Embedded Control Laboratory

Ball on Inclined Plane

Report of Lab Exercise 2

WS 2019/2020

Group 04

1536988 Venkata Sai Tarak Padarthi

1536199 Kaushik Goud Chandapet

1533846 Sai Radha Krishna Vangaveeti

Date: 09.12.2019

Contents

1. INTRODUCTION

This report includes the modelling and simulation results of BoiP system with Closed Loop control in Xcos. PID controller is tuned to reduce the error magnitude for both continuous and discrete systems. An approach has been made to include jumping reference concept to validate the tuned PID parameters.

2. OBJECTIVES

1. To create an Xcos model for BioP subsystem with motor angle β as input and position of ball as output.
2. Normalise the length of plane with 0.0 as midpoint and 1.0 as one end of the plane.
3. Create a closed loop by including continuous PID controller and tune the parameters with P=0.7, I=0.2 and D=0.4 as initial parameters.
4. To replace the continuous PID controller with the discrete PID block.
5. To implement jumping reference to validate the tuned PID.

3. EQUATIONS USED

3.1 Continuous PID equations:

$$u(t)=K_p*e(t) + K_i*\int_0^t e(\tau)d\tau + K_d*(de/dt) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ (1)$$

The controller parameters are proportional gain K_p, integral gain K_i and derivative gain K_d. u is the control signal in the above equation. e is the error.

The controller can also be written in the format mentioned below.

$$u(t)=K_p*\left(e(t) + K_i*(1/T_i)*\int_0^t e(\tau)d\tau + T_d*(de/dt)\right) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ (2)$$

T_i is the integral time constant and T_d is the derivative time constant. The proportional part deals with present value of error, the integral part deals with average of the past value of errors and the derivative part acts on future errors which is based on linear extrapolation.

When Laplace transform is applied

$$C(s)=K_p + (K_i/s) + K_d*s \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ (3)$$

Low Pass Filter(LPF) sends low frequency values and blocks high frequency values there by reducing the noise. Derivative term is modified for maximum reduction of noise. The modified derivative term is given below.

$$C(s)=K_p + (K_i/s) + (N*K_d /(1+N/s)) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ (4)$$

Here N is the filter coefficient.

3.2 Discrete PID equations:

To obtain discrete PID equations we can apply forward Euler, backward Euler approximations to continuous PID equations. So, we get a new set of equations which are in z domain(complex frequency domain representation). Since for a computer/processor, time is not continuous, we use discrete systems. For this experiment we used Backward Euler approximation equations. The equations for Discrete PID is shown below.

$$C(s)=K_p + ((K_i*T_s*z)/(z-1)) + (K_d*N*(z-1)/(1+N*T_s)*z-1) \quad \ldots\ldots\ldots\ldots\ldots\ldots (5)$$

From the above equation we can conclude that sampling period is affecting the derivative and integral terms.

Ts is sampling time which is taken as 0.01 in the experiment.

N is the filter coefficient and the value chosen as 60.

4. IMPLEMENTED XCOS MODELS

4.1 Continuous PID

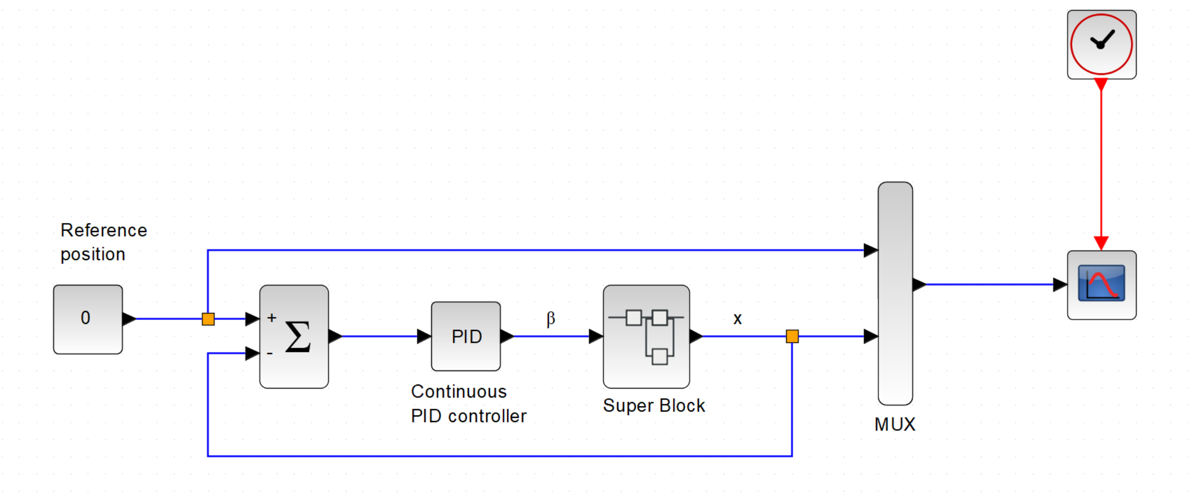

Fig. 1: Xcos model of continuous PID control

An Xcos model is implemented with Continuous PID with feedback loop as shown in Fig.1. Middle position of the plain is provided as reference. Super block takes angle of motor (β) as input and gives position (x) as output. Position is then compared with reference position and error is calculated. Based on the error the input value β is controlled by PID controller. PID is tuned with parameters P=2.8, I=0.01 and D=1.3.

4.2 Discrete PID

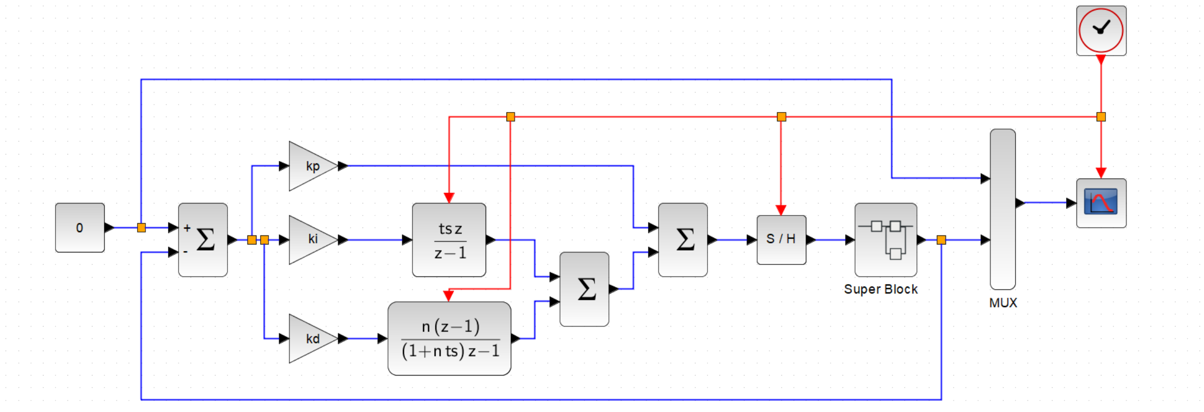

Fig. 2: Xcos model of Discrete PID control

Discrete time PID means the continuous time is converted to discrete time. It is sampled thereafter. For this sole purpose, we use Sample and hold block which samples the values of PID and holds it's value for specified period of time. Z transformation converts the continuous time signal to discrete time signal. The controller receives measurement from the system, it processes them and sends the control signal to the actuator.

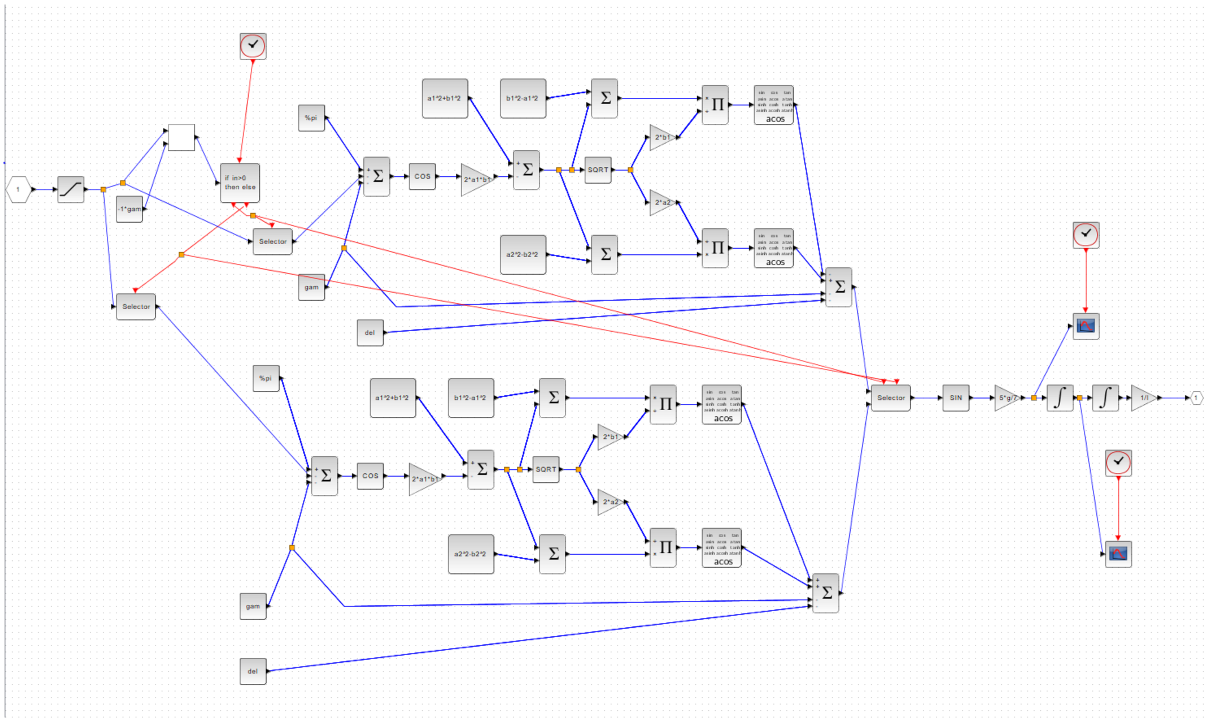

Fig. 3: Xcos implementation of Super Block

5. RESULTS

5.1 Continuous PID

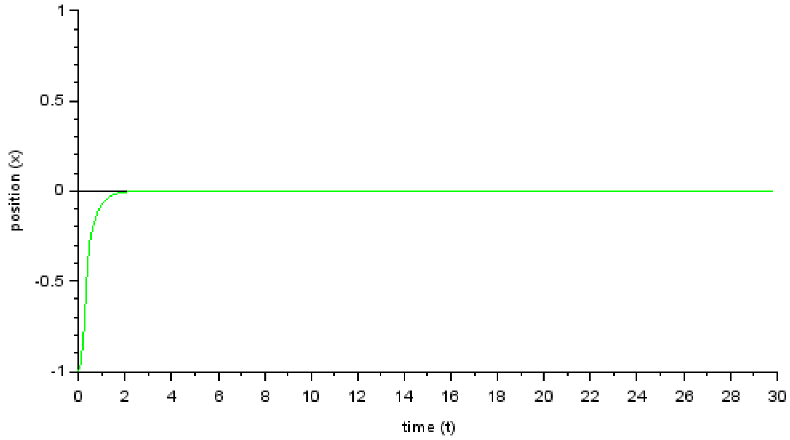

Fig. 4: Position vs Time plot for Continuous PID

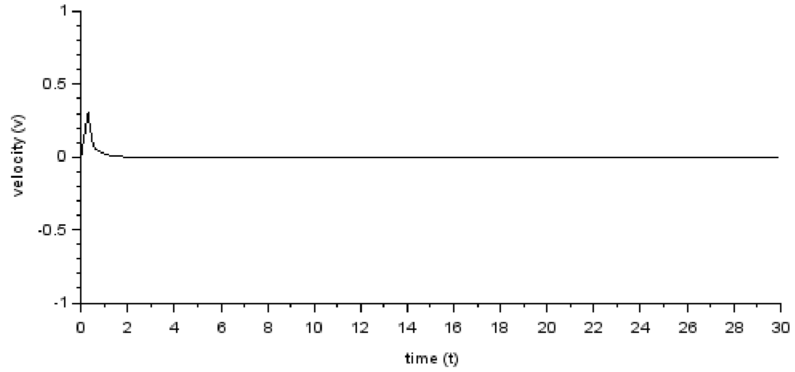

Fig. 5: Velocity vs Time plot for Continuous PID

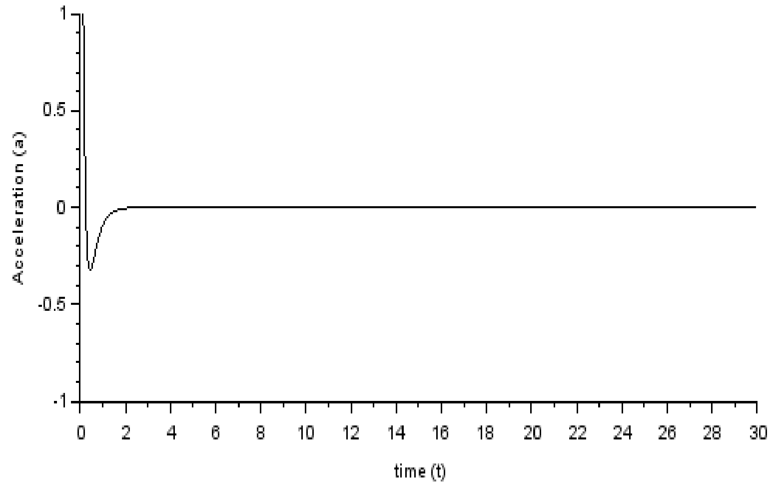

Fig. 6: Acceleration vs Time plot for Continuous PID

5.2 Discrete PID

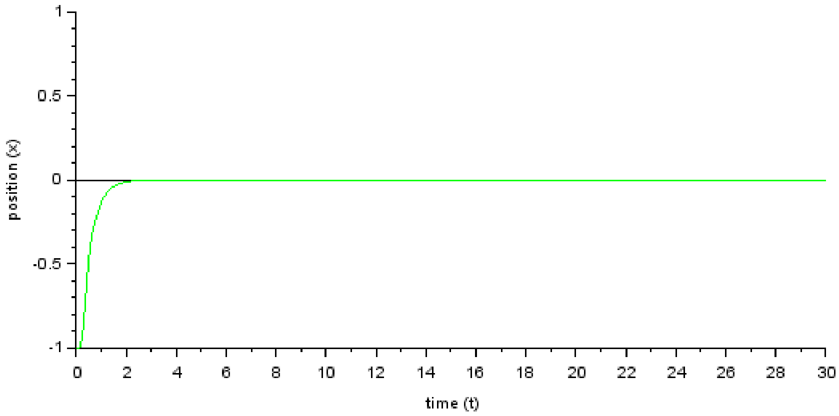

Fig. 7: Position vs Time plot for Discrete PID

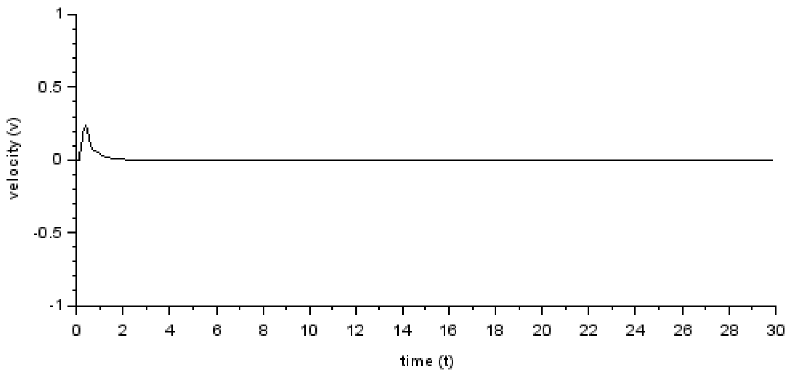

Fig. 8: Velocity vs Time plot for Discrete PID

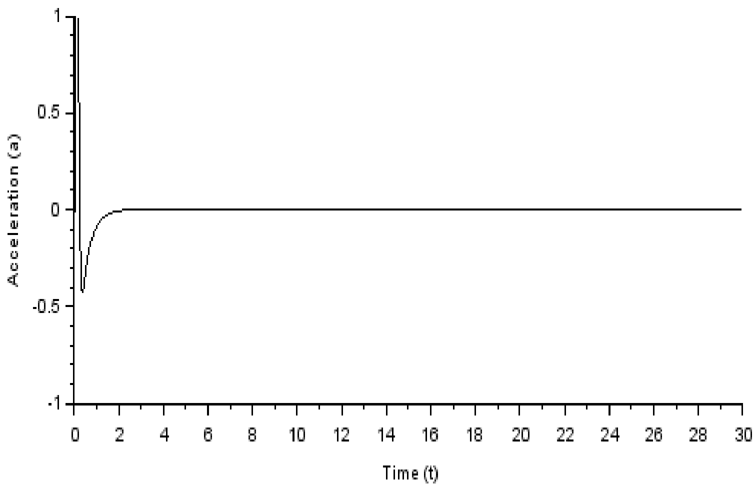

Fig. 9: Acceleration vs Time plot for Discrete PID

Observations:

1.) There is no overshoot in the position graph.

2.) Settling time of position graph is around 1.8 seconds.

3.) Steady state error is completely compensated.

6. DISCUSSIONS

1) *What are the expected responses of the controlled system when P-, I-, and D controllers are individually tuned?*

- *P-Controller* gives output which is proportional to current error e(t). Comparing the set point value with feedback process value, the resulting error is multiplied with proportional constant to get the output. Speed of the response is increased due to increase in sensitivity when the proportional constant Kc increases.
- *I-Controller* is very much effective in summing up the error in process and minimizing the steady state error deviation. As integral constant Ki increases, error stays for long time implying the speed of the controller is low. Thus, Integral controller have the effect of eliminating the error, but it can make the transient response even worse. It affects the stability of the system.
- *D-Controller* is used to speed up the error change in time by predicting the future behaviour of error. So, it will increase the stability of the system by improving the overshoot, rising time and the settling time. By increasing the derivative constant Kd, speed of the response increases as it gives a kick start for the output. It affects the transient state response.

2) *Why is it important to limit the input signal (angle β) to the controlled system? How can it be done? What should be the limitation?*

From the first exercise, it was observed that the graph shows a linear approximation within a certain range of angle β.

With the use of a saturation block, the value of input signal (angle β) can be limited and the limiting range can be mentioned in the block which is -40 degrees to +40 degrees.

3) *Why and where do you use limited integrals in the controlled system? What are the limits?*

Limited integrator can be used to prevent the output of the integral from exceeding specified levels. With the use of this Limited integrator, when the output of the integral reaches the limits, the integral action gets turned off preventing integral wind up determining the limit as the output.

We use two stages of integration in BOIP system to detect the position of the ball on the plane and to balance it. Integrating the acceleration equation (relation between the acceleration of the ball and the angle α) gives velocity equation which on further integration gives the position equation.

For integrating the acceleration equation, we use the initial (zero), minimum and maximum value of the velocity which can be attained by the ball on the plane. V_max is given by: $vmax = \sqrt{(20/7) g . \sin(\alpha) . l} = 0.537\ m/s$. This maximum velocity is the upper and lower limits for the velocity integration since this would be the extreme velocity that the ball can reach.

For integrating the velocity equation, we use the limits -l to +l, the extreme positions that the ball can travel (where 2l is the length of the inclined plane). We consider pivot position as the reference or zero position.

4) *Add a jumping reference for the system that randomly picks a new reference value from [0.3; 0.5; 0.7] and [-0.3; -0.5; -0.7] every 10s alternatively. An example may be [0.3; -0.5; 0.7; -0.3; 0.5; -0.3; ……]. Plot the desired ball positions and the actual ball positions for 10 jumps. What time period makes sense for the simulation interval? Can the system keep up? Why and why not?*

A random generator is given as input to the Event selector to trigger it and generate a random signal, which is then passed to Selector block to select one of the given input values randomly according to the random input and gives it as output to the reference block. Thus, When the random input changes every 10 seconds, the system positions the ball to the random reference positions.

To observe how the system behaves based on reference value change interval, we can reduce the time interval further. On reducing the reference value change interval to 2 seconds which is still greater than the settling time of system, the ball can attain all the mentioned positions. But, when the reference value change interval is further reduced to a value less than the settling time of system i.e., 0.5 seconds, then the ball won't be able to attain these positions because the reference value changes before ball reaches previous reference and it is not able attain required position.

So, the time interval of random change of reference value should be greater than the settling time of the system for the ball to reach the reference position.

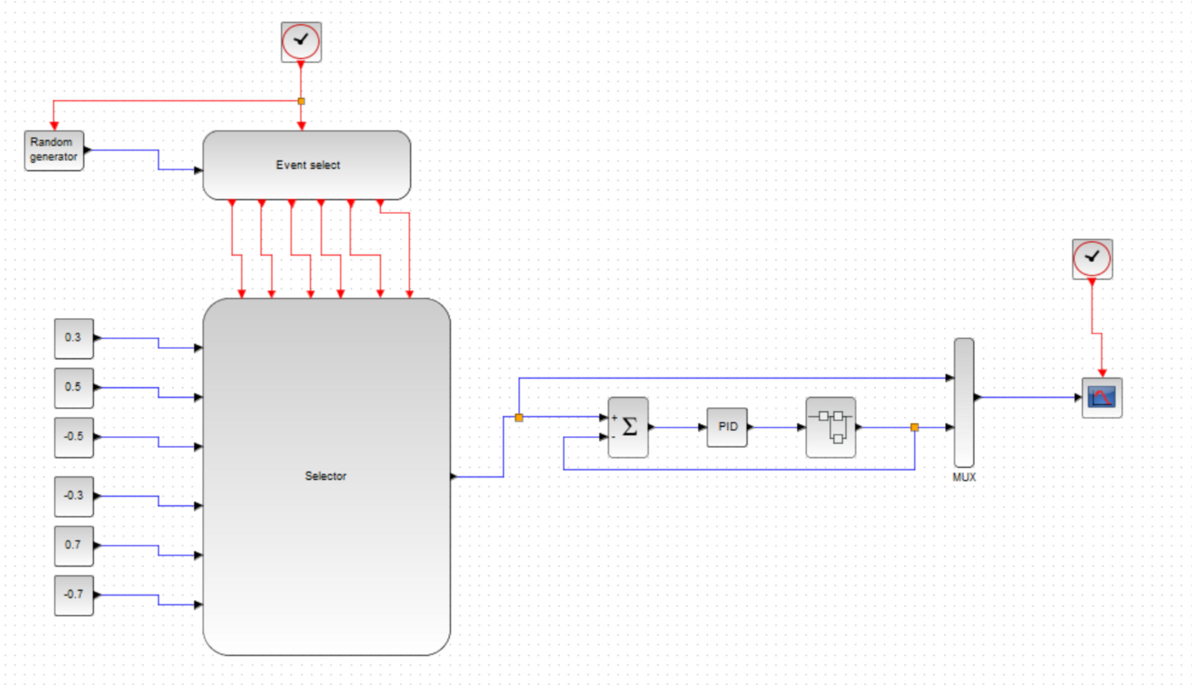

Fig. 10: Xcos model of continuous PID control with jumping reference

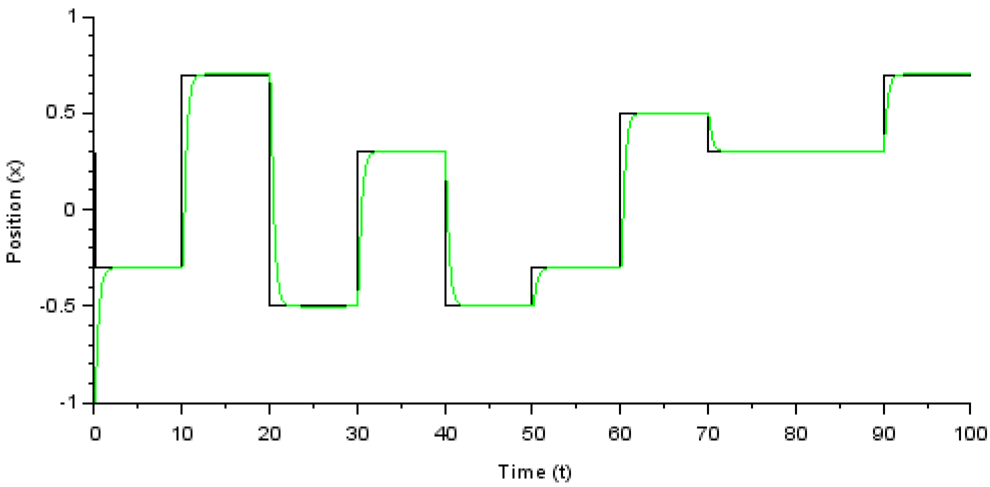

Fig. 11: Position vs Time plot for continuous PID control with jumping reference

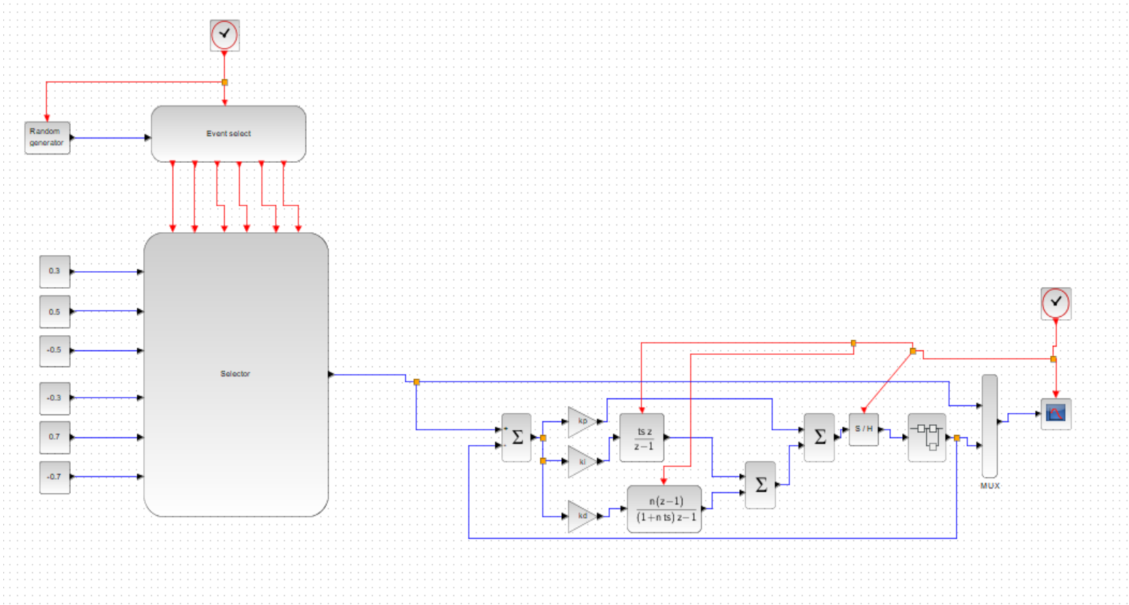

Fig. 12: Xcos model of discrete PID control with jumping reference

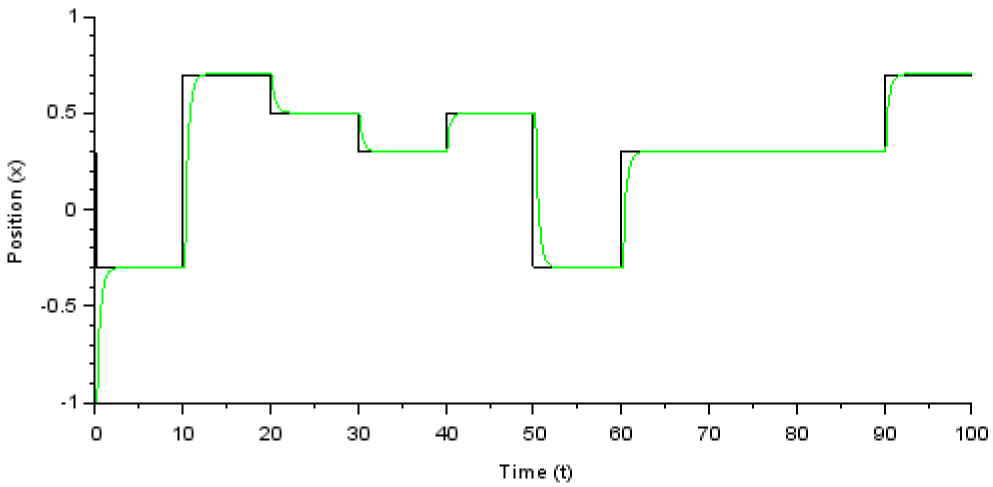

Fig. 13: Position vs Time plot for discrete PID control with jumping reference