes obtenidas en el PID Tuner la procedemos a evaluar en simulink



Le damos run en simulink y nos aparece la siguiente gráfica:

Nombre: Denisse Fiallos Cañarte

Proyecto de sistemas de control Embebidos

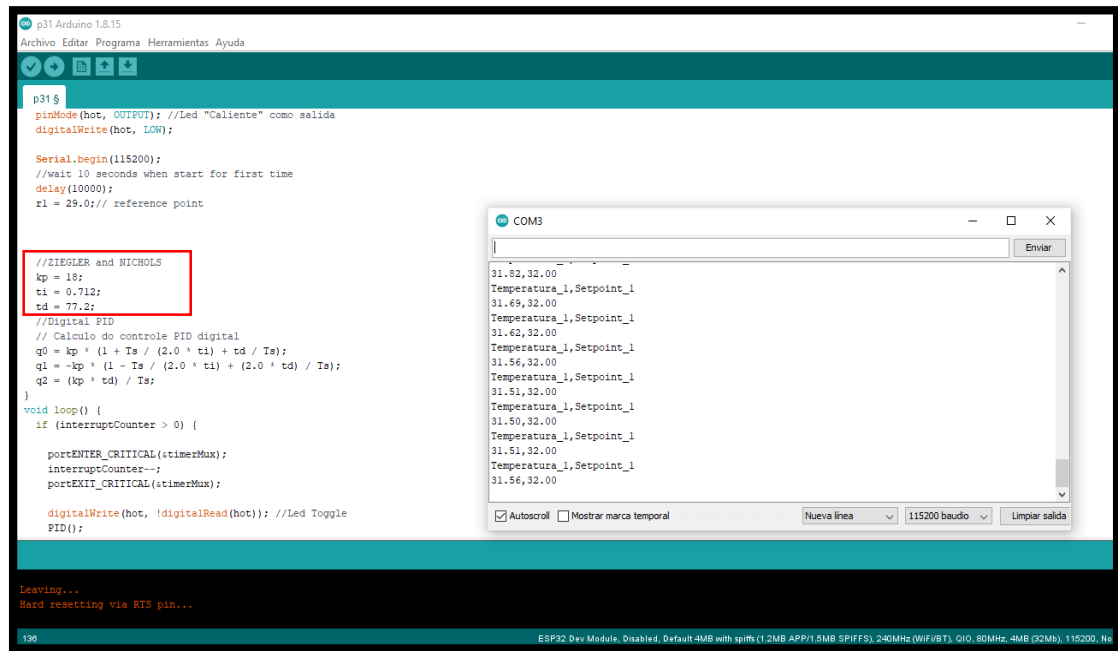


Nombre: Denisse Fiallos Cañarte

Proyecto de sistemas de control Embebidos

5. El controlador deberá ser implementado en el ESP32 (Adicional)

Procedemos a ingresar los valores de Kp, Ki, Kd que obtuvimos en el PID Tuner en la programación que se va a cargar en el TSC-Lab.



```
p31 $
pinMode(hot, OUTPUT); //Led "Caliente" como salida
digitalWrite(hot, LOW);

Serial.begin(115200);
//wait 10 seconds when start for first time
delay(10000);
r1 = 29.0; // reference point

//ZIEGLER and NICHOLS
kp = 18;
ti = 0.712;
td = 77.2;

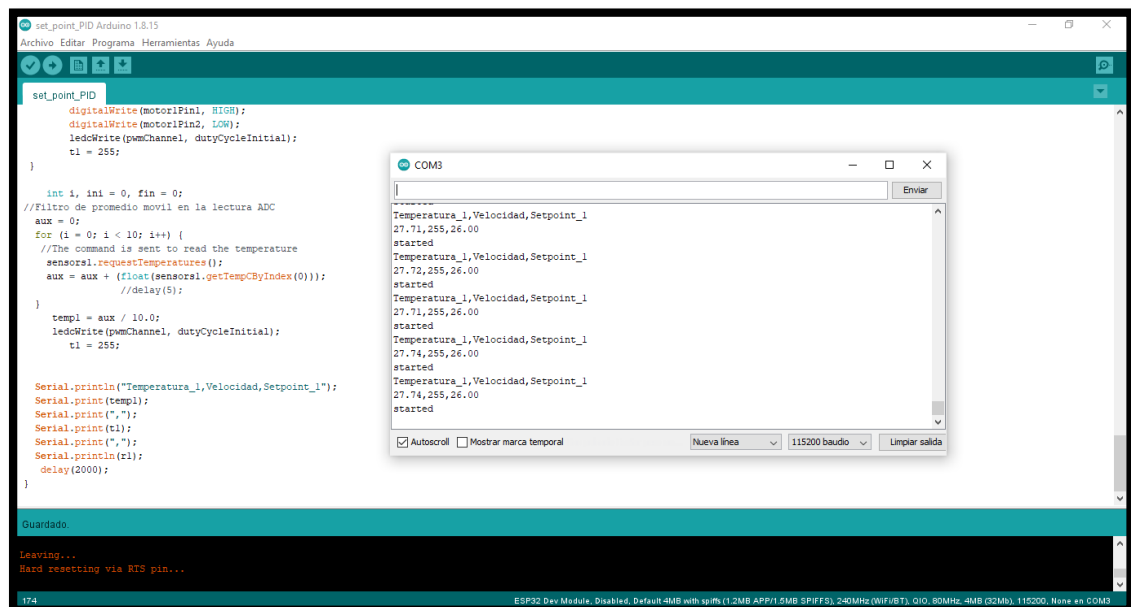
//Digital PID
// Calculo do controle PID digital
q0 = kp * (1 + Ts / (2.0 * ti) + td / Ts);
q1 = -kp * (1 - Ts / (2.0 * ti) + (2.0 * td) / Ts);
q2 = (kp * td) / Ts;

void loop() {
  if (interruptCounter > 0) {
    portENTER_CRITICAL(&timerMax);
    interruptCounter--;
    portEXIT_CRITICAL(&timerMax);

    digitalWrite(hot, !digitalRead(hot)); //Led Toggle
    PID();
  }
}
```

Leaving...
Hard resetting via RTS pin...

ESP32 Dev Module. Disabled. Default-4MB with spiffs (1.2MB APP/1.5MB SPIFFS), 240MHz (WiFi/BT), QIO, 80MHz, 4MB (Q2MB), 115200, No



```
set_point_PID
digitalWrite(motor1Pin1, HIGH);
digitalWrite(motor1Pin2, LOW);
ledcWrite(pwmChannel, dutyCycleInitial);
tl = 255;

int i, ini = 0, fin = 0;
//Filtro de promedio movil en la lectura ADC
aux = 0;
for (i = 0; i < 10; i++) {
  //The command is sent to read the temperature
  sensors1.requestTemperatures();
  aux = aux + (float(sensors1.getTempCByIndex(0)));
  //delay(5);
}
templ = aux / 10.0;
ledcWrite(pwmChannel, dutyCycleInitial);
tl = 255;

Serial.println("Temperatura_1,Velocidad,Setpoint_1");
Serial.print(templ);
Serial.print(",");
Serial.print(tl);
Serial.print(",");
Serial.println(zl);
delay(2000);
}
```

Guardado.

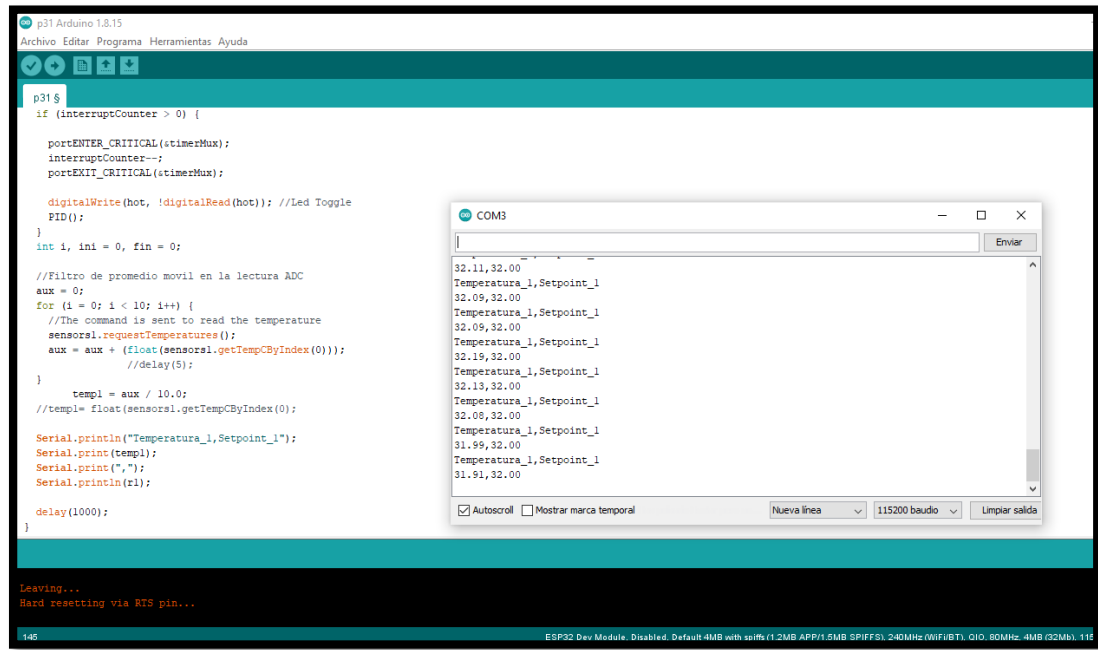
Leaving...
Hard resetting via RTS pin...

ESP32 Dev Module. Disabled. Default-4MB with spiffs (1.2MB APP/1.5MB SPIFFS), 240MHz (WiFi/BT), QIO, 80MHz, 4MB (Q2MB), 115200, None en COM3

Observamos que los valores mostrados en el monitor serial no pasan del setpoint ingresado (32 grados), para comprobarlo se pueden visualizar las siguientes figuras:

Nombre: Denisse Fiallos Cañarte

Proyecto de sistemas de control Embebidos

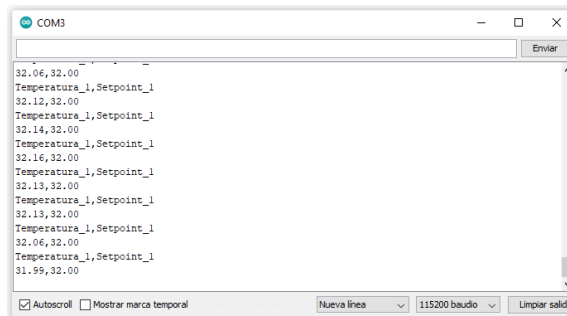


The screenshot shows the Arduino IDE interface. The main window displays a C++ program for temperature control. The program includes a serial monitor window (COM3) showing the output of the program. The output consists of pairs of temperature and setpoint values, such as "32.11,32.00" and "Temperatura_1,Setpoint_1".

```
p31 $  
if (interruptCounter > 0) {  
    portENTER_CRITICAL(&timerMax);  
    interruptCounter--;  
    portEXIT_CRITICAL(&timerMax);  
  
    digitalWrite(hot, !digitalRead(hot)); //Led Toggle  
    FID();  
}  
  
int i, ini = 0, fin = 0;  
  
//Filtro de promedio movil en la lectura ADC  
aux = 0;  
for (i = 0; i < 10; i++) {  
    //The command is sent to read the temperature  
    sensor1.requestTemperatures();  
    aux = aux + (float(sensor1.getTempCByIndex(0)));  
    //delay(5);  
}  
    temp1 = aux / 10.0;  
    //temp1= float(sensor1.getTempCByIndex(0));  
  
    Serial.println("Temperatura_1,Setpoint_1");  
    Serial.print(temp1);  
    Serial.print(",");  
    Serial.println(1);  
  
    delay(1000);  
}
```

Leaving...
Hard resetting via RTS pin...

ESP32 Dev Module. Disabled. Default 4MB with spiiffs (1.2MB APP/1.5MB SPIFFS). 240MHz (WiFi/BT). QIO. 80MHz. 4MB (32Mb). 115200 baud. 115200 baud. 115200 baud.

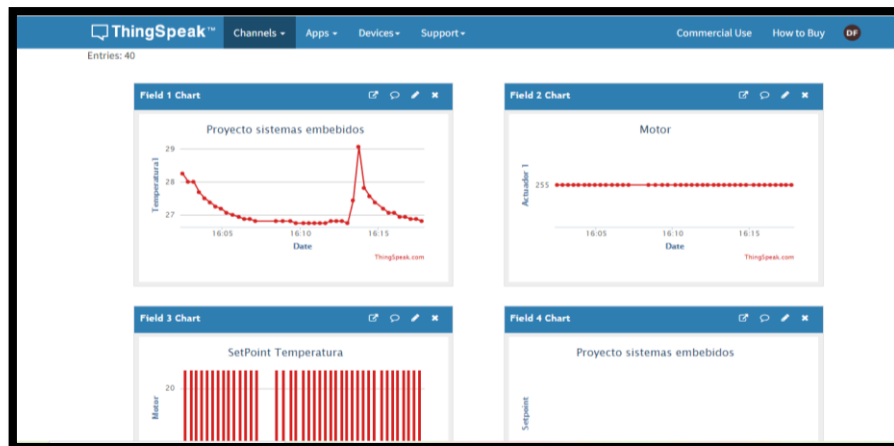


The screenshot shows the serial monitor window (COM3) displaying the output of the program. The output consists of pairs of temperature and setpoint values, such as "32.06,32.00" and "Temperatura_1,Setpoint_1".

```
32.06,32.00  
Temperatura_1,Setpoint_1  
32.12,32.00  
Temperatura_1,Setpoint_1  
32.14,32.00  
Temperatura_1,Setpoint_1  
32.16,32.00  
Temperatura_1,Setpoint_1  
32.13,32.00  
Temperatura_1,Setpoint_1  
32.13,32.00  
Temperatura_1,Setpoint_1  
32.06,32.00  
Temperatura_1,Setpoint_1  
31.99,32.00
```

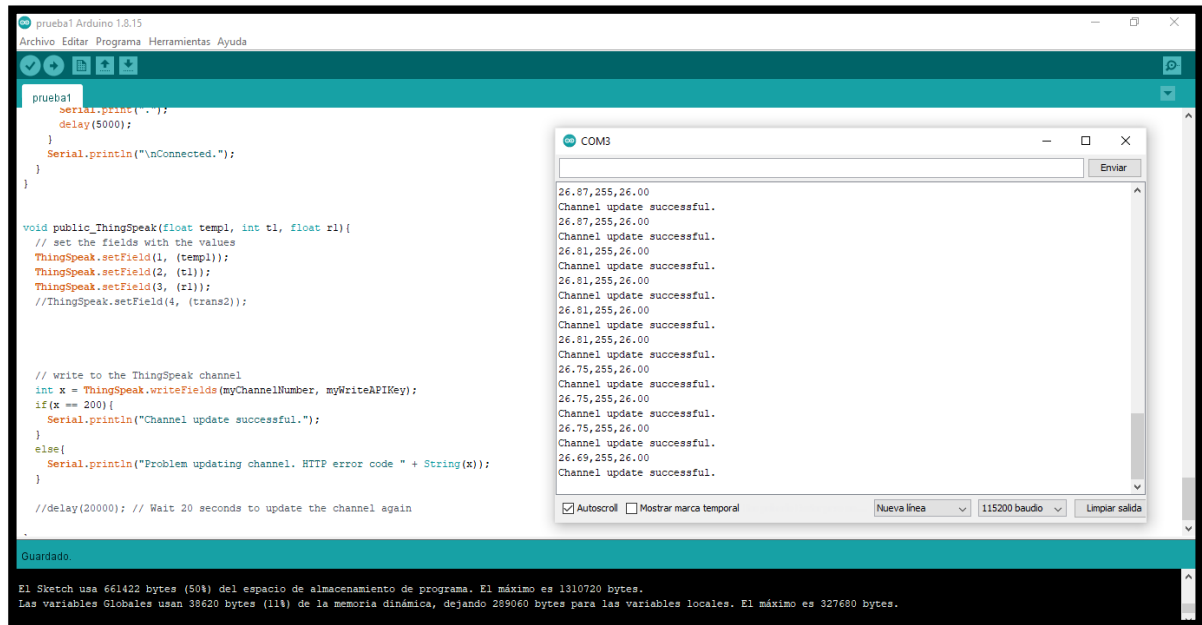
- Finalmente, el proyecto deberá incluir el envío de datos a una de las plataformas IoT vistas en clase

[Proyecto sistemas embebidos - ThingSpeak IoT](#)



Nombre: Denisse Fiallos Cañarte

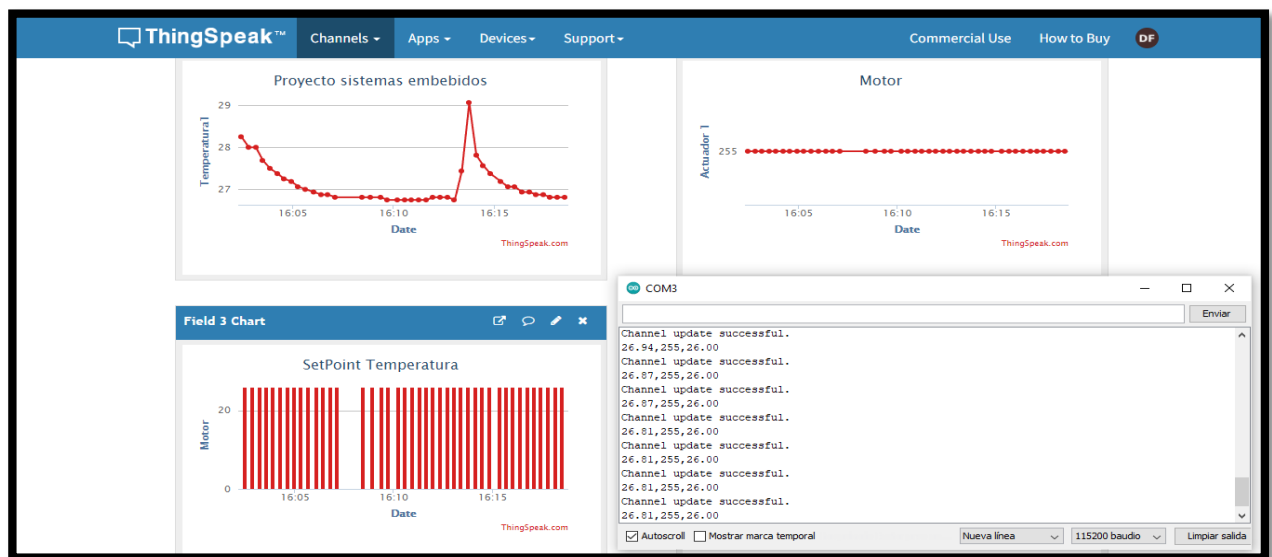
Proyecto de sistemas de control Embebidos



The screenshot shows the Arduino IDE interface. The sketch is named 'prueba1' and is written in C++. It includes a serial print statement to check connection status, a delay of 5000ms, and a function 'void public ThingSpeak' that sets fields with temperature, setpoint, and motor status. The main loop writes to the ThingSpeak channel and prints success or error messages. The serial monitor window shows the following data being transmitted:

```
26.87,255,26.00
Channel update successful.
26.87,255,26.00
Channel update successful.
26.81,255,26.00
Channel update successful.
26.81,255,26.00
Channel update successful.
26.81,255,26.00
Channel update successful.
26.81,255,26.00
Channel update successful.
26.75,255,26.00
Channel update successful.
26.75,255,26.00
Channel update successful.
26.75,255,26.00
Channel update successful.
26.69,255,26.00
Channel update successful.
```

Se realizó exitosamente el envío de datos a la plataforma IoT ThingSpeak, en el cual se observa la variación de temperatura mientras el motor está activo y el setpoint es 26 grados.



Podemos observar a través del monitor serial el envío de datos exitosamente hacia la plataforma.

Es necesario tener en cuenta las claves ID channel y el API KEY para que se establezca el enlace entre la tarjeta y la plataforma IoT thingSpeak.

Proyecto de sistemas de control Embebidos

7. Resultados, donde discutirán los resultados obtenidos en los puntos 4 y 5 (descritos arriba).

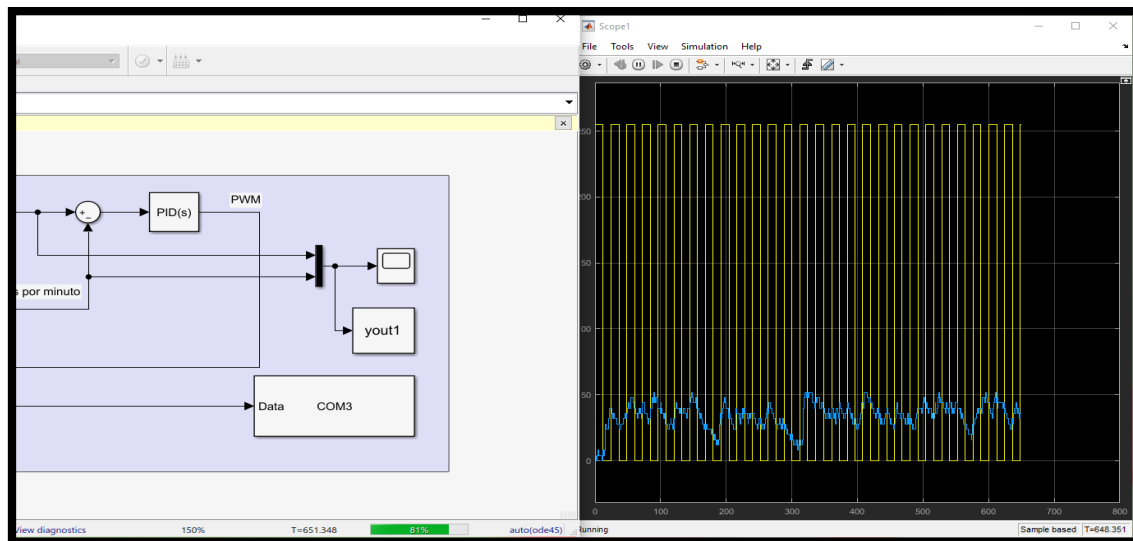
ITEM 4 – CONSLUSIONES

Se decidió diseñar el controlador PID con las siguientes consideraciones:

- El Settling time debe ser el más pequeño, ya que la curva debe estabilizarse lo más rápido posible así se podrá evaluar de mejor manera el setpoint del sistema.
- El análisis de la gráfica vs **Input disturbance rejection**, me permite observar que tan grande o pequeño puede ser las perturbaciones en el sistema. Lo ideal es que esta no sea ni muy grande ni muy pequeña, mantenerla en un valor.
- Es importante analizar el Rise Time debido a que está muy relacionado con el overshoot el cual nos indicara el porcentaje que se puede sobrepasar la estabilidad del sistema.

Ingresando los datos k_p , k_i , k_d al sistema en simulink tuve algunos problemas debido a que en mi versión de simulink no se encuentran los bloques indicados por el profesor. Logre que el simulink me grafique pero realmente no estoy muy segura de que si esta correcta o no la gráfica.

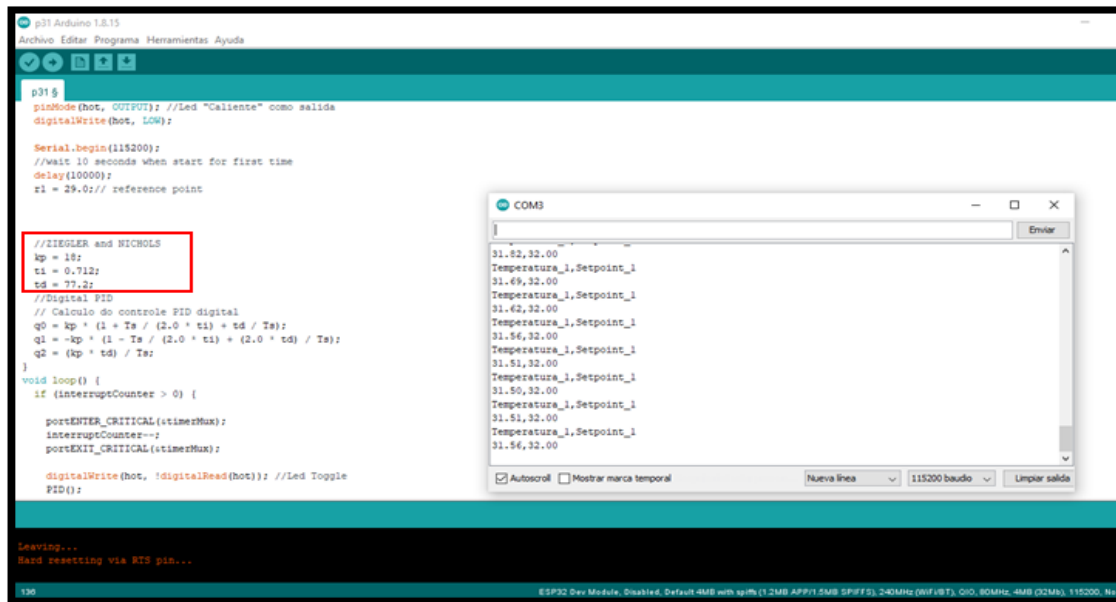
Lo que sí es importante recalcar que la gráfica del sistema se mantiene estable o dentro de los parámetros lo que permite que se pueda trabajar con el setpoint de manera correcta.



Proyecto de sistemas de control Embebidos

ITEM 5 - CONSLUSIONES

Se ingresaron los valores obtenidos de las constantes de PID y se cargó el programa indicado (ejemplo 31). Al abrir el monitor serial se observa que la temperatura varia hasta que llega al setpoint ingresado en el programa ('r') el cual, queda oscilando hasta un máximo de overshoot 8% luego procede a descender un 8% aproximado.



8. Conclusiones, incluir 6 como mínimo

Al realizar los ítems solicitados para realizar el proyecto podemos observar lo siguiente:

- Al momento de importar los datos obtenidos con la planta, si en la gráfica aparecen perturbaciones una manera de suavizarla es utilizar el comando smooth. El cual permite suavizar la gráfica para poder identificar el mejor modelo y así obtener constantes PID adecuadas para el sistema.
- Se escogió el modelo Bj44444 se llegó a la conclusión de ese modelo revisando la gráfica del modelo de salida en la cual se visualiza un BEST FIT de 87.99%.
- Una vez obtenida la función de transferencia se visualiza la gráfica directamente en el workspace de matlab y se observa que es bastante parecida a la gráfica estimada por ello se concluye que el modelo está correcto.

Nombre: Denisse Fiallos Cañarte

Proyecto de sistemas de control Embebidos

- Se utilizó PID Tuner para el análisis con la función de transferencia, esta herramienta permite tratar la señal y adecuarla según los criterios aprendidos en materias anteriores:
- El Settling time debe ser el más pequeño, ya que la curva debe estabilizarse lo más rápido posible así se podrá evaluar de mejor manera el setpoint del sistema.
- El análisis de la gráfica vs **Input disturbance rejection**, me permite observar que tan grande o pequeño puede ser las perturbaciones en el sistema. Lo ideal es que esta no sea ni muy grande ni muy pequeña, mantenerla en un valor.
- Es importante analizar el Rise Time debido a que está muy relacionado con el overshoot el cual nos indicara el porcentaje que se puede sobrepasar la estabilidad del sistema.
- Al ejecutar el programa ingresado las constantes de PID permite observar que el sistema funciona muy bien. Al momento de que la temperatura llega al setpoint esta se mantiene y en casos puede oscilar con un overshoot de entre 8 – 10%. El sistema diseñado de la planta es considerablemente estable.