UNIVERSIDAD DE CONCEPCIÓN

FACULTAD DE INGENIERÍA

DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

Informe Laboratorio de control e instrumentación

Laboratorio N°2: Modelación de Sistemas.

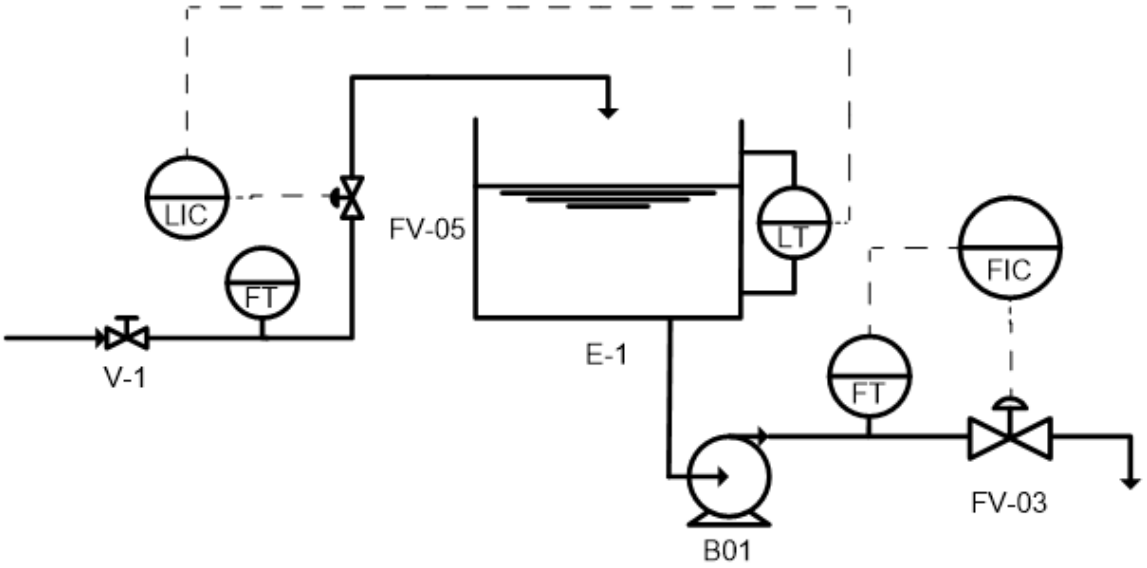
Jocelyn Matus Ancavil.
Rafael Burgos Yousuff.

Profesor Juan Pablo Segovia Vera.

Concepción, miércoles 28 de septiembre 2020.

Setups de la planta:

Lazo simple:



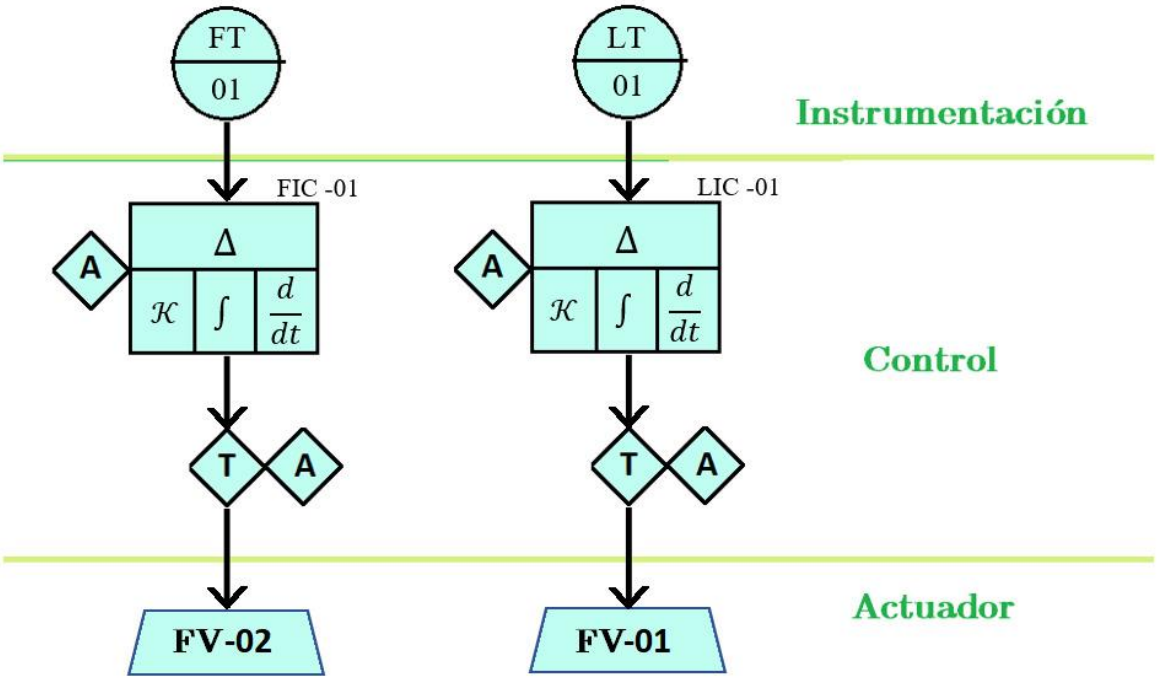


Diagrama SAMA de lazo de control simple

Lazo en cascada:

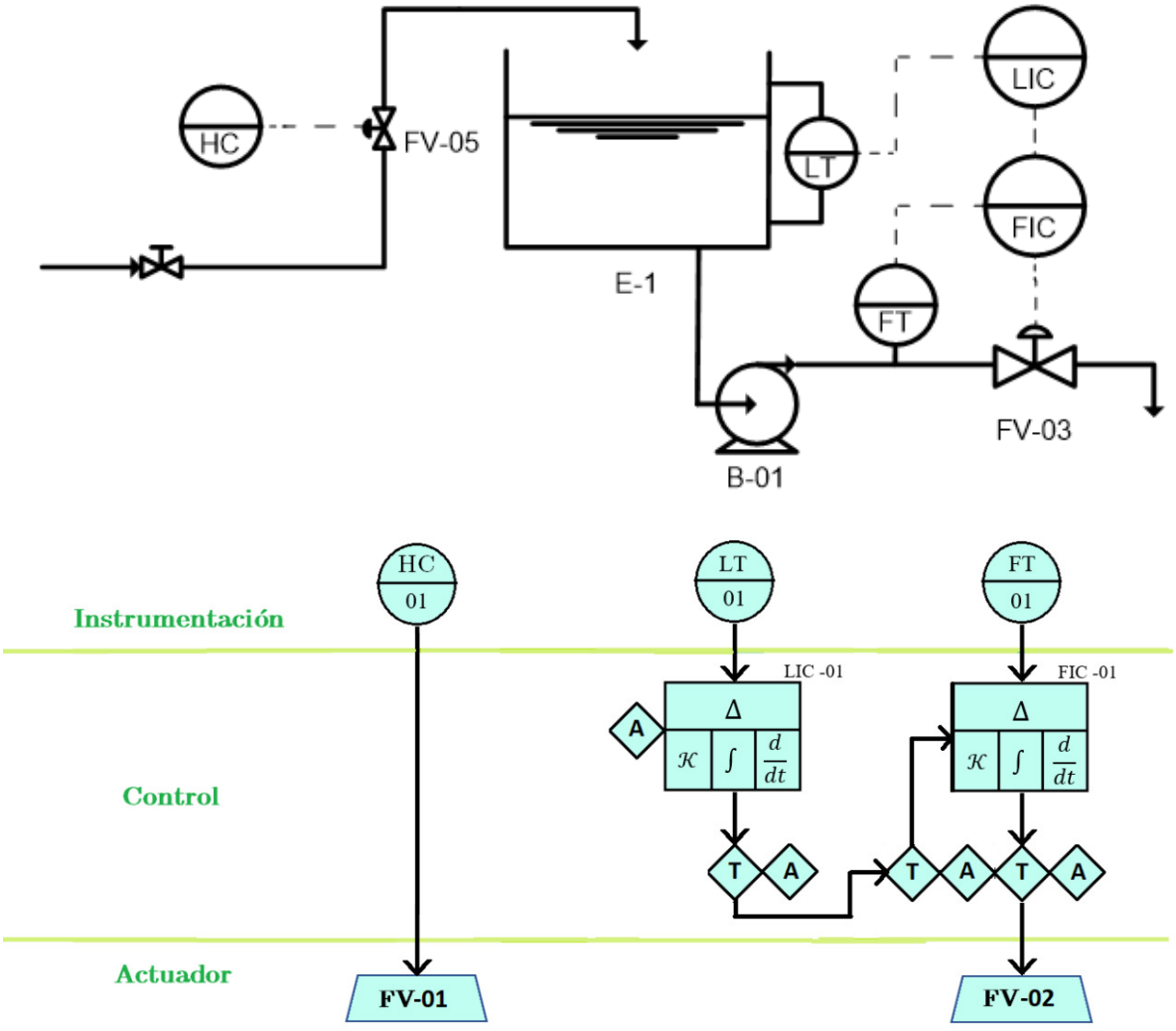


Diagrama SAMA de lazo de control en cascada.

Diagramas de bloques:

Diagrama de bloques en lazo simple:

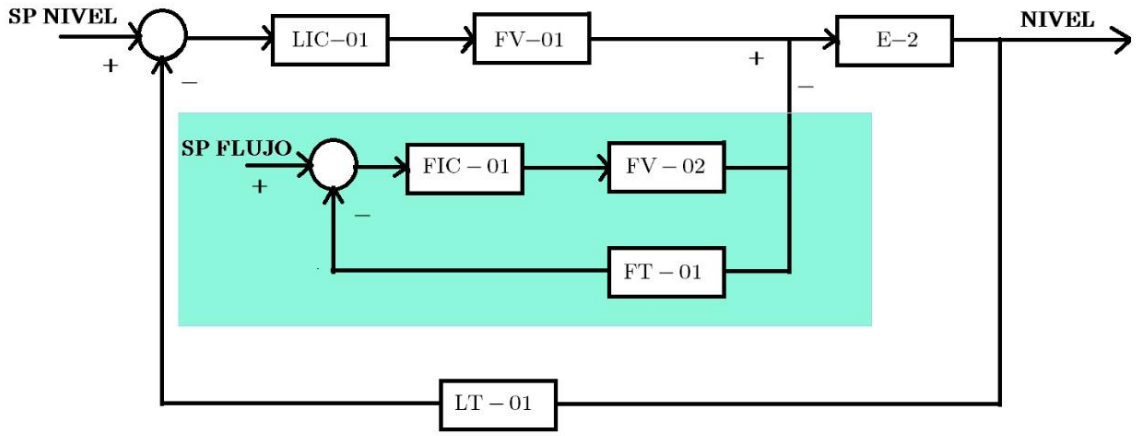
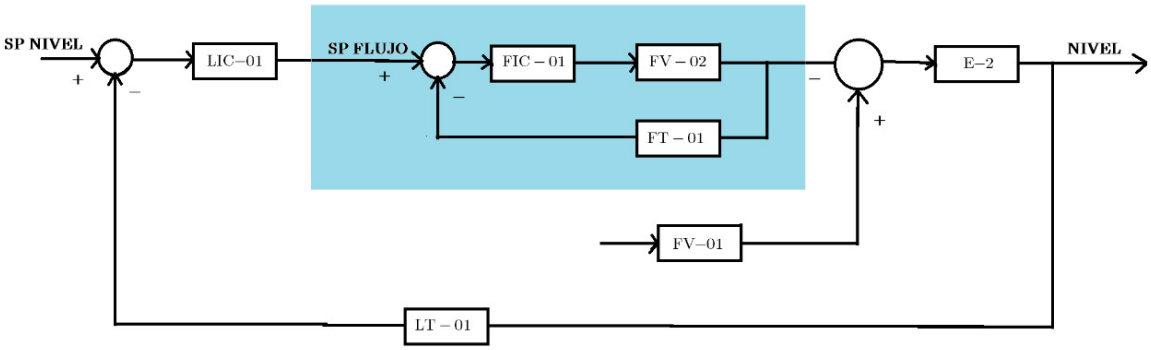


Diagrama de bloque en lazo configuración cascada:

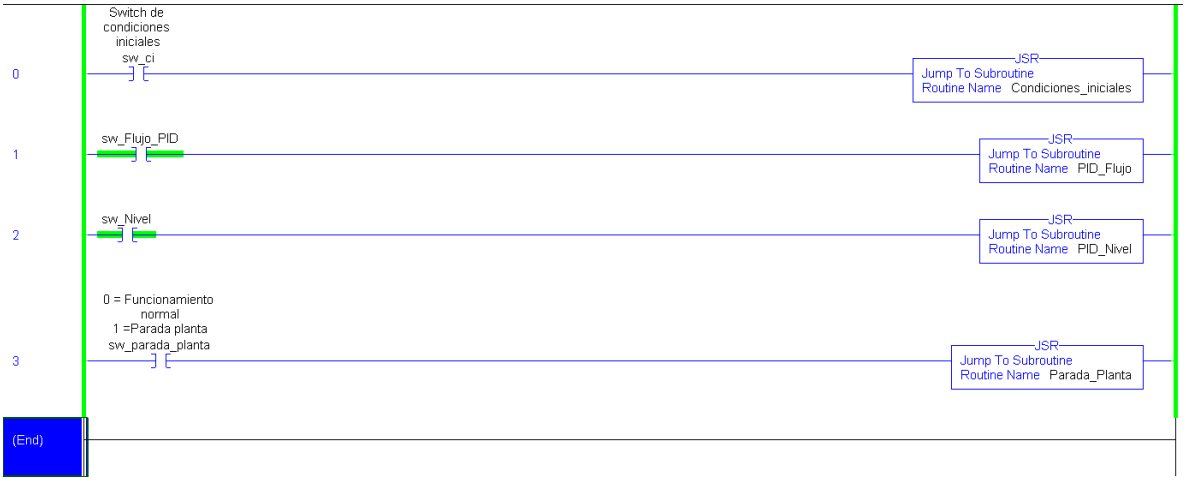


a) Programación y configuración para ambas estructuras de control sobre RSLogix 5000.(1.5 puntos)

En lo referente a la programación, esta se realizó en RSLogix 5000 a través de texto estructurado y lógica de escalera, en lo referente al controlador, mientras que se realizó una simulación de la planta con el programa Simulink, el cual fue conectado al RSLogix a través de un OPC.

Antes de mostrar los programas para cada configuración, se va a mostrar los textos estructurados que pueden, y son usados, para las ambas configuraciones, además de la lógica de escalera que se realizó.

Lógica de escalera:



Para que nosotros queramos que se inicie la simulación con ciertos valores iniciales, se tiene la subrutina Condiciones_iniciales, la cual es un texto estructurado.

```
sw_m_al:=0;
SW_GS:=0;

//Se van a definir las condiciones iniciales del controlador de la planta para el control de flujo y nivel

//Condiciones iniciales lazo de flujo

Tm_f:= 0.1;
Kc_f:= 1.8;
Ti_f:= 4;
Td_f:= 0;
sw_am_f:= 1;
sw_d_f:= 0;
sw_a_f:= 1;
nb_f:= 0.9;
dumax_f:= 20;
umax_f:= 100;
umin_f:= 0;

//Inicializacion estados pasados del lazo de flujo

ek1_f:= 0;
ek2_f:= 0;
uk1_f:= 0;
yk1_f:= 0;
yk2_f:= 0;

//Condiciones iniciales lazo de nivel

Tm_n:= 0.1;
Kc_n:= 6;
Ti_n:= 100000;
Td_n:= 0;
sw_am_n:= 1;
sw_a_n:= 1;
sw_d_n:= 0;
nb_n:= 0.9;
dumax_n:= 10;
umax_n:= 100;
umin_n:= 0;

//Inicializacion estados pasados del lazo de nivel

ek1_n:= 0;
ek2_n:= 0;
uk1_n:= 0;
yk1_n:= 0;
yk2_n:= 0;

//Condiciones iniciales de los switches del ladder

sw_Nivel:= 1;
sw_Flujo_PID:= 1;


sw_ci:= 0;
```

Se puede ver que, cuando se inicia la simulación, esta empieza con ciertos valores. Al final del código, se tiene la igualdad sw_ci = 0, para que ya se siga actualizando los valores a los

mismos que están en el texto condiciones_iniciales. Esto se puede ver en la imagen de lógica de escalera, donde el switch de condiciones iniciales está en 0

Además, existe un texto estructurado para parar la planta, el cual, como dice el texto, para la planta de manera segura cuando nosotros queramos, cambiando el bit de parada de planta (sw_parada_planta) de 0 a 1.

```
//Vaciar el estanque, con la planta en manual

sw_am_f:= 0;
sw_am_n:= 0;
FV_05:= 0;
FV_03:= 100;
FV_01:= 0;
FV_02:= 0;

//Apagar las bombas
```

Ahora, para el caso de lazo de flujo, se utilizó una configuración de controlador PI, la cual, como texto estructurado, es la siguiente:

```
if sw_m_al then sp_Flujo:= uk1_n + sign_n*abs(duk_n); end_if;

//Parametros del modelo PID discreto

q0_f:= Kc_f*(1+(Tm_f/(2*Ti_f))+(Td_f/Tm_f)*sw_d_f); //Los q estan relacionados con la derivada del error

q1_f:= -Kc_f*(1-(Tm_f/(2*Ti_f))+(2*Td_f/Tm_f)*sw_d_f);

q2_f:= (Kc_f*Td_f/Tm_f)*sw_d_f;

p0_f:= -(1-sw_d_f)*(Kc_f*Td_f/Tm_f); //Los p estan relacionados con la derivada de PV

p1_f:= (1-sw_d_f)*(2*Kc_f*Td_f/Tm_f);

p2_f:= -(1-sw_d_f)*(Kc_f*Td_f/Tm_f);

//Valores actuales de error y el PV(y(t))

yk_f:= FT_01; //Guarda lo que entrega el sensor de flujo

ek_f:= sp_Flujo-yk_f;

//Switch del accionamiento

ek_f:= sw_a_f*ek_f;

// Banda de proteccion contra el ruido
if ABS(ek_f) < nb_f then ek_f:= 0; end_if;

//Calculo variacion del accionamiento
duk_f:= q0_f*ek_f + q1_f*ek1_f + q2_f*ek2_f + p0_f*yk_f + p1_f*yk1_f + p2_f*yk2_f;
```

Al principio, en parámetros del modelo PID discreto, se discretiza el controlador PID, que cual, como se vio en clases, consta de 6 términos: 3 términos q0_f, q1_f, q2_f, basados en la acción derivativa en el error, y los 3 términos p0_f, p1_f, p2_f, los cuales están basados en la acción derivativa de la variable de proceso. La suma de estos 6 términos es igual al duk_f.

Nosotros podemos elegir si queremos basar la acción derivativa en el error, o en la variable del proceso, a través del valor de sw_d_f, el cual si es 0, la acción derivativa se basa en PV, usando solo los términos p0_f, p1_f y p2_f para el duk_f, y si es 1, la acción derivativa se basaría en el error, ocupando los términos q0_f, q1_f y q2_f para el duk_f.

Cabe destacar que, aunque aparezcan términos relacionados a la derivada, estos, al tener el parámetro Td_f = 0, no afectan en el controlador, teniendo solamente un controlador PI.

En valores actuales de error y el PV(y(t)), en el código, se guarda, en yk_f, lo que entrega el sensor de flujo en el tiempo actual, y se actualiza el error como el set point de flujo menos yk_f.

En el switch de accionamiento, se decide si el proceso se comporta como un proceso inverso (inc/dec) o un proceso directo (inc/inc).

En lo que se refiere a la banda de protección de ruido, se tiene que si el error no supera cierto valor (en este caso se propuso de 0.9), entonces el valor del error es de 0. Esto se realiza para que el controlador mantenga su salida ante cambios menores de error, para que no se genere desgaste mecánico por siempre estar accionando el controlador.

```
// Calculo del signo de duk
if duk_f>0 then sign_f:= 1;
ELSIF duk_f<0 then sign_f:= -1;
ELSE sign_f:= 0; end_if;

//Limitador de variacion (Proteccion del actuador)
 If abs(duk_f) >dumax_f then duk_f:= dumax_f; end_if;

//Calculo del uk_f

 if sw_am_f then uk_f:= uk1_f + sign_f*abs(duk_f); end_if; //Si esta en automatico, entonces el uk_f se actualiza al valor pasado de uk_f mas el duk_f


 // Limitador maximo y minimo del actuador (Proteccion del actuador)
 if uk_f < umin_f then uk_f:= umin_f; end_if;
 if uk_f >umax_f then uk_f:= umax_f; end_if;

 //Enviar resultado de accionamiento a la valvula correspondiente

 if sw_am_f then FV_03:= uk_f; //àqui, si esta el switch en modo automatico, entonces
                               // se lleva a la valvula lo que se calculo en uk_f
 ELSE FV_03:= FV_03; sp_Flujo:= FT_01; end_if; //Si esta en manual, nosotros elegimos
                                               // à que valor estara la valvula, y
                                               //el set point requerido para esto
                                               //se actualizara con
                                               //el sensor correspondiente
```

En lo que se refiere al cálculo del signo del duk, ya que el RSLogix no tiene una función matemática implementada para poder calcular el signo de variables reales, entonces se creó una mini función para eso, para calcular el signo del duk.

En lo referente al limitador de variación, se usa para la protección del actuador, y también para que no se genere un cambio escalón de más del 10% del valor de uk, para que el sistema no sufra de un cambio tan brusco de amplitud.

Después, pasando por el cálculo del uk_f, se tiene el limitador máximo y mínimo del actuador, el cual define los límites de acción que tiene el actuador. En general, uno no espera, de manera realista, que una válvula se abra más allá que de su máxima capacidad.

Por esto, si el actuador, a través de sus cálculos matemáticos, dice que debe entregar un valor mayor a 100%, el cual es el valor máximo de la válvula, este se limita a cierto valor, que, en este caso, es el 100%. Lo mismo para el limitador mínimo, el cual esta calculado para un 0%.

Después de todo esto, se envía los datos de lo calculado al accionamiento. Si el controlador está en modo automático, se envía lo que se calculó en uk, y si está en manual, el usuario elige a que valor la válvula va a estar.

```
//Memoria Actuador
uk1_f:= uk_f;

//Memoria error
ek2_f:= ek1_f;
ek1_f:= ek_f;

// Memoria PV
yk2_f:= yk1_f;
yk1_f:= yk_f;
```

Ya que RSLogix repite el código desde arriba hacia abajo, cada vez que pasa un ciclo de muestreo, no se puede guardar de manera automática los valores pasados de los cálculos. Es por esto que se crean comandos de códigos que guardan los valores cada vez que se ejecuta el código.

Ahora, el texto estructurado para el PID de nivel es el siguiente:

```
if sw_m_al then sw_a_n:= -1; //Esto es para configuracion cascada
else sw_a_n:= 1; end_if;

//Parametros del modelo PID discreto

q0_n:= Kc_n*(1+(Tm_n/(2*Ti_n))+(Td_n/Tm_n)*(sw_d_n)); //Los q estan relacionados con la derivada del error

q1_n:= Kc_n*(-1+(Tm_n/(2*Ti_n))+(2*Td_n/Tm_n)*(sw_d_n));

q2_n:= (Kc_n*Td_n/Tm_n)*sw_d_n;

p0_n:= -(1-sw_d_n)*(Kc_n*Td_n/Tm_n); //Los p estan relacionados con la derivada de PV

p0_n:= (1-sw_d_n)*(2*Kc_n*Td_n/Tm_n);

p1_n:= -(1-sw_d_n)*(Kc_n*Td_n/Tm_n);

//Valores actuales de error y el PV(y(t))

yk_n:= LT_01; //Guarda lo que envia el sensor de nivel

ek_n:= sp_Nivel-yk_n;

// Switch de accionamiento

ek_n:= sw_a_n*ek_n;

// Banda de proteccion contra el ruido
if ABS(ek_n) < nb_n then ek_n:= 0; end_if;

//Calculo variacion del accionamiento
duk_n:= q0_n*ek_n + q1_n*ek1_n + q2_n*ek2_n + p0_n*yk_n + p0_n*yk1_n + p1_n*yk2_n;
```

```
// Calculo del signo de duk
if duk_n>0 then sign_n:= 1;
ELSIF duk_n<0 then sign_n:= -1;
ELSE sign_n:= 0; end_if;

//Limitador de variacion (Proteccion del actuador)
 If abs(duk_n) >dumax_n then duk_n:= dumax_n; end_if;

//Calculo de uk_n
 if sw_am_n then uk_n:= uk1_n + sign_n*abs(duk_n); end_if; //Si el switch esta en automatico, entonces se actualiza el uk_n al valor anterior de uk_n mas el duk_n

 // Limitador maximo y minimo del actuador (Proteccion del actuador)
 if uk_n < umin_n then uk_n:= umin_n; end_if;
 if uk_n >umax_n then uk_n:= umax_n; end_if;

 //Enviar resultado de accionamiento a la valvula correspondiente

if sw_am_n AND NOT sw_m_al then FV_05:= uk_n; // Configuracion simple
elsif sw_am_n AND sw_m_al then FV_05:= FV_05; //Configuracion cascada
else FV_05:= FV_05; sp_Nivel:= LT_01; end_if; //Control manual con set point tracking

 //Memoria Actuador
 uk1_n:= uk_n;

 //Memoria error
 ek2_n:= ek1_n;
 ek1_n:= ek_n;

 // Memoria PV
 yk2_n:= yk1_n;
 yk1_n:= yk_n;
```

De la misma manera que para el texto estructurado de flujo, se tiene el mismo código, pero esta vez esta modificado para los valores que le corresponden al lazo de nivel.

Aunque, para poder realizar la estructura de control de cascada, se tuvo que realizar modificaciones al código, las cuales son distintas para el código de lazo de flujo y el de lazo de nivel. Las diferencias son las siguientes:

Texto estructurado de PID Nivel:

```
if sw_m_al then sw_a_n:= -1; //Esto es para configuracion cascada
else sw_a_n:= 1; end_if;
```

Estas líneas de código, que se encuentran al principio del texto estructurado, se verifica, a través del tag sw_m_al, si se quiere tener una estructura de control simple o en cascada. En el caso de que se quiera un control en cascada, el accionamiento del proceso se establece como directo, porque, si se pusiera inverso, la ganancia que recibiría el error seria positiva, creando una alimentación positiva, siendo que se necesita una negativa.

En el caso contrario que se quiera trabajar en lazo con estructura simple, el accionamiento del proceso se trabaja de manera inversa.

```
 //Enviar resultado de accionamiento a la valvula correspondiente

if sw_am_n AND NOT sw_m_al then FV_05:= uk_n; // Configuracion simple
elsif sw_am_n AND sw_m_al then FV_05:= FV_05; //Configuracion cascada
else FV_05:= FV_05; sp_Nivel:= LT_01; end_if; //Control manual con set point tracking
```

En lo referente a este código, que se encuentra casi al final del texto estructurado de lazo de nivel, este indica que va a pasar con los resultados calculados que serán enviados al accionamiento. En el caso de que se esté trabajando el controlador de modo automático y en lazo simple, los resultados que se van a enviar va a ser los calculados en uk_k.

Si se tiene que se está trabajando en modo automático y se quiere trabajar en configuración cascada, entonces se tiene que la válvula FV_05 se trabaja de manera manual, lo cual es un requerimiento para poder trabajar en configuración cascada.

Si no fuera ninguno de los casos anteriores, entonces el controlador está actuando de manera manual con set point tracking.

En lo siguiente esta la modificación que se realizó en el código de lazo de flujo:

```
if sw_m_al then sp_Flujo:= uk1_n + sign_n*abs(duk_n); end_if;
```

Si se está trabajando en modo cascada, entonces el set point de flujo va a ser igual uk_n más duk_n, lo cual es igual a lo que calculo el controlador de nivel en modo automático; o sea, los valores calculados del controlador PID discretizado son enviados al set point de flujo.

b) Factory Acceptance Test (FAT) para control de flujo y nivel del sistema, mediante conexión RS Logix 5000 vía OPC a simulink, utilización de modelos obtenidos en laboratorio N°1.(1.5 puntos)

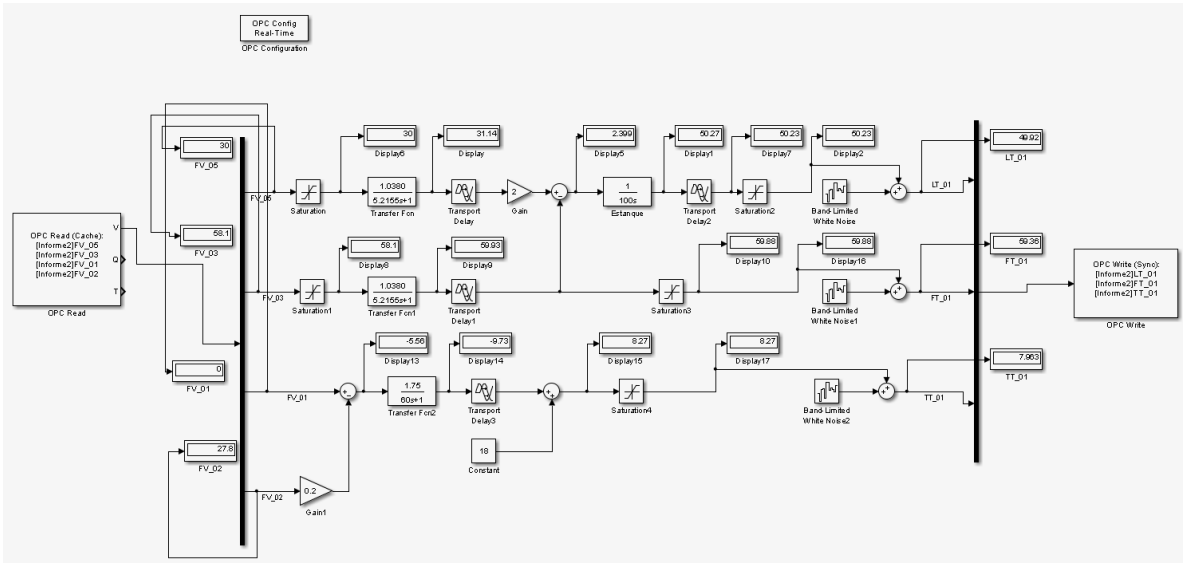
Primero que nada, los valores de la función de trasferencia para ambas válvulas son las siguientes:

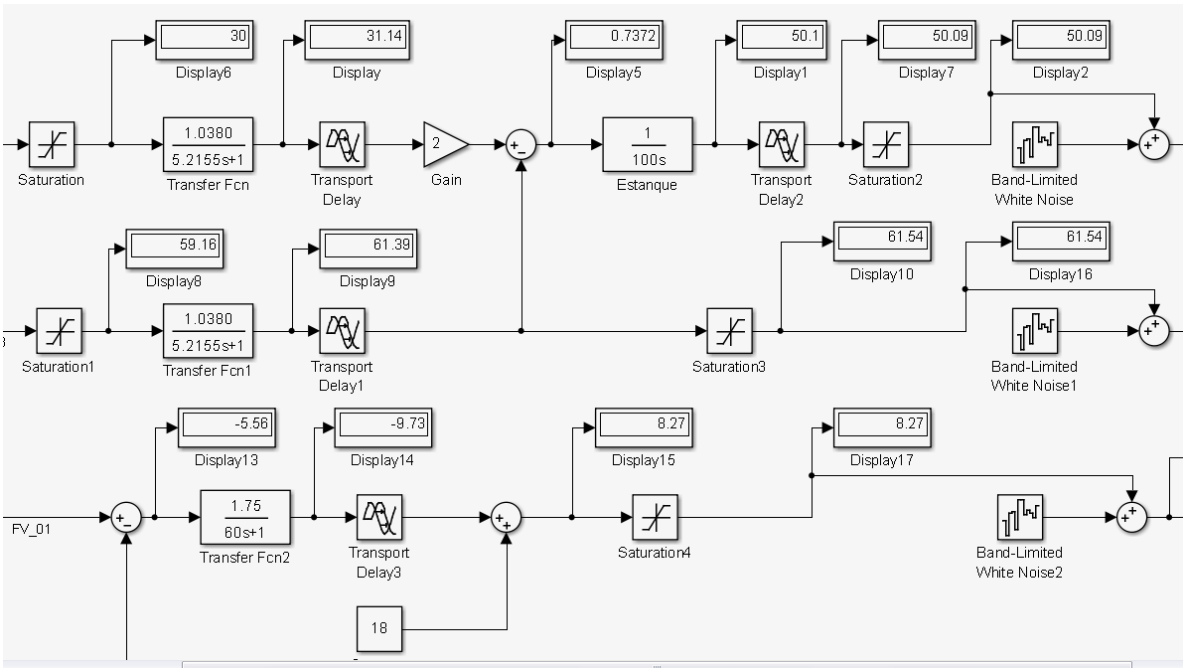
Ganancia = 1.038

Tau (constante de tiempo)= 5.2155

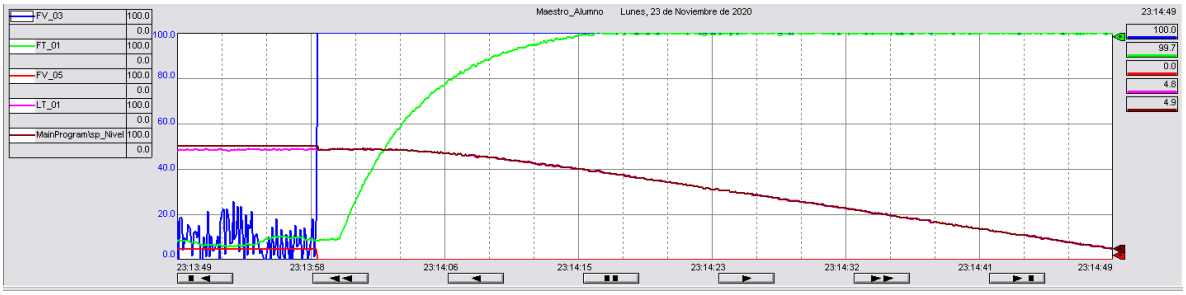
Retardo = 0.6 [s]

Ahora, se muestra como es la planta que se realizó en simulink para la simulación de esta:





Aquí esta la realización del código de parada de planta



Ahora, se va a mostrar en la misma gráfica, como funcionan los lazos de fujo y de nivel

Foto flujo manual, nivel manual

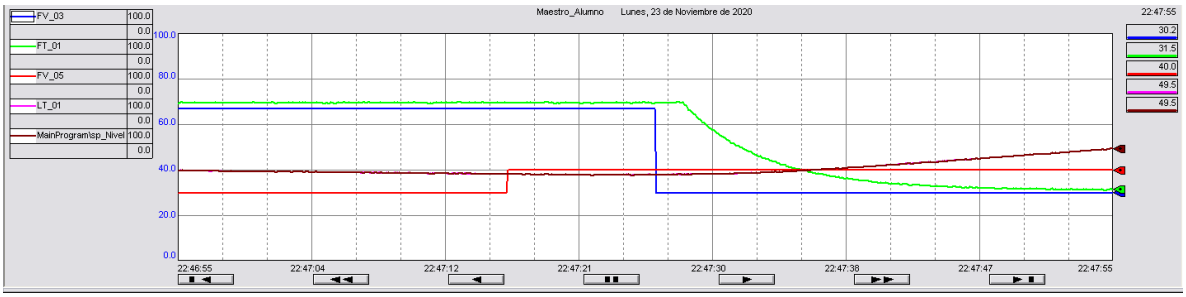


Foto flujo automático, nivel manual

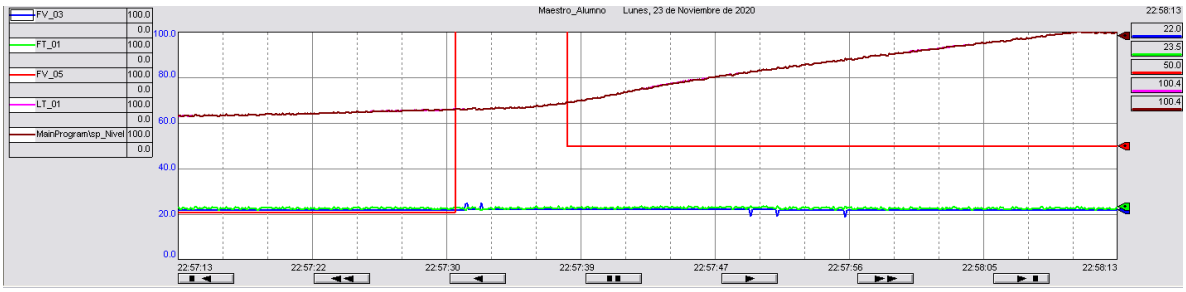


Foto flujo manual, nivel automático

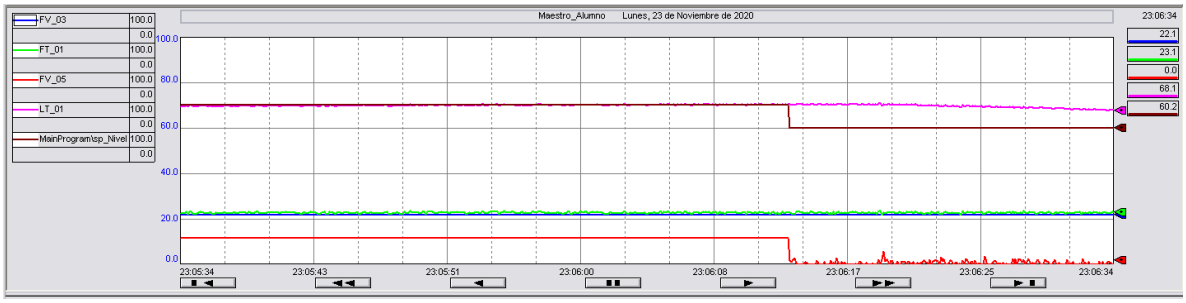


Foto flujo automático, nivel automático

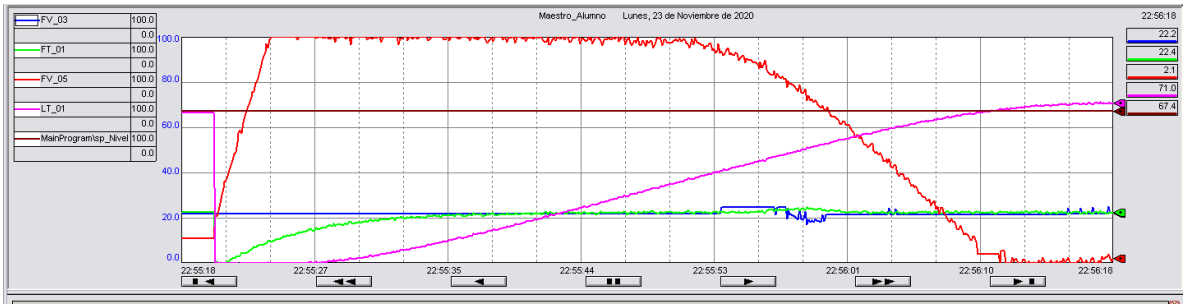
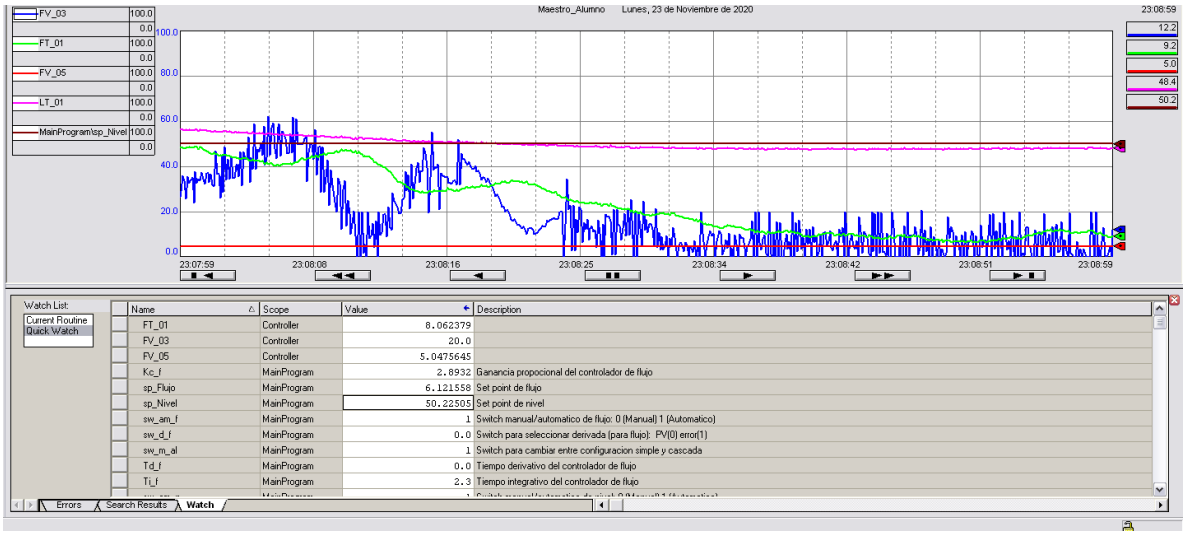


Foto operación en cascada



c) Contrastar para diversos métodos como; root locus, Sintonización en lazo abierto, lazo cerrado y método prueba y error (Ultimate).(1 ptos)

Para las sintonizaciones, primero se ve para el nivel de flujo, y este en configuración simple

Lazo Nivel:

Métodos de Ajuste de lazo abierto:

Para todos estos métodos, los valores del controlador se calculan desde la respuesta del sistema a un escalón. De esta respuesta en lazo abierto, se obtienen los valores del sistema en lazo abierto (ganancia, tau y retardo), y, desde estos valores, dependiendo del método, se obtiene los parámetros del controlador.

En este caso, por el lab n°1, se tienen los valores de ganancia, tau y retardo, los cuales son los siguientes:

Tau = 5.2155

Ganancia = 1.038

Retardo = 0.6

Y, dependiendo del método, los valores que se tendrán para el controlador serán distintos.

Antes de seguir adelante, se eligió, para el lazo de flujo, un controlador PI, por lo tanto, los cálculos de estos métodos están basados para un controlador PI.

ZG:

Ahora, para el método de Zieger Nichols, se utilizó las siguientes ecuaciones, las cuales fueron proporcionadas a través de recursos entregados en el curso:

Proportional + Reset	$KK_c = 0.9\,(t_0/\tau)^{-1.0}$ $T_i/\tau = 3.33\,(t_0/\tau)$

Lo cual, en Matlab, se tradujo a lo siguiente:

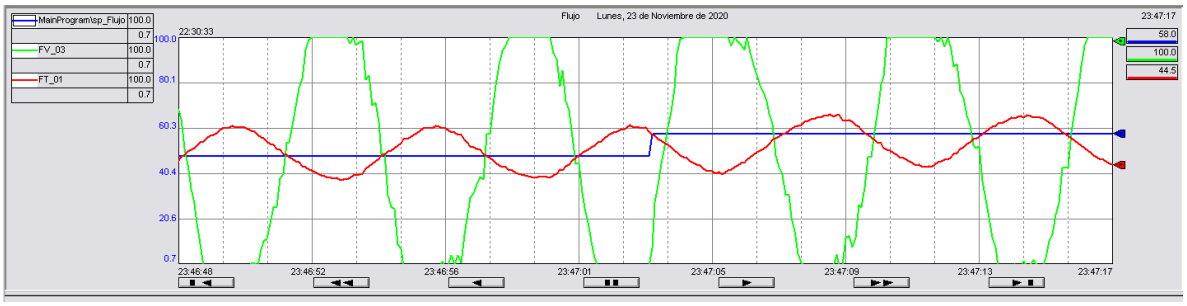
```
Kc_ZN= (0.9/ganancia)*((t0/tau)^(-1))

Ti_ZN= tau*3.33*(t0/tau)
```

Lo cual dio lo siguiente:

Kc = 7.5368

Ti = 1.998

Entonces, con estos valores, se realizo la respuesta a escalon:



La cual oscila indeterminadamente.

CC:

Para el caso de Cohen-Coon, se tiene lo siguiente:

$$KK_c = 0.9\,(t_0/\tau)^{-1.0} + 0.082$$

$$T_i/\tau = \frac{3.33\,(t_0/\tau)[1 + (t_0/\tau)/11.0]}{1.0 + 2.2\,(t_0/\tau)}$$

Lo cual se traduce a lo siguiente en Matlab:

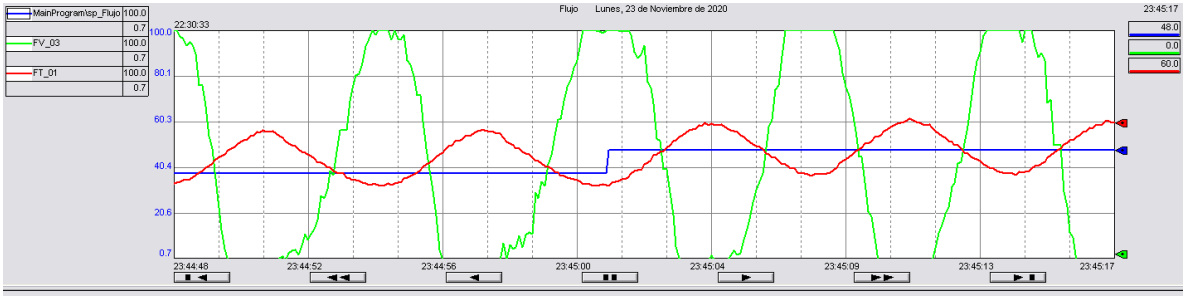
```
% Cohen-Coon

Kc_CC= (1/ganancia)*(0.9*((t0/tau)^-(1)) + 0.082)

Ti_CC= tau*((3.33*(t0/tau)*(1+((t0/tau)/11)))/(1+(2.2*(t0/tau))))
```

Lo cual da como resultado lo siguiente:

Kc = 7.6158

Ti = 1.6111

Entonces, utilizando estos valores, se tiene lo siguiente en la simulación:



De la misma manera que antes, oscila constantemente

3C:

Para el caso 3C, se tiene lo siguiente:

$$KK_c = 0.928\,(t_0/\tau)^{-0.946}$$

$$T_i/\tau = 0.928\,(t_0/\tau)^{0.583}$$

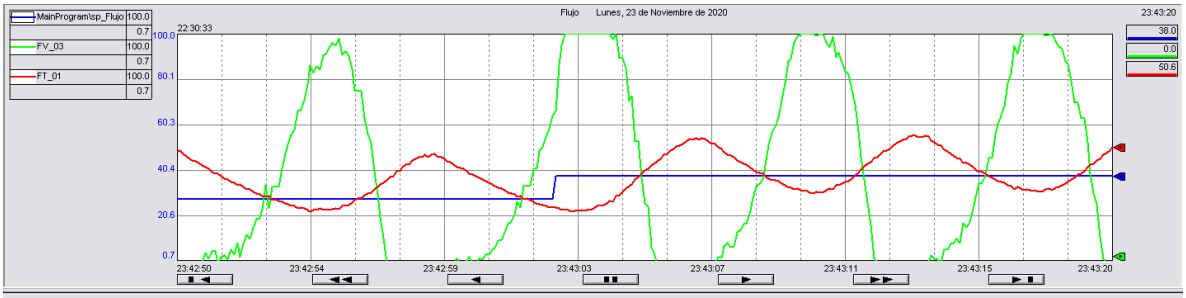
Lo cual traducido en Matlab es lo siguiente:

```
% 3C

Kc_3C= (0.928/ganancia)*((t0/tau)^(-0.946)

Ti_3C= (0.928*tau)*((t0/tau)^(0.583))
```

Lo cual da como resultado:

Kc = 6.9148

Ti = 1.3719



Lamentablemente, oscila indefinidamente, de igual manera que los dos anteriores.

Métodos basados en criterios integrados:

Los parámetros del controlador son basados de tal manera que el error satisfaga algún criterio de integral de error.

Antes de ver cada caso en particular, en todos los casos se tiene lo siguiente, para conseguir los parámetros requeridos, se tiene la siguiente ecuación:

$$Y = A(t_0/\tau)^B$$

La cual, dependiendo del parámetro que se quiera obtener, se tiene lo siguiente:

Y = K*Kc, para obtener el parámetro de ganancia.

Y= Tau/Ti, para obtener el parámetro Ti

Y= Td/Tau, para obtener el parámetro Td

A y B son valores fijos que se obtienen de cierta tabla

K = Ganancia del proceso.

Para los valores A y B, se obtienen de la siguiente tabla:

CONTROLADOR	CRITERIO	MODO	A	B
P	IAE	P	0,902	-0,985
	ISE	P	1,411	-0,917
	ITAE	P	0,490	-1,084
PI	IAE	P	0,984	-0,986
		I	0,608	-0,707
	ISE	P	1,305	-0,959
		I	0,492	-0,739
	ITAE	P	0,859	-0,977
		I	0,674	-0,680
PID	IAE	P	1,435	-0,921
		I	0,878	-0,749
		D	0,482	1,137
	ISE	P	1,495	-0,945
		I	1,101	-0,771
		D	0,560	1,006
	ITAE	P	1,357	-0,947
		I	0,842	-0,738
		D	0,381	0,995

IAE:

En este caso, es la integral del valor absoluto del error:

$$IAE = \int_0^\infty [Y(t) - r(t)]\, dt$$

Y, ya que estamos con un controlador PI, se tiene que los valores A y B son los siguientes:

IAE	P	0,984	-0,986
	I	0,608	-0,707

Para el cálculo de cada parámetro, se realizó lo siguiente en Matlab:

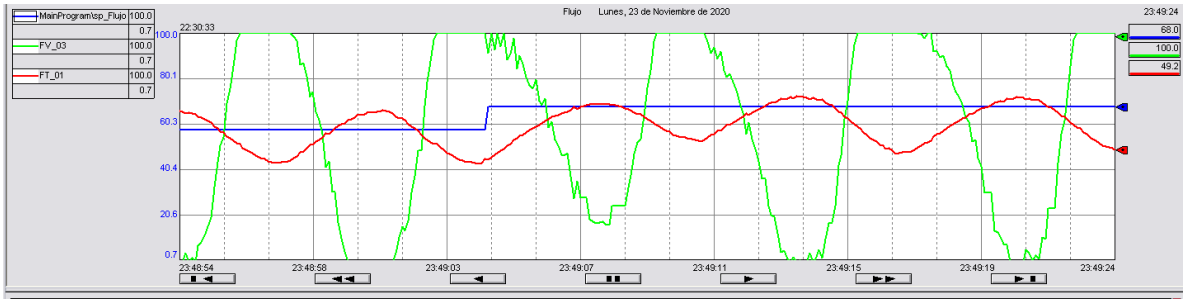
```
% IAE

A_IAE_P= 0.984;
B_IAE_P= -0.986;

A_IAE_I= 0.608;
B_IAE_I= -0.707;


Kc_IAE= (A_IAE_P/ganancia)*((t0/tau)^(B_IAE_P))

Ti_IAE= tau/(A_IAE_I*((t0/tau)^(B_IAE_I)))
```

Lo cual da como resultado lo siguiente:

Kc = 7.9945

Ti = 1.8595

Con estos valores de parámetros, se procede a usarlos y simularlos:



Lamentablemente oscila.

ISE:

Para este caso, es la integral del error cuadrático:

ISE: Integral del error cuadrático.

$$ISE = \int_0^\infty [Y(t) - r(t)]^2 \, dt$$

Y, ya que queremos calcular los parámetros para un controlador PI, se tiene los siguientes valores para A y B:

ISE	P	1,305	-0,959
	I	0,492	-0,739

Para el cálculo de los parámetros, se tiene lo siguiente en Matlab:

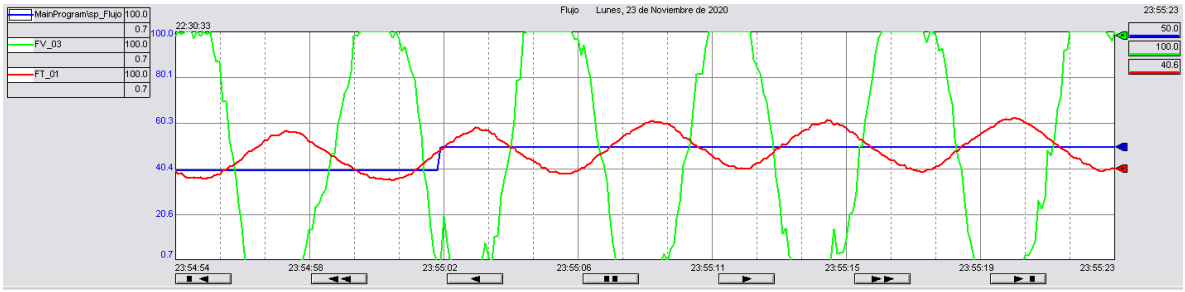
```
% ISE

A_ISE_P= 1.305;
B_ISE_P= -0.959;

A_ISE_I= 0.492;
B_ISE_I= -0.739;


Kc_ISE= (A_ISE_P/ganancia)*((tO/tau)^(B_ISE_P))

Ti_ISE= tau/(A_ISE_I*((tO/tau)^(B_ISE_I)))
```

Lo cual da como resultado lo siguiente:

Kc = 10.0012

Ti = 2.1443

Y actualizando estos valores en la simulación se tiene lo siguiente:



Nuevamente oscila.

ITAE:

Para este caso, el criterio es el de la integral del producto entre el valor absoluto del error y el tiempo:

$$ITAE = \int_0^{\infty} t\,[Y(t) - r(t)]\,dt$$

Para el cálculo correspondiente a cada parámetro, se tiene que los valores de A y B son los siguientes:

ITAE	P	0,859	-0,977
	I	0,674	-0,680

Entonces, en Matlab se tiene el siguiente código:

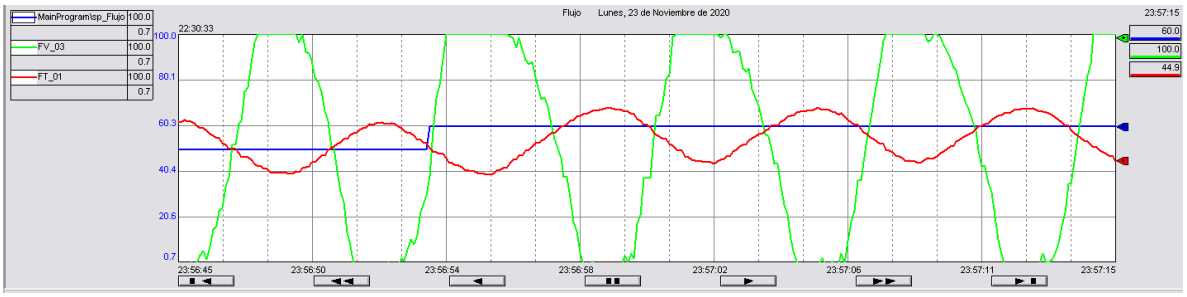
```
% ITAE

A_ITAE_P= 0.859;
B_ITAE_P= -0.977;

A_ITAE_I= 0.674;
B_ITAE_I= -0.680;



Kc_ITAE= (A_ITAE_P/ganancia)*((t0/tau)^(B_ITAE_P))

Ti_ITAE= tau/(A_ITAE_I*((t0/tau)^(B_ITAE_I)))
```

Lo cual da como resultado lo siguiente:

Kc =

Ti =

Y la simulación con estos parámetros es lo siguiente:



Lamentablemente, oscila.

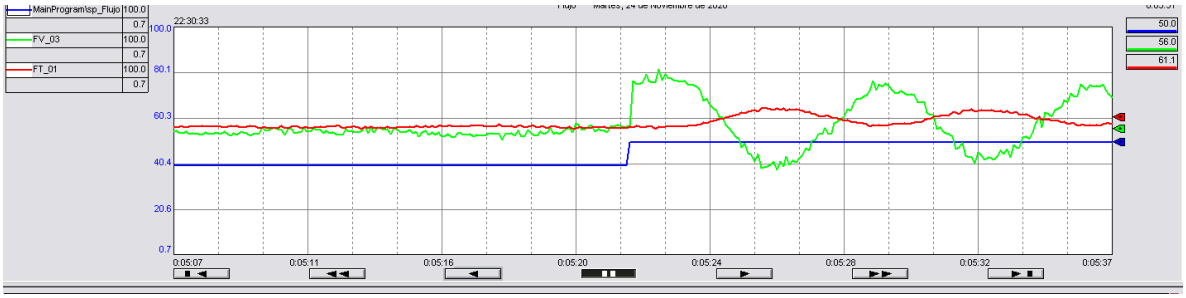
Métodos de ajuste de lazo cerrado:

Ziegler Nichols:

Para este método, se ve a que valor de ganancia proporcional el sistema en lazo cerrado oscila. Ya teniendo una oscilación permanente, se ve el tiempo del periodo de la oscilación. Con el valor de Kc oscilatorio y el periodo, se calculan los demás parámetros.

En primer paso, se deja el termino Ti lo más alto posible, y desde un valor bajo de Kc, se procede a aumentarlo hasta que la respuesta oscile.

En la imagen de abajo se ve que a cierto Kc, de valor 4.8, el sistema oscila, con un periodo de 7 segundos.



Entonces, a partir de esto, se tiene la siguiente tabla para calcular los valores de los parámetros. En nuestro caso, como queremos un controlador PI, se tiene lo siguiente:

- modo PI: $K_c = 0,45\,K_u$

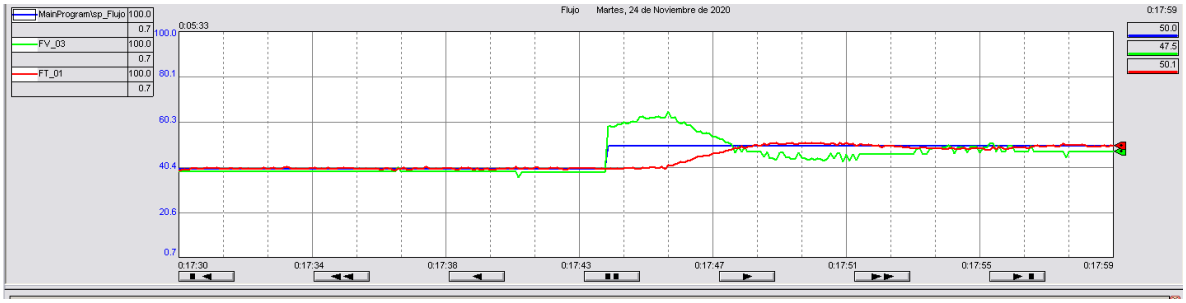
$T_i = P_u / 1.2$

Los cuales, reemplazando datos, da lo siguiente:

Kc = 2.16

Ti = 5.833

Reemplazando estos datos en el controlador, se obtiene la siguiente respuesta:



Root locus:

En lo que se refiere al método de root locus, se va a utilizar el comando sisotool de Matlab, el cual ayuda a sintonizar lazos de control de manera intuitiva.

Para poder realizar esto, es necesario tener listo las funciones de transferencias de cada bloque para que estas puedan ser utilizadas por el sisotool. Se puede ver a continuación como son las funciones de transferencia:

Primero, para el lazo de flujo, se consideró, para la planta, una ganancia positiva y unitaria, no teniendo efecto en el lazo de control.

Para la función de transferencia de la válvula, se consideró que actúa como un sistema de primer orden con retardo:

$$FT_{valvula} = \frac{K \cdot e^{-\theta s}}{\tau s + 1}$$

Con:

K = Ganancia de la válvula = 1.038

Tau (τ)= Constante de tiempo de la válvula = 5.2155

θ = Retardo de la válvula = 0.6

Como se puede ver, no se puede trabajar directamente con esta función de transferencia por la existencia de una exponencial en el denominador, el cual significa un retardo en el sistema. Para que esta sea trabajable, se va a utilizar una segunda aproximación de Pade para la función exponencial:

$$e^{-\theta s} \approx \frac{8 - 4 \cdot \theta s + \theta^2 \cdot s^2}{8 + 4 \cdot \theta s + \theta^2 \cdot s^2}$$

Con lo cual, se tiene lo siguiente:

$$FT_{valvula} = \frac{K}{\tau s + 1} \cdot \frac{8 - 4 \cdot \theta s + \theta^2 \cdot s^2}{8 + 4 \cdot \theta s + \theta^2 \cdot s^2}$$

Y, siguiendo adelante, se necesita saber cuál es la función de transferencia del controlador, el cual, en este caso, es PI, el cual es el siguiente:

$$FT_{PI} = K_c \frac{T_i \cdot s + 1}{T_i}$$

Lamentablemente, no se puede trabajar de manera simbólica en el sisotool, por lo cual hay que entregarles valores a los parámetros. Estos son:

Kc = 1.8

Ti = 4

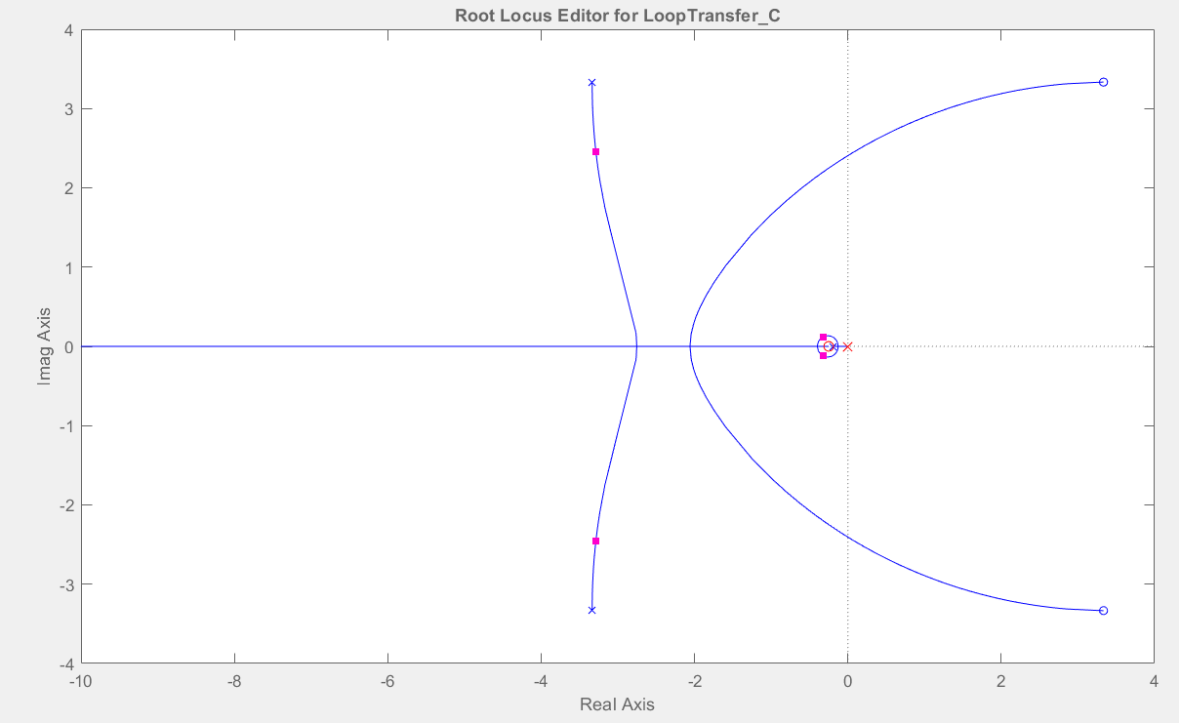
Antes de seguir, esto es lo que se escribió como código en Matlab para tener las funciones de transferencia:

```
k = 1.038;
tau = 5.2155;
retardo = 0.6;

TiF = 4;
TiN = 25;
Td = 0.0000005;
KcF = 1.8;
KcN = 8;
syms s

%Lazo flujo

Flujo_valvula_y_planta_numerador = [(retardo^2)*k -4*retardo*k 8*k];
Flujo_valvula_y_planta_denominador = [(retardo^2)*tau (4*retardo*tau +(retardo^2)) (8*tau)+(4*retardo) 8];

Flujo_controlador_numerador = [TiF*KcF KcF];
Flujo_controlador_denominador = [TiF 0];

Flujo_FT_Valvula_Planta = tf(Flujo_valvula_y_planta_numerador, Flujo_valvula_y_planta_denominador)
Flujo_FT_Controlador = tf(Flujo_controlador_numerador, Flujo_controlador_denominador)
```

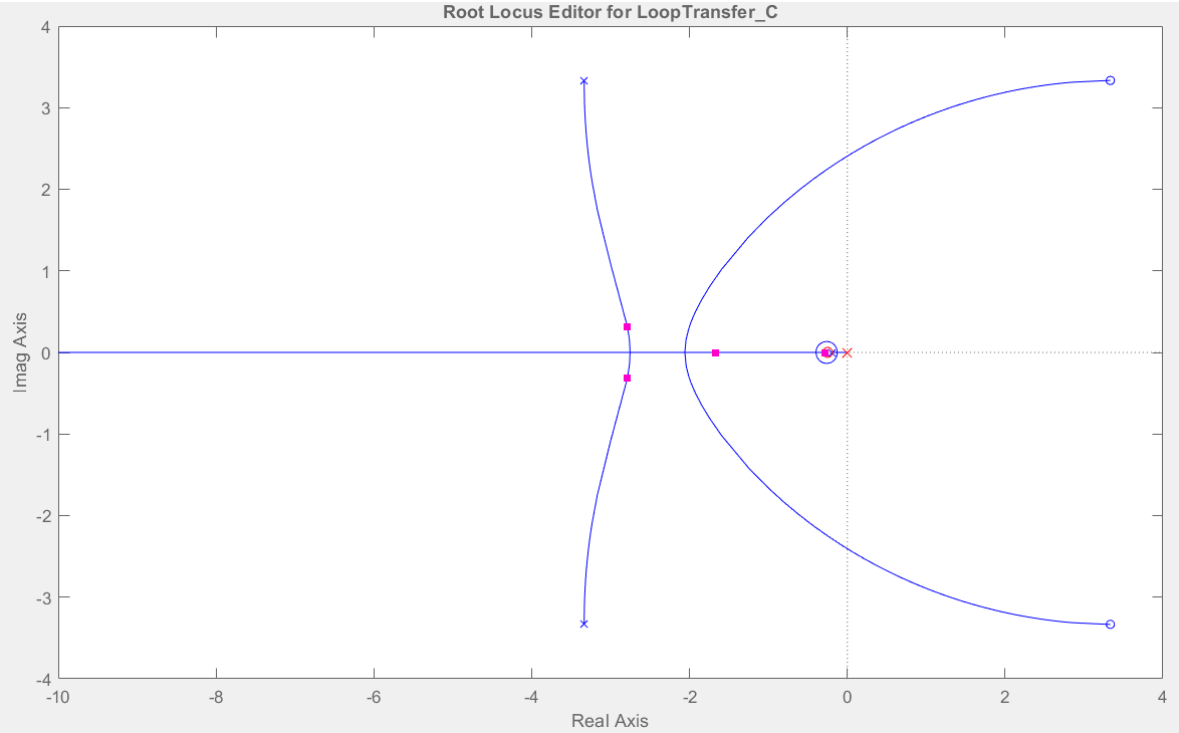
Y se obtuvo lo siguiente, como función de transferencia:

```
Flujo_FT_Valvula_Planta =

     0.3737 s^2 - 2.491 s + 8.304
  -----------------------------------
  1.878 s^3 + 12.88 s^2 + 44.12 s + 8

Continuous-time transfer function.


Flujo_FT_Controlador =

  7.2 s + 1.8
  -----------
      4 s
```

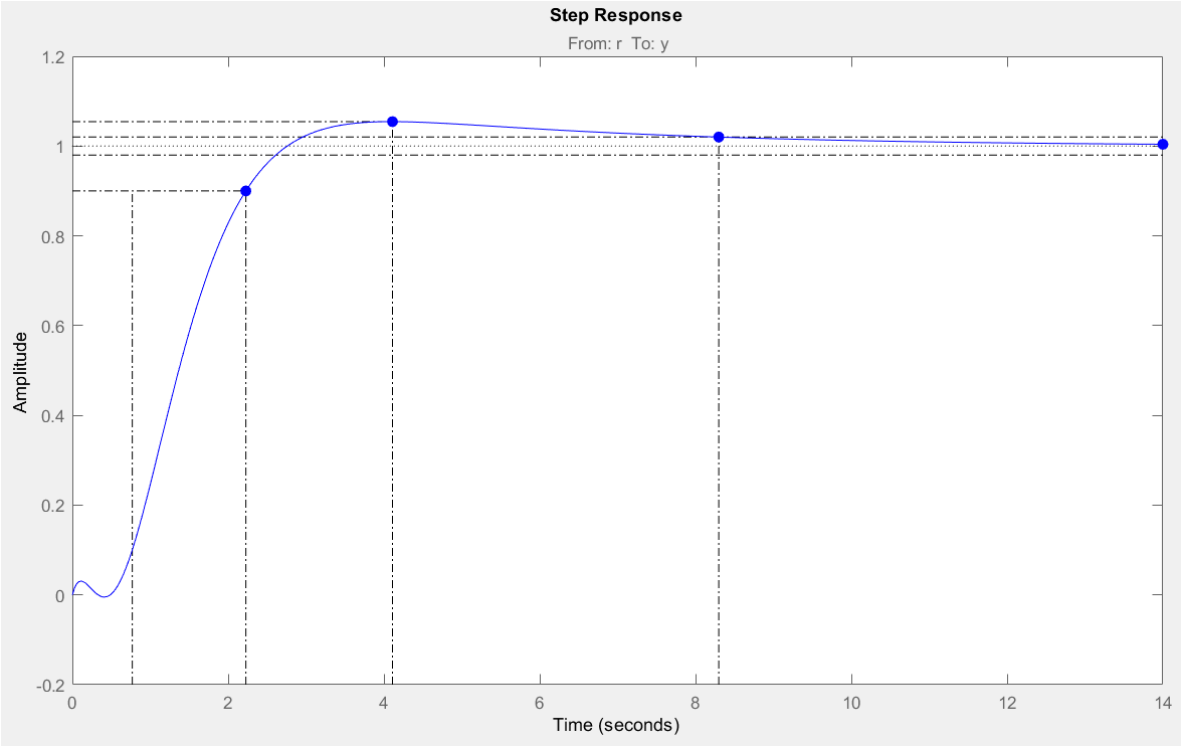
Entonces, se inicia el comando sisotool, se modifica los bloques correspondientes para que tengas las funciones de transferencia respectivas, y aquí se encuentra el root locus del lazo de flujo:



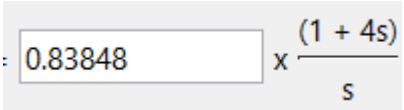
Ahora, por ejemplo, no se quiere que el sistema tenga un gran sobrepaso, así que se va a realizar la siguiente modificación:



Y la repuesta a escalón, según sisotool, es la siguiente:



Numéricamente, se tiene que se modificó el valor de Kc de esta manera:

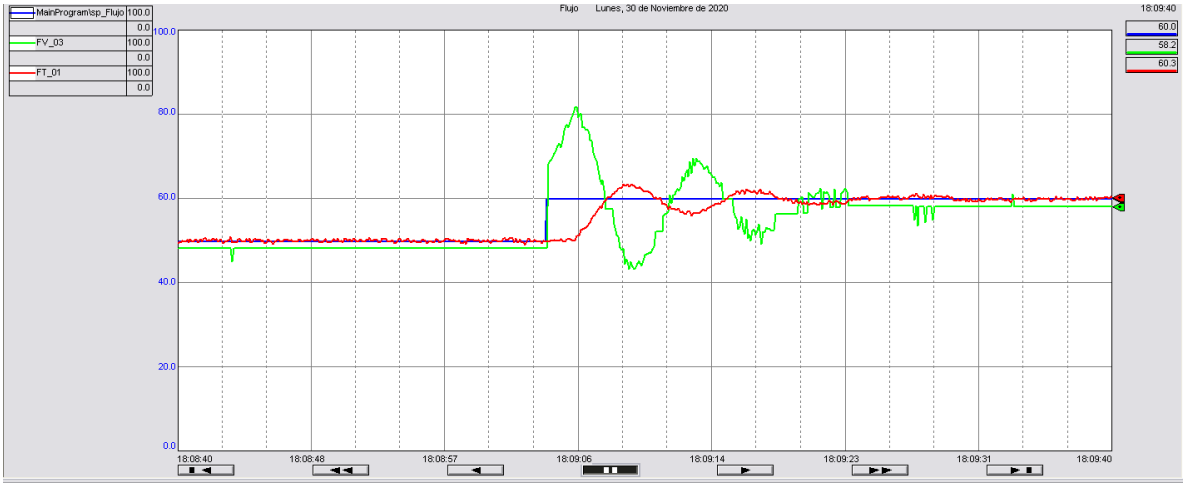


$$\frac{K_c}{T_i} = 0.83848, y\ como\ T_i = 4$$

$$K_c = T_i \cdot 0.83848 = 4 \cdot 0.83848$$

$$K_c = 3.35392$$

Entonces, con este valor de Kc, se procede a simularlo en el RSLogix:



No es parecido a la respuesta que entrego sisotool a respuesta escalón, pero es conforme a lo que queríamos.

Prueba y error (Ultimate):

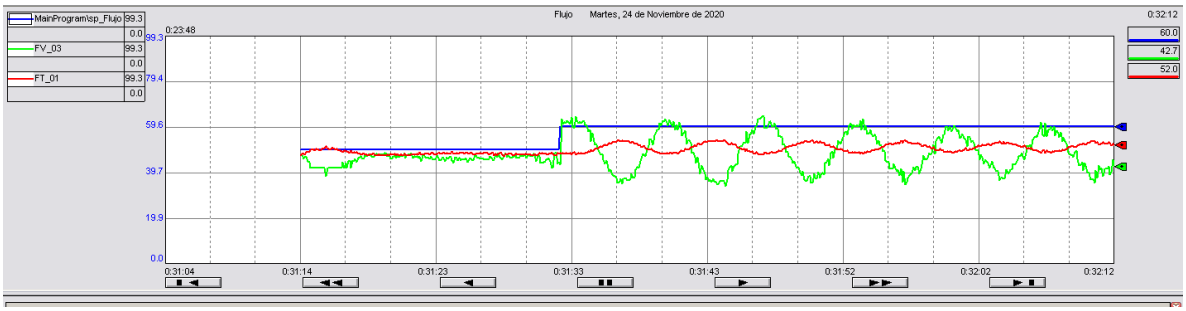
En lo referente a este método, se ve, a través de un análisis gráfico, los valores correctos para una sintonización adecuada del controlador.

Los pasos son los siguientes:

1) Primero, desconectar cualquier acción derivativa e integral del controlador.
2) Poner una baja ganancia.
3) Aumentar la ganancia y realizar una prueba escalón.
4) Repetir el punto 3) hasta que la respuesta sea oscilatoria.
5) De ahí, disminuir la ganancia a la mitad.
6) Poner la acción integral a un valor muy alto y observar la respuesta escalón.
7) Reducir Ti hasta un valor donde la respuesta de oscilatoria.
8) Aumentar el valor de Ti al doble de lo que se obtuvo en 7).
9) Poner acción derivativa Td hasta que aparezca ruido en la respuesta escalón.
10) Reducir a la mitad el valor Td obtenido en 9).
11) Variar la ganancia en torno a los valores calculados y examinar la respuesta hasta que entregue una respuesta satisfactoria.

Entonces, siguiendo estos pasos, exceptuando la parte derivativa, ya que queremos sintonizar un controlador PI, se obtuvo lo siguiente:

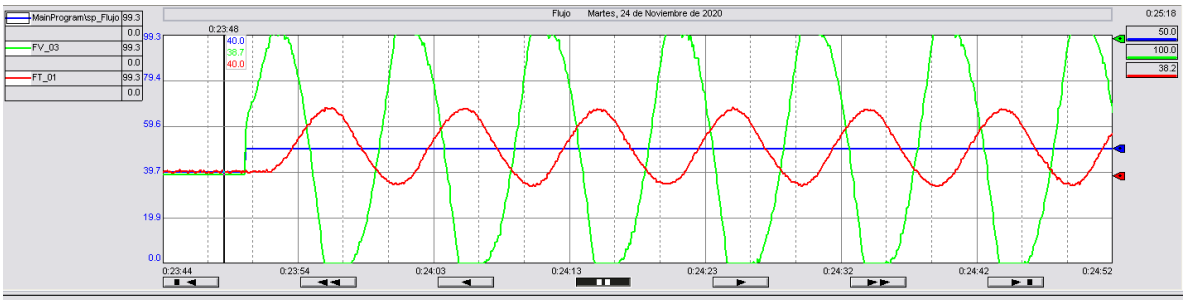
Kc oscilatorio:



El Kc oscilatorio que se obtuvo fue de 4.8.

Ahora, reduciendo el Kc a la mitad, 2.4, se realiza la respuesta escalón para la acción integral, hasta que este oscile. Lo que se obtuvo se muestre en el gráfico de abajo

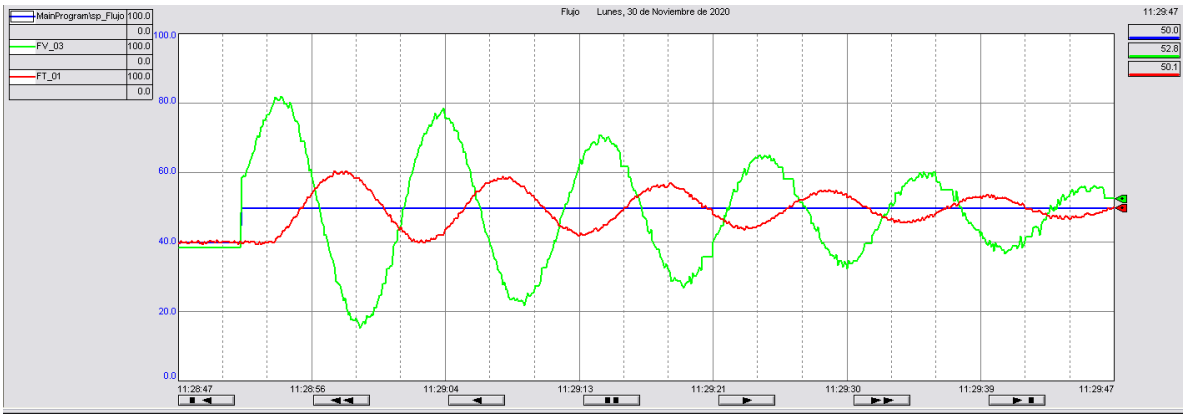
Ti oscilatorio:



El valor que se obtuvo fue de Ti = 1.

Entonces, ahora vemos la respuesta que tiene el controlador con el Kc = 2.4, y el Ti al doble de lo que se obtuvo, 2.0, antes de afinar los valores de estos.

Respuesta sin afinar los parámetros:

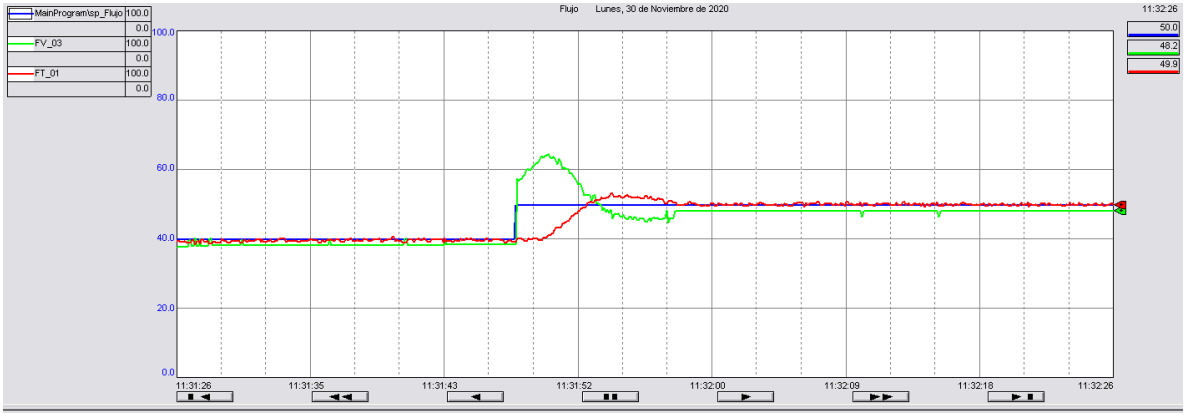


Como se puede ver, todavía no tiene una respuesta satisfactoria. Aquí se procede a justar los parámetros, los cuales, después de ajustarlos, quedan con los siguientes valores:

Kc = 1.8

Ti = 4

Y su respuesta a escalón es la siguiente:



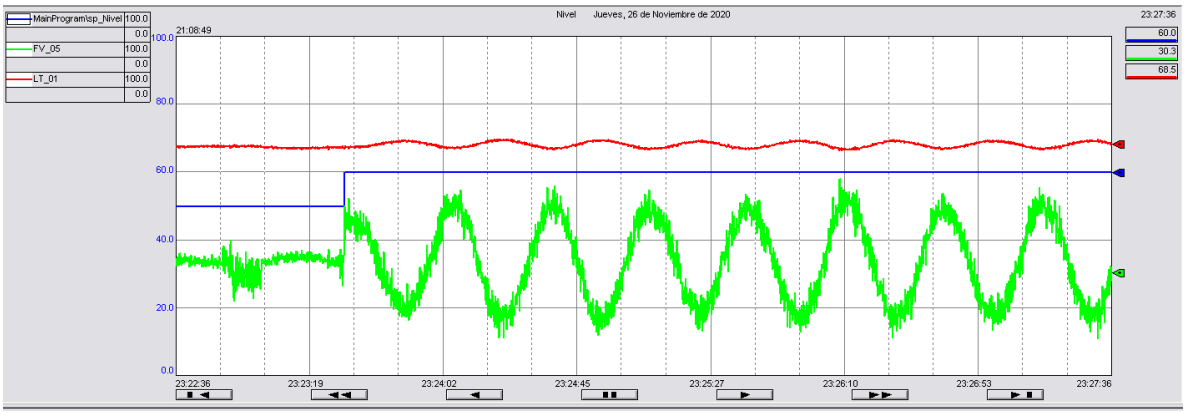
Lazo de nivel:

Lazo con estructura simple:

Zieger Nichols:

De la misma manera que se realizó anteriormente para el lazo de flujo, pero esta vez, con un controlador PID, se procede a encontrar un Kc oscilatorio, con acción derivativa nula y acción integrativa demasiado alta para que pueda afectar en la respuesta.

El Kc oscilatorio que se encontró tiene un valor de 14.3, con un periodo de 26 segundos. En la grafica de abajo se puede ver la respuesta a escalón de manera gráfica:



Entonces, con los valores de Kc oscilatorio y periodo conocidos, se obtiene los valores de los parámetros del controlador PID de nivel a través de la siguiente tabla:

- modo PID:	K_c	=	$0,6\,K_u$
	T_i	=	$0,5\,P_u$
	T_d	=	$P_u/8$

Los cuales dan como parámetros los siguientes:

Kc = 8.58

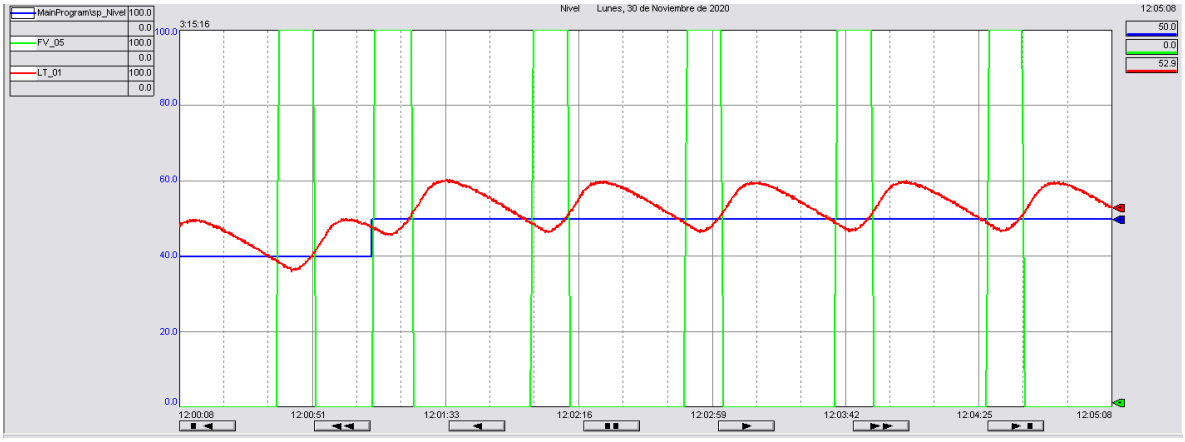
Ti = 13

Td = 3.25

Entonces, con estos valores en consideración, se ve la repuesta escalón con estos parámetros:

Respuesta con derivada en el PV:

Respuesta con derivada en el error:



En cualquiera de los dos casos, la sintonización no fue la correcta. Cuando la acción derivativa estaba basada en la variable de proceso, este subió indefinidamente, y cuando este estaba basado en el error, esta subia y bajaba de manera casi inmediata, haciendo oscilar la posición del nivel de estanque indefinidamente.

Root Locus:

De la misma manera que se realizó en el lazo de flujo, se realiza el mismo procedimiento, pero con la diferencia de que ahora se considera una planta con un integrador y un controlador PID.

Para la función de transferencia del tanque, se tiene que se modela de la siguiente manera:

$$FT_{tanque} = \frac{1}{100 \cdot s}$$

Y la válvula, de la misma manera que en lazo de flujo, se modela de la siguiente manera, ya realizando la aproximación de segundo orden de Pade para el retardo de esta:

$$FT_{valvula} = \frac{K}{\tau s + 1} \cdot \frac{8 - 4 \cdot \theta s + \theta^2 \cdot s^2}{8 + 4 \cdot \theta s + \theta^2 \cdot s^2}$$

Con:

K = Ganancia de la válvula = 1.038

Tau (τ)= Constante de tiempo de la válvula = 5.2155

θ = Retardo de la válvula = 0.6

De esta manera, la función de transferencia del controlador PID es el siguiente:

$$FT_{PID} = K_c \cdot \frac{T_d \cdot T_i \cdot s^2 + T_i \cdot s + 1}{T_i \cdot s}$$

Ya que la herramienta de Matlab sisotool no trabaja con valores simbólicos, se le tiene que entregar valores numéricos a los parámetros:

Kc = 8

Ti = 25

Td = 0.0000005

En este caso, la acción derivativa va a estar fijada en la variable de proceso, para evitar que acción derivativa se sobreponga sobre la acción proporcional e integrativa si hubiera un cambio muy brusco en el error.

El código en Matlab que se utilizó para estas funciones de transferencias son las siguientes:

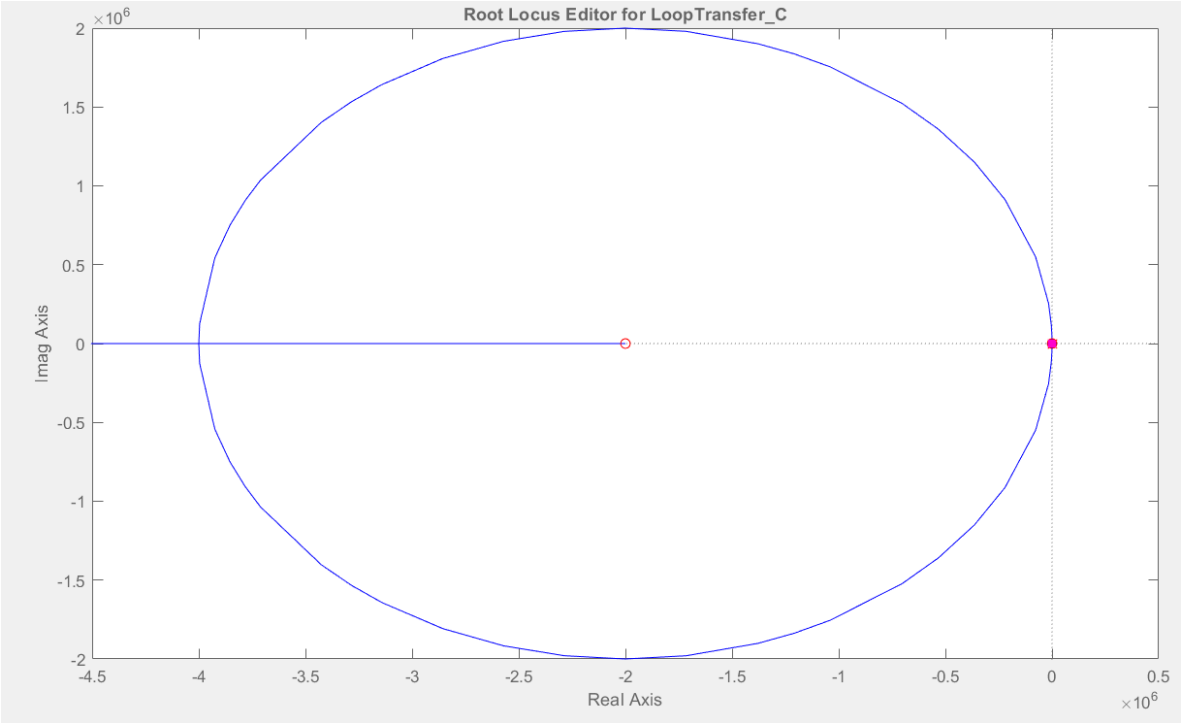
```
k = 1.038;
tau = 5.2155;
retardo = 0.6;

TiF = 4;
TiN = 25;
Td = 0.0000005;
KcF = 1.8;
KcN = 8;
syms s
```

```
%Lazo Nivel

Nivel_valvula_y_planta_numerador = [(retardo^2)*k -4*retardo*k 8*k];
Nivel_valvula_y_planta_denominador = [100*(retardo^2)*tau (400*retardo*tau +100*(retardo^2)) 100*(8*tau)+100*(4*retardo) 8*100 0];

Nivel_controlador_numerador = [Td*TiN*KcN TiN*KcN KcN];
Nivel_controlador_denominador = [TiN 0];

Nivel_FT_Valvula_Planta = tf(Nivel_valvula_y_planta_numerador, Nivel_valvula_y_planta_denominador)
Nivel_FT_Controlador = tf(Nivel_controlador_numerador, Nivel_controlador_denominador)
```

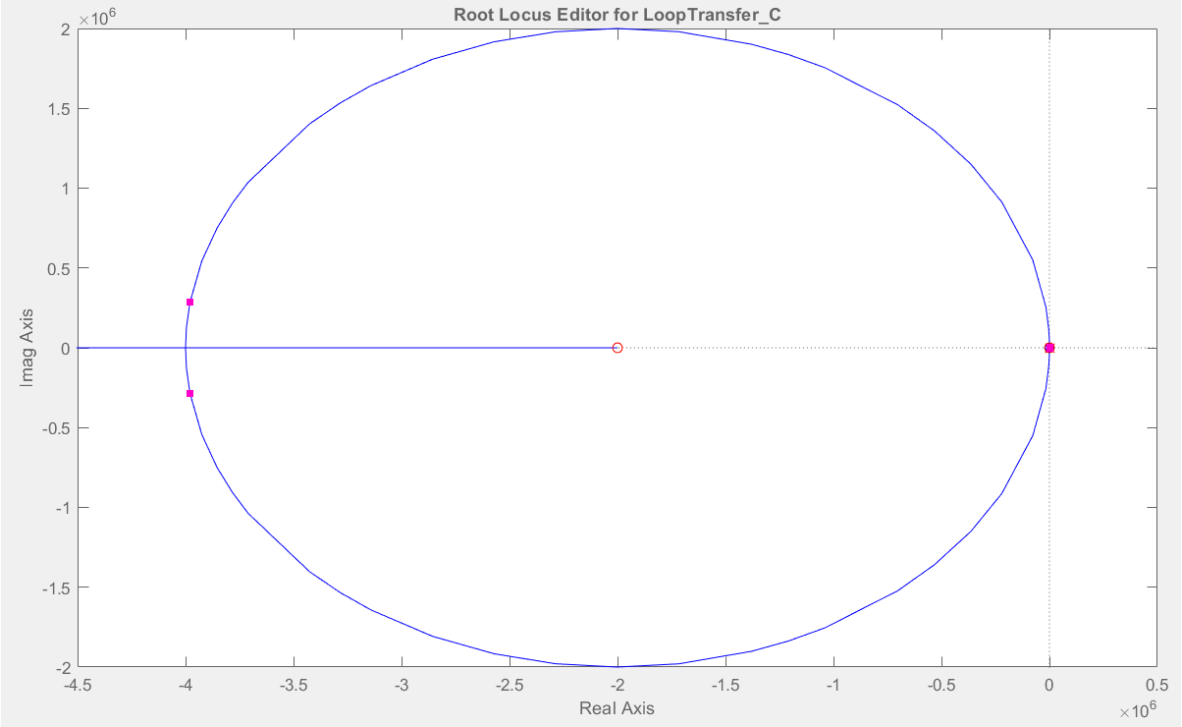
Y lo resultante del código son las siguientes funciones de transferencia:

```
Nivel_FT_Valvula_Planta =

        0.3737 s^2 - 2.491 s + 8.304
  ---------------------------------------
  187.8 s^4 + 1288 s^3 + 4412 s^2 + 800 s

Continuous-time transfer function.


Nivel_FT_Controlador =

  0.0001 s^2 + 200 s + 8
  ----------------------
           25 s

Continuous-time transfer function.
```

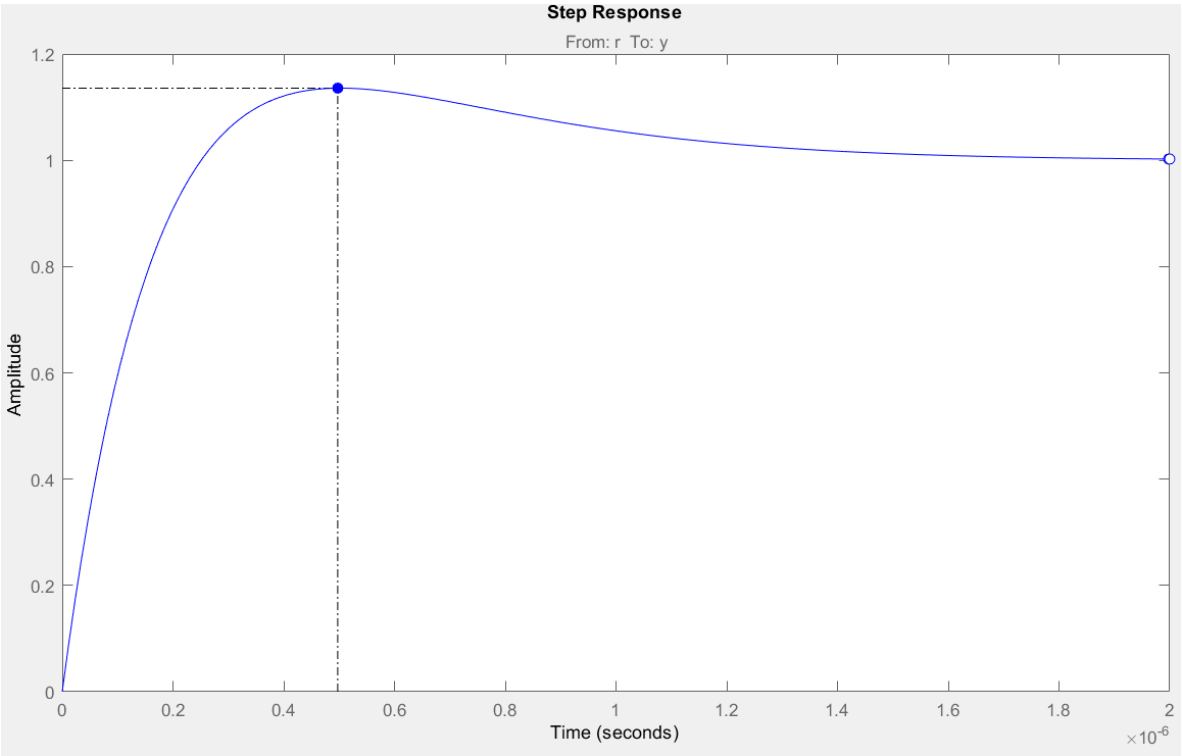
Entonces, aplicando sisotool, con los parámetros elegidos, se tiene el siguiente diagrama de root locus:



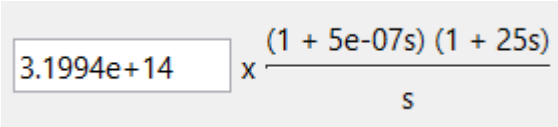
Entonces, modificando el root locus para que tenga menor sobrepaso, se modifica la función de transferencia del controlador:



Y la repuesta escalón que tiene el sistema ahora es el siguiente:



El valor que tiene el parámetro Kc es el siguiente:

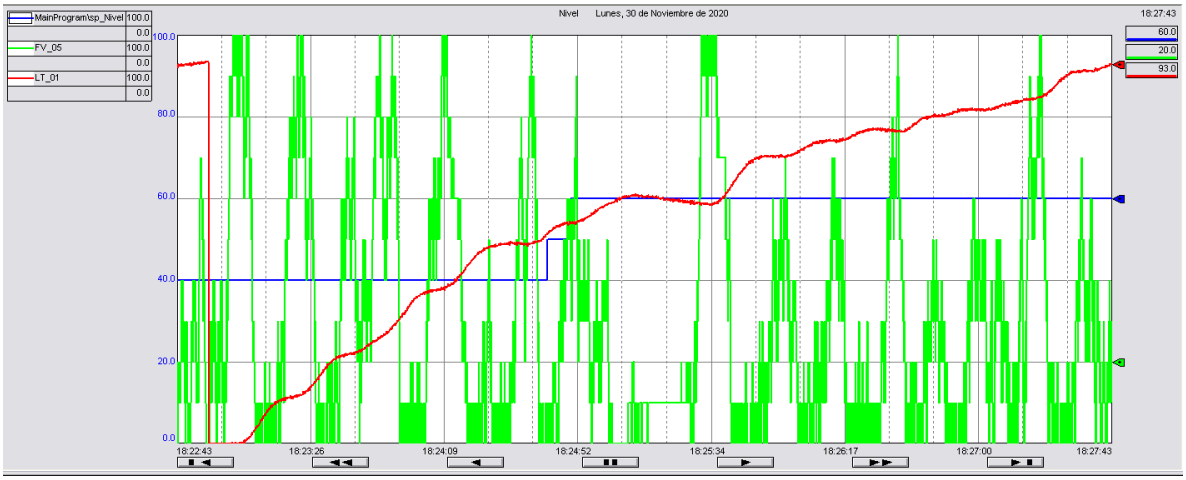


$$\frac{K_c}{T_i} = 3.1993 \cdot 10^{14}, y\ como\ T_i = 12$$

$$K_c = T_i \cdot 3.1993 \cdot 10^{14} = 12 \cdot 3.1993 \cdot 10^{14}$$

$$K_c = 3.8392 \cdot 10^{15}$$

Entonces, con este valor de Kc, se procede a simularlo en el RSLogix:

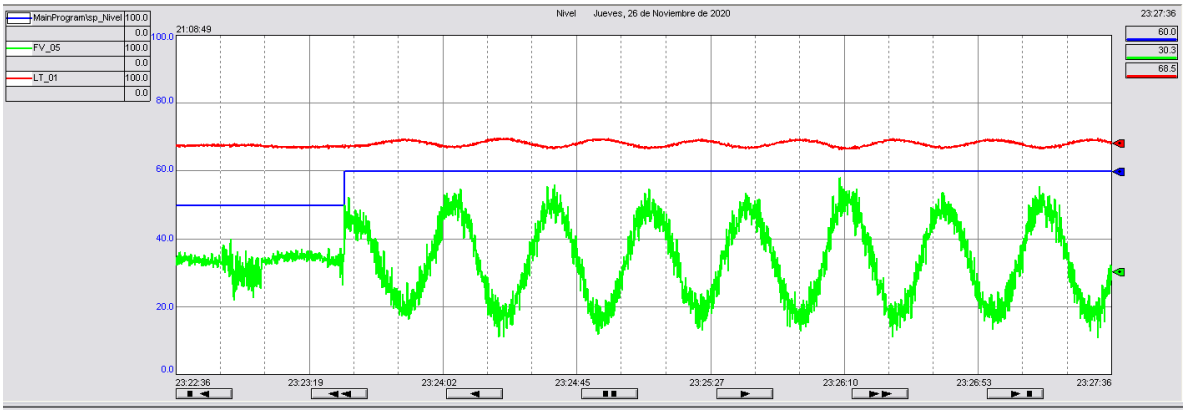


Lamentablemente, en este caso, el root or Matlab no fue correcto en su utilización, siendo que lo más probable que la aproximación de pade no fue una buena aproximación, o, en Matlab se trabaja muy idealizado el control, mientras que eso no sucede trabajando con RSLogix.

Prueba y error (Ultimate):

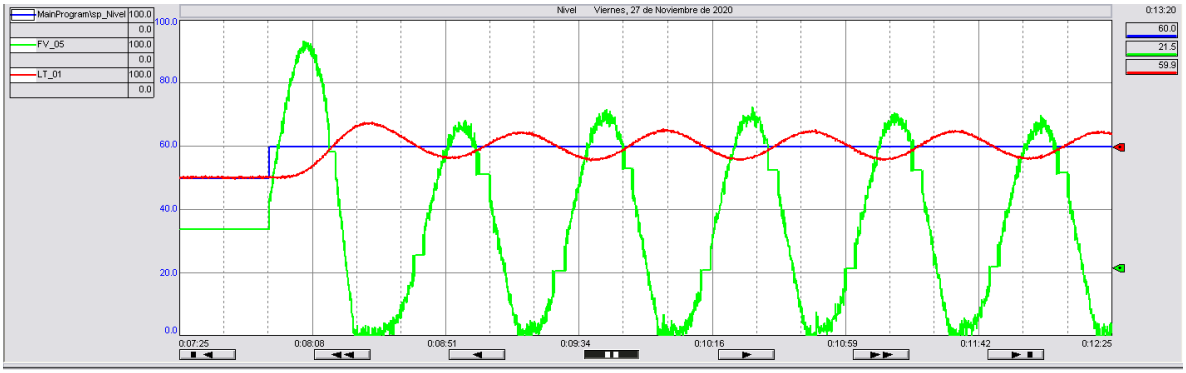
De la misma manera que se realizó en el lazo de flujo, se procede a ver a que valor de Kc oscila el lazo de nivel, teniendo el valor de Td a 0, y el valor de Ti a lo más alto posible.

Kc oscilatorio:



El valor Kc que se obtuvo fue de 14.3. Entonces, ahora se quiere encontrar el valor de Ti que haga oscilar el sistema a una respuesta escalón. Esto se realiza con dividendo a la mitad el valor de Kc, a un valor de 7.15.

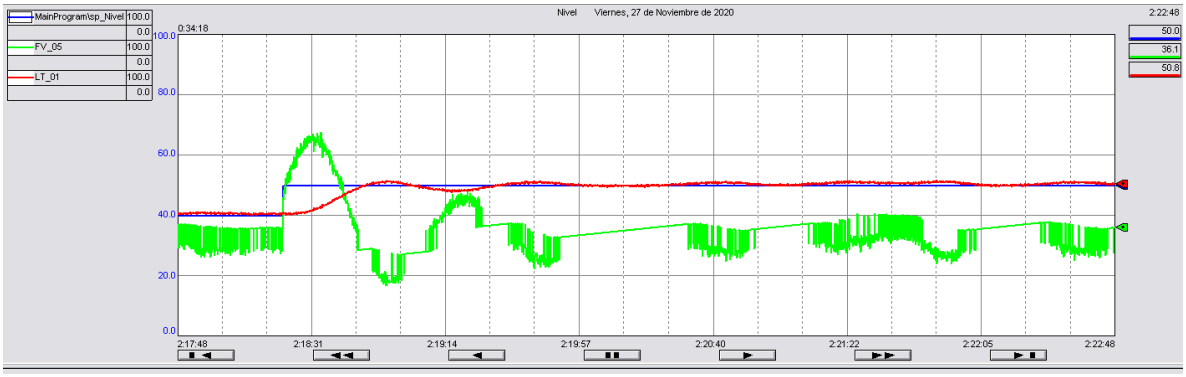
Ti oscilatorio:



El valor obtenido, el cual oscila, es el valor de Ti = 12.

Ahora, para ver el valor de Td el cual el sistema tenga ruido, se tiene que realizar una respuesta escalón al sistema con los valores de Kc = 7.15, y un valor de Ti al doble del Ti oscilatorio, Ti = 24.

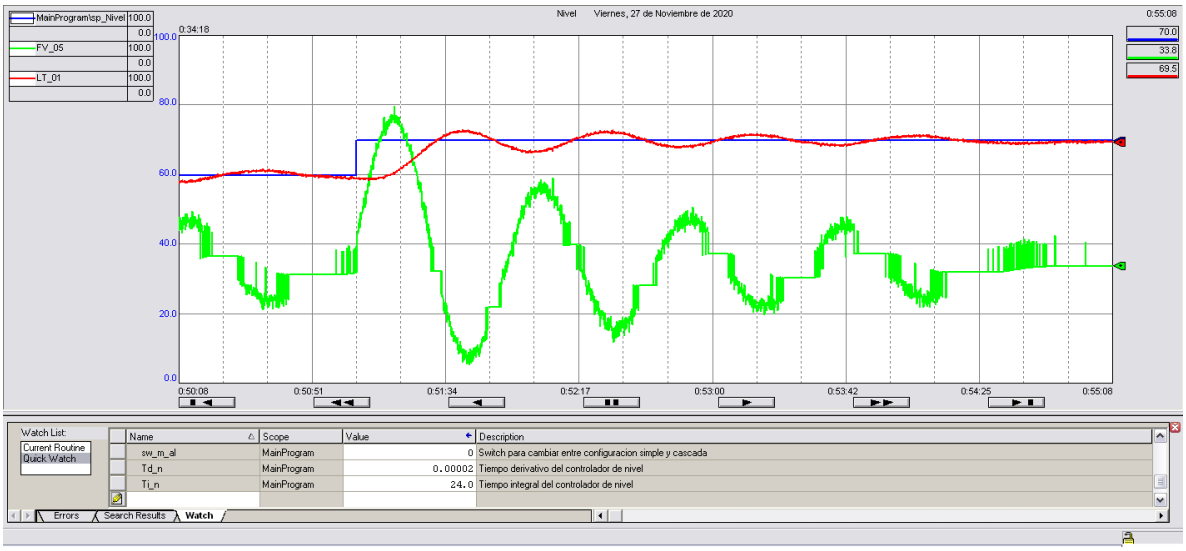
Td, derivada en PV



Aquí, el valor que se obtuvo, cuando el sistema presenta ruido, y con la acción derivativa basa en el PV, es de un valor de 0.000001.

Ahora, realizando el mismo proceso de nuevo, pero con la acción derivativa en el error.

Td, derivada en error:



Aquí, el valor que se obtuvo fue de Td = 0.00002.

Ahora, sin ajustar los valores, se procede a obtener la repuesta al sistema con los valores anteriormente obtenidos, y con los valores de Td amplificados al doble.

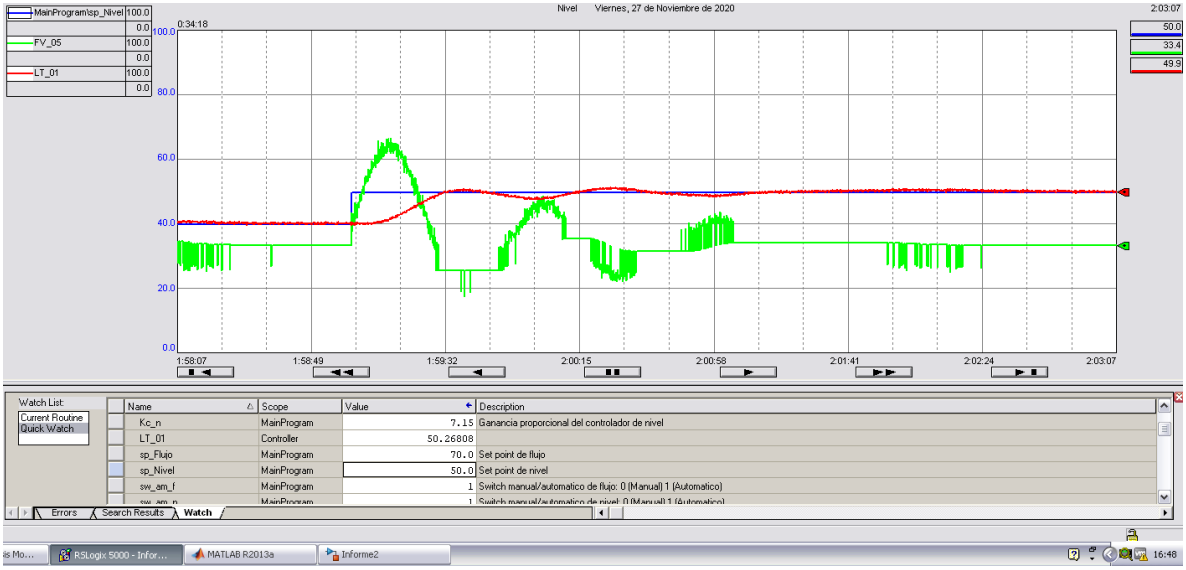
Para la respuesta del sistema con acción derivativa basada en PV, se utilizo los siguientes valores:

Kc = 7.15

Ti = 24

Td = 0.0000005

Y abajo se muestra su respuesta:



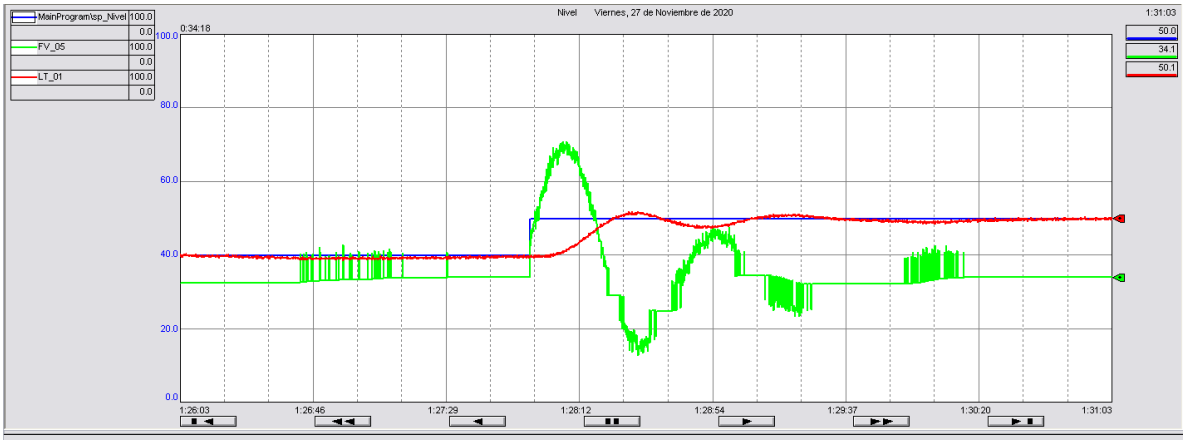
De la misma manera, se procede a ver la respuesta al sistema, pero esta vez con la acción derivativa basada en el error. Por esto, se utilizó los siguientes valores:

Kc = 7.15

Ti = 24

Td = 0.00001

Y se muestra su respuesta:



Se puede ver que, para los dos casos, se puede ver una respuesta conforme a los esperado y es acorde a la especificación que se quiere como respuesta: que no oscile, y que tenga un poco de sobrepaso.

Ahora, afinando, los valores, se tiene lo siguiente:

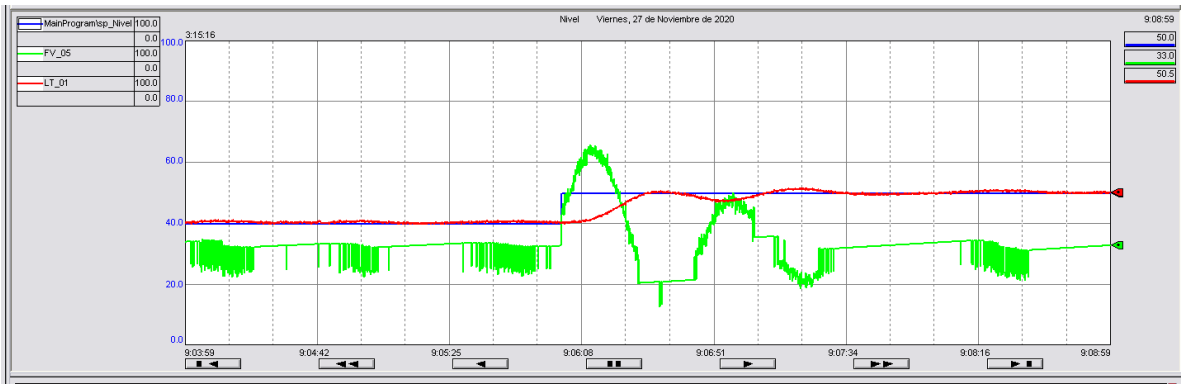
Cuando el controlador está basado en acción derivativa en PV:

Kc = 8

Ti = 25

Td = 0.0000005

Respuesta con valores ajustados, derivada en PV:



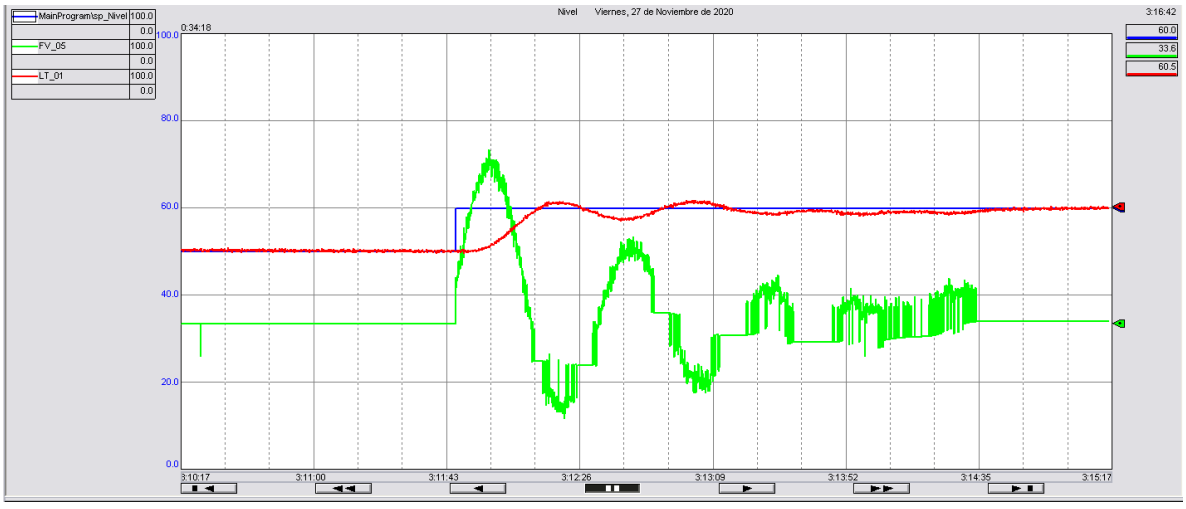
Cuando el controlador está basado en acción derivativa en el error:

Kc = 8

Ti = 25

Td = 0.00001

Respuesta con valores ajustados, derivada en error:



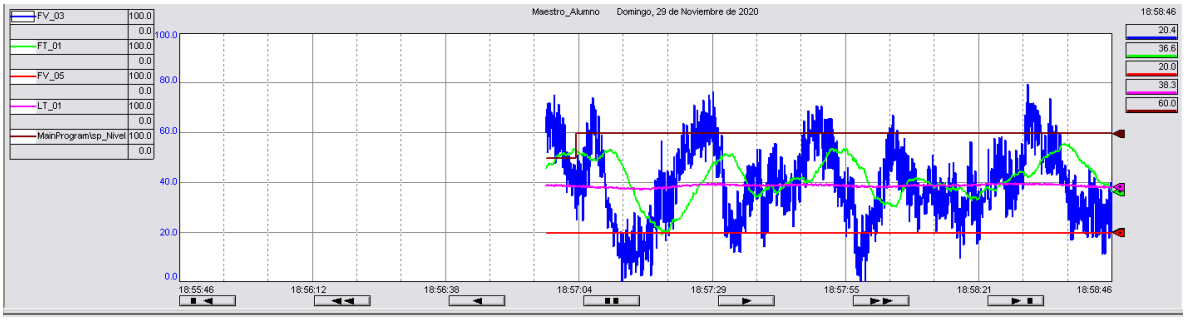
Lazo cascada:

De la misma manera que se realizó anteriormente, pero esta vez cuando la configuración de control está en modo cascada, se va a realizar dos métodos de sintonización de lazo cerrado: Zieger Nichols, y Prueba y error.

Zieger Nichols:

Siguiendo los requerimientos de este método, se dejó el valor de Td en 0 y el valor de Ti en el mayor posible. Con eso, se buscó un valor Kc que haga tener al nuevo sistema una respuesta oscilatoria.

Lamentablemente, no se pudo tener una respuesta satisfactoria para poder encontrar un valor Kc oscilatorio. Se intento buscar lo más aproximado, lo cual se muestra en la imagen siguiente:



De lo que se pudo obtener, el Kc que se obtuvo de 11, y el periodo fue de 22 segundos.

Con estos datos, se tuvo que los valores de los parámetros del controlador son los siguientes:

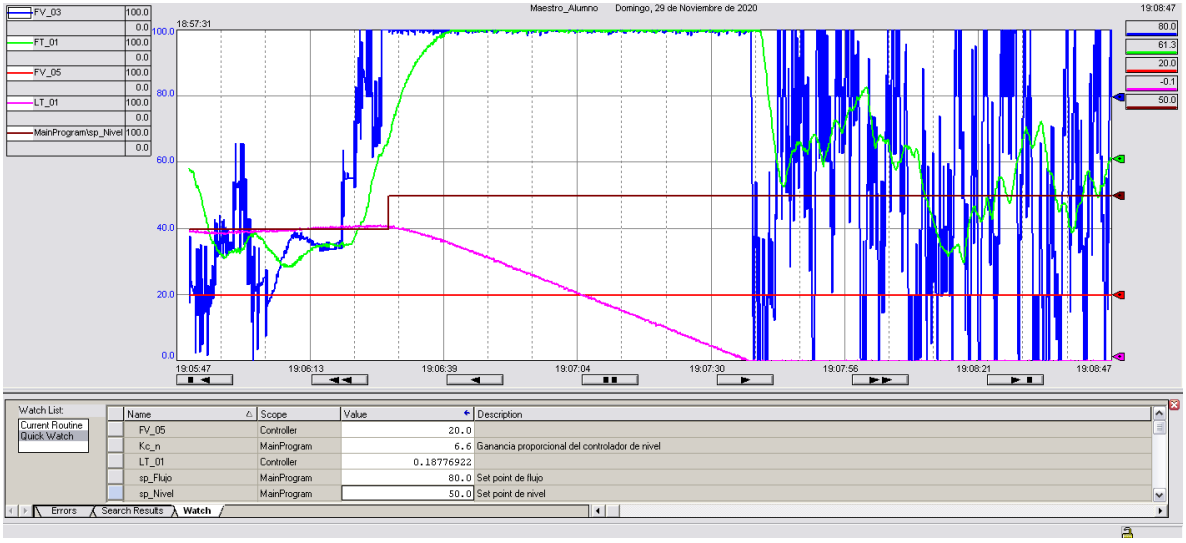
Modo PID	$0.6 \cdot K_c$	$0.5 P_{cu}$	$P_u/8$

Kc = 6.6

Ti = 11

Td = 2.75

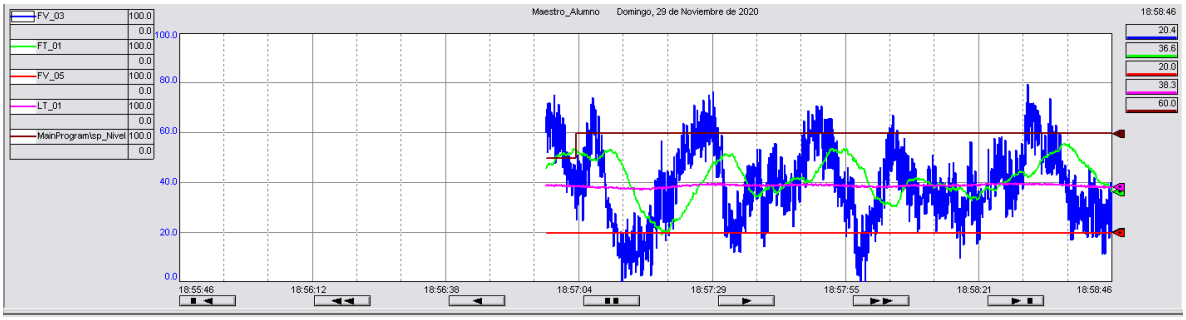
Con estos parámetros, se realizó una prueba escalón al sistema, y esto es lo que se obtuvo:



Lamentablemente, el nivel del estanque baja hasta el nivel 0, mientras que el flujo de FV_03 aumenta hasta el límite máximo durante el tiempo, para después variar erráticamente, centrado en el valor del set point de nivel que se entregó, pero, aun así, no sube el nivel del estanque. Se va a ver más adelante que es por el parámetro de Td que se encuentra muy alto.

Prueba y error (Ultimate):

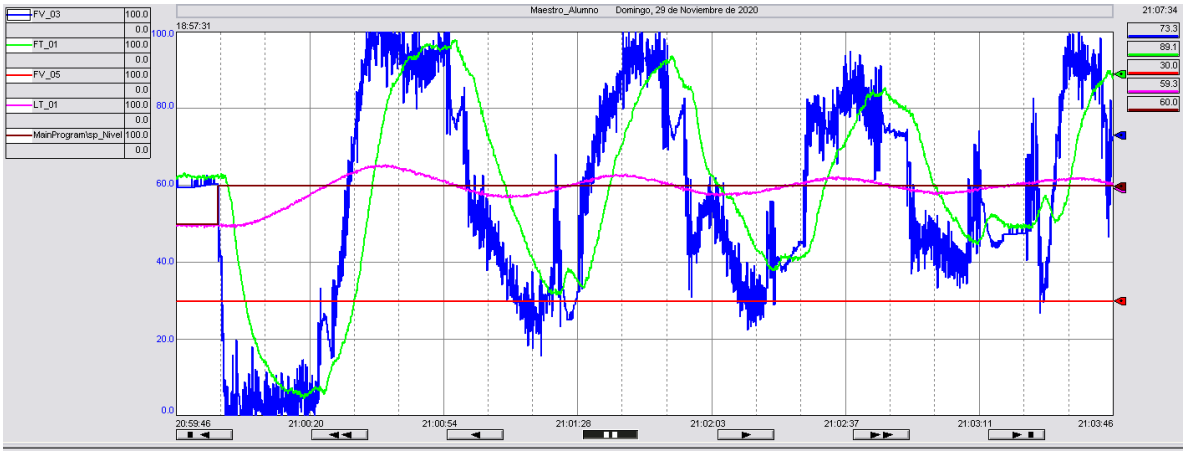
Repitiendo el método de prueba y error, se procede a encontrar el Kc oscilatorio.



Reutilizando lo que se calculó en el método de Zieger Nichols, el Kc oscilante que se encontró es de un valor de 11.

Con esto en consideración, para poder encontrar el valor de Ti que haga oscilar el sistema, se aplica un escalón al sistema a cierto valor de Ti, mientras que la ganancia Kc que se utiliza es la mitad del Kc oscilatorio, un valor de 5.5.

De esto, se obtuvo lo siguiente:

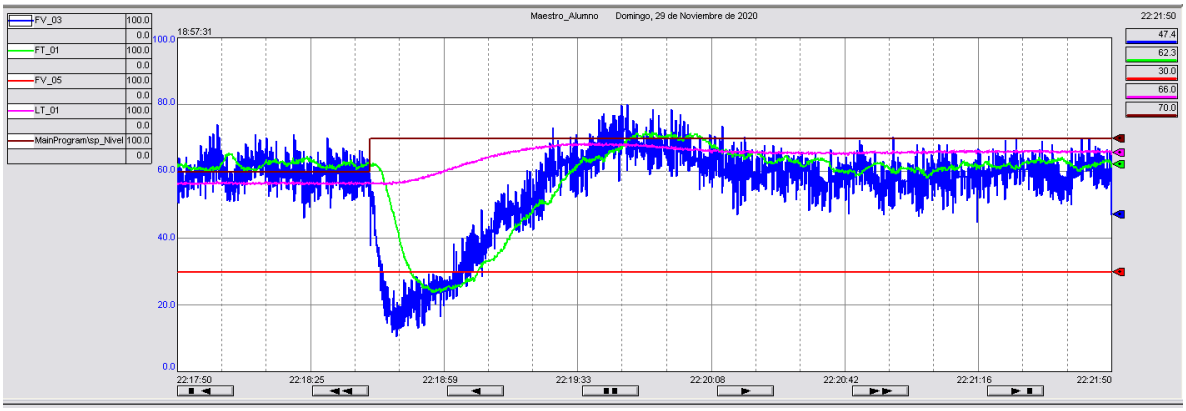


El valor de ti que se obtuvo, que hace oscilar el sistema, es de 10.

Ahora, para poder calcular el Td, se elige que la acción derivativa se base en la variable de proceso. Esto es porque, a una respuesta escalón, la acción derivativa del control no se impone a la acción integral y proporcional del sistema de manera significativa, lo cual es lo que se busca aquí, ya que el lazo de nivel es lento en comparación al lazo de flujo.

Por esto, para poder encontrar un Td que haga que el sistema presente ruido, se aplica el mismo Kc de antes, 5.5, y el valor de Ti se aumenta al doble, 20.

Por prueba, se obtuvo lo siguiente:



Se obtuvo un valor de 0.00001 en Td, que es cuando se presenta ruido en el sistema.

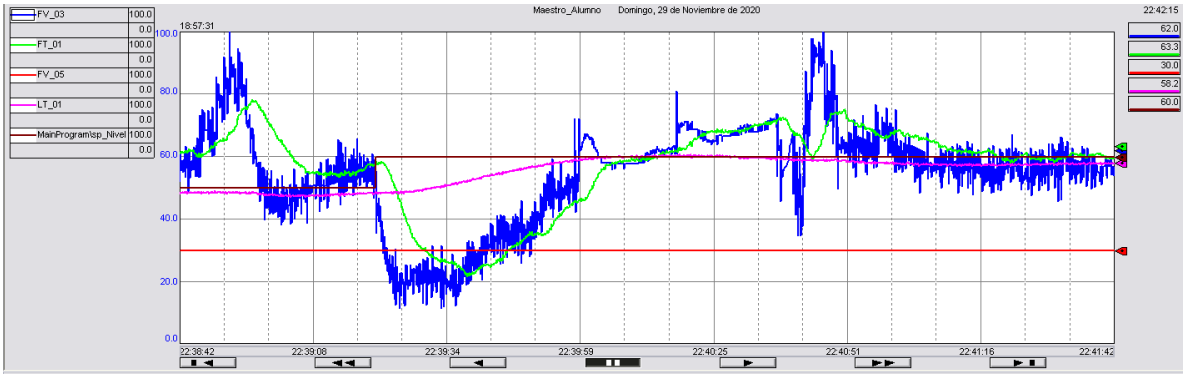
Tomando esto en cuenta, los parámetros no ajustados del sistema son los siguientes:

Kc = 5.5

Ti = 20

Td = 0.000005

Y la repuesta del sistema, con estos parámetros, es el siguiente:



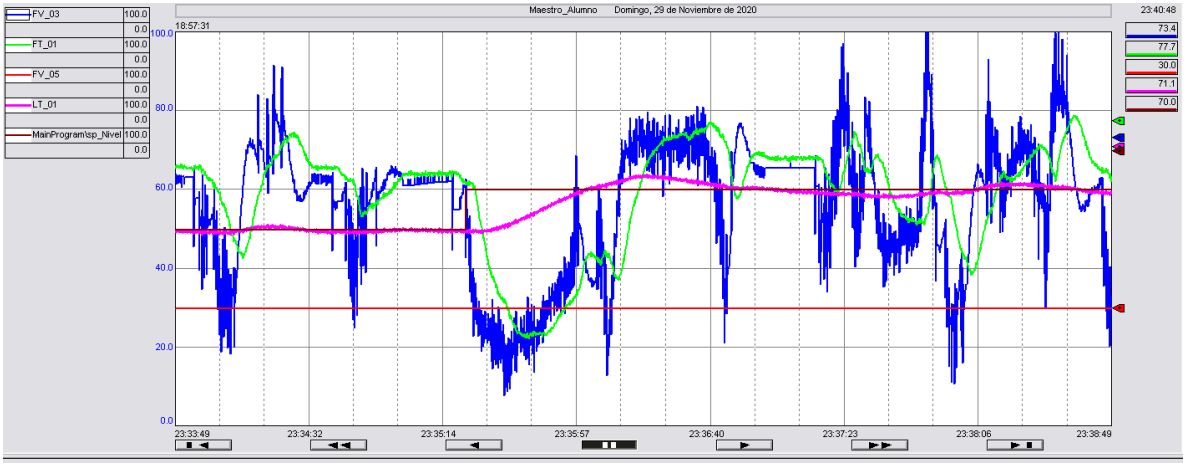
Y, a partir de esto, se empieza a justar los parámetros para encontrar una respuesta más acorde a lo que queremos. De esto, se encontró los siguientes valores finales para los parámetros del controlador:

Kc = 5

Ti = 20

Td = 0.00000007

Y aquí está la respuesta del sistema a estos valores:



d) Contrastar de resultados entre métodos teóricos (lazo abierto) y prácticos (lazo cerrado).(1 pto)

Lazo de flujo:

Estos son los valores que se encontraron por métodos de lazo abierto:

Método	Kc	Ti
Zieger Nichols	7.5368	1.998
CC	7.6158	1.6111
3C	6.9148	1.3719
IAE	7.9945	1.8595
ISE	10.0012	2.1443
ITAE	6.8445	1.7784

Métodos de lazo cerrado:

Método	Kc	Ti
Zieger Nichols	2.16	5.833
Root locus	3.35392	4
Ultimate	1.8	4

La diferencia entre los dos métodos es que, al final, los métodos de lazo cerrado son muchos mas confiables que los de lazo abierto, ya que se puede dar un mejor ajuste de los parámetros, además de que, con el método ultimate, uno puede realizar la sintonización sin importar el tipo de sistema que haya, y se puede asegurar que este va a ser un proceso correcto de control.

Ahora, para el lazo de nivel:

Lazo simple de control:

Método	Kc	Ti	Td
Zieger Nichols	8.58	13	3.25
Root locus	$3.8392 \cdot 10^{15}$	12	0.0000005
Ultimate	8	25	0.00001

Lazo en cascada:

Método	Kc	Ti	Td
Zieger Nichols	6.6	11	2.75
Root locus	5	20	0.00000007

En referencia a la diferencia entre cada método, en general el método ultimate es el más confiable, ya que uno mismo afina los parámetros según como responde la máquina.

e) Conclusiones y Resultados.(1 pto)

Como conclusión, la sintonización de los parámetros de control, sin importar el lazo, toma mucho tiempo en poder llegar a un valor correcto de sintonización, aun más, como en el caso del lazo nivel, donde se encuentra un integrados incluido en el lazo.

Aun así, con los métodos de lazo abierto, que son metódicos, y los de lazo cerrados, como el Zieger Nichols y Root locus, uno puede estimar un valor acorde según desde esos valores, como una guía para una sintonización más correcta.

El hecho de que, en esta planta, no haya funcionado ningún tipo de sintonización de lazo abierto es por el hecho de que la constante de tiempo es muy grande en comparación con el el retardo, e igual, porque esos métodos se usan en tiempo continuo, se tendría que tener un tiempo de muestreo mucho menor, para que sea lo más parecido posible a un proceso continuo.

En comparación con el lazo simple y el lazo en cascada, el lazo simple trabaja independiente de como este trabajando los otros lazos, en este caso, el lazo de flujo con el de nivel, mientras que en el de cascada, estos están interconectados.

Al tener el lazo de flujo dentro del lazo de nivel, un lazo mucho más rápido que el lazo de nivel, este puede adecuadarse mucho más rápido a cualquier variación que se pudiera tener en el proceso, ya que el lazo de nivel, al ser más rápido, corrige el error de manera más rápida, para que el nivel no se mueva tanto y no varié tanto.