Heated Liquid Tank

FM1220-1 20H Automatic Control

University of South-Eastern Norway

Faculty of Technology and Engineering, Campus Porsgrunn

Submitted by:

Dhanush Wagle

Student Number: 238810

MSc. Electrical Power Engineering

Fall 2020

Submitted to:

Professor Finn Aakre Haugen, PhD

Date: September 14th, 2020

Table of Contents

1. Description and Design

1.1 System Description

Figure 1 shows a heated tank. Liquid (assumed water) flows into and out of the tank. The volume of liquid is constant (this can be realized with overflow or level of regulation). The inflow and outflow are thus equal. There is a heat transfer between the liquid and the air outside the tank. In the tank there are homogeneous conditions thanks to a mixer (there is this no spatial variations in temperature). It is assumed that the mixer does not add power to the liquid. It is assumed that there is a time delay in the response in the temperature if there is a change in the suppled power. A time delay will be observed in any practical tank due to the unavoidable imperfect mixing.

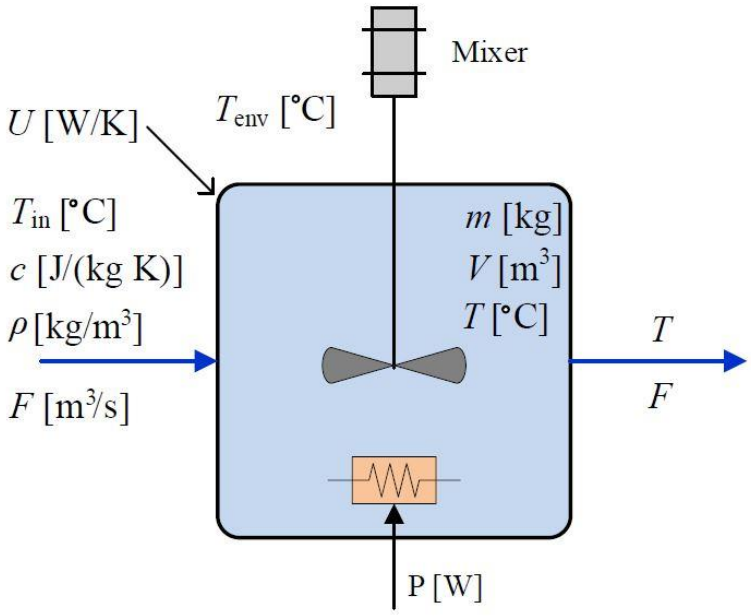

Figure: 1

1.2 Variables and Parameters

Variables and parameters of the heated tank are defined in Table 1.

Symbol	Value (default)	Unit	Description
P	0	W	Supplied power
T	20	°C	Temperature of liquid
T_{env}	20	°C	Environmental temperature
T_{in}	20	°C	Temperature of liquid inflow
F	$0.25{\cdot}10^3$	m^3/s	Liquid flow
c	4200	J/(kg·K)	Specific heat capacity of liquid
ρ	1000	kg/m^3	Density of liquid
V	0.2	m^3	Liquid volume in tank
U	1000	W/K	Heat transfer coefficient of tank
τ	60 s	s	Time delay in the temperature response

Table: 1

1.3 Overall Block Diagram

Figure 2 shows an overall block diagram of the heated tank.

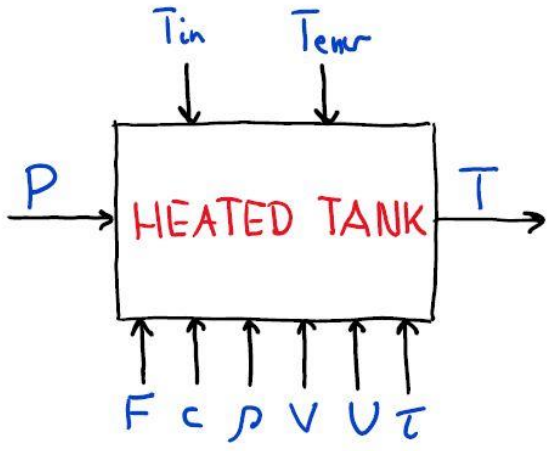

Figure: 2

1.4 Mathematical Model

A mathematical model of the temperature variation, based on energy balance of the liquid in the tank, is:

$$c\rho V T(t)' = P(t-\tau) + c\rho F\left[T_{\text{in}}(t) - T(t)\right] + U\left[T_{\text{env}}(t) - T(t)\right]$$

1.5 Tasks

1. **Implementation:**

   Implement a simulator of the tank heater in Python. The simulator must be implemented with "native" code in For Loop based on the Euler Forward discretization of the model (a built-in simulation functions of Python should not be used). Select an appropriate time-step yourself. The following variables should be plotted: T, Tin, Tenv, P. (All the temperatures should be plotted in one subplot, and P in another subplot).

2. **Simulation:**

   Check that the simulator behaves correctly i.e. according to the give model both dynamically (as seen from the transient responses) and in steady state. In this check, you can apply a step change n P. Check points: Is the time delay and time constant in the response in T the same as according to the model?

3. **Stability of the simulator:**
   Demonstrate that the simulator becomes numerically inaccurate, and possibly unstable, if you select a (too) large simulator time step.

2. Python Code

- *First the pyplot and numpy module is imported:*

```
import matplotlib.pyplot as plt
import numpy as np
```

- *Then the time is set:*

```
ts = 1  # Time-step [s]
t_start = 0  # Start Time[s]
t_stop = 2000  # Stop Time[s]
N_sim = int((t_stop-t_start)/ts) + 1 # Total Simulation Step
```

- *Then the constant and parameters are defined:*

```
F = 0.00025 # Liquid flow, m^3/s
c = 4200 # Specific heat capacity of liquid, J/kg.K
rho = 1000 # Density of liquid, kg/m^3
V = 0.2 # Liquid volume in tank, m^3
U = 1000 # Heat transfer coefficient in tank, W/K
t_delay = 60  # Time delay, seconds
```

- *Next, time delay is initialized:*

```
u_delayed_init = 0.0
N_delay = int(round(t_delay/ts)) + 1
delay_array = np.zeros(N_delay) + u_delayed_init
```

- *Again, arrays are initialized to store the values:*

```
T_plot_array = np.zeros(N_sim)
T_env_plot_array = np.zeros(N_sim)
T_in_plot_array = np.zeros(N_sim)
```

```
P_plot_array = np.zeros(N_sim)
t_plot_array = np.zeros(N_sim)
P_delayed_plot_array = np.zeros(N_sim)
```

- *The initial condition are set:*

```
# P = 0 # Supplied power, watts
T = 293 # Temperature of liquid, kelvin
T_env = 293 # Environmental temperature, kelvin
T_in = 293 # Temperature of liquid inflow, kelvin
```

- *Now, for-loop is created for the step input of the power between 0-10kw*

```
for k in range(0, N_sim):

    t = k*ts  # Time

    if (t >= t_start and t < 250):
        P = 0
    elif (t >= 250 and t < 500):
        P = 10000
    elif (t >= 500 and t < 750):
        P = 0
    elif (t >= 750 and t < 1000):
        P = 10000
    elif (t >= 1000 and t < 1250):
        P = 0
    elif (t >= 1250 and t < 1500):
        P = 10000
    elif (t >= 1500 and t < 1750):
        P = 0
    elif (t >= 1750 and t <= t_stop):
        P = 10000
```

- *The time delay is written as:*

```
P_delayed = delay_array[-1]
delay_array[1:] = delay_array[0:-1]
delay_array[0] = P
```

- *Using, Euler-forward discretization for the change in Temperature:*

```
dT_dt = (1/(c*rho*V))*(P_delayed+(c*rho*F*(T_in - T))+(U*(T_env - T)))
T_p1 = T + (ts*dT_dt)
```

- *Storing the values for plotting:*

```
t_plot_array[k] = k
T_plot_array[k] = T_p1
T_env_plot_array[k] = T_env
T_in_plot_array[k] = T_in
P_plot_array[k] = P
P_delayed_plot_array[k] = P_delayed
```

- *Shift time from T+1 to T:*

```
T = T_p1
```

- *Finally, all the values are plotted:*

- *Tin, T, Tenv vs time plot:*

```
plt.subplot(2, 1, 1)
plt.plot(t_plot_array, T_plot_array, 'r')
plt.plot(t_plot_array, T_env_plot_array, 'b--')
plt.plot(t_plot_array, T_in_plot_array, 'y--')
```

```
plt.legend(labels=('T', 'T_Env', 'T_In'), loc='upper right')
plt.grid()
plt.xlabel('t [s]')
plt.ylabel('[Kelvin]')
```

- *P, P_delayed vs time plot:*

```
plt.subplot(2, 1, 2)
plt.plot(t_plot_array, P_plot_array, 'g')
plt.plot(t_plot_array, P_delayed_plot_array, 'r')
plt.grid()
plt.xlabel('[Seconds]')
plt.ylabel('[Watt]')
plt.legend(labels=('P_Real', 'P_Delayed'), loc='upper right')
plt.show()
```

3. Results

1. Implementation

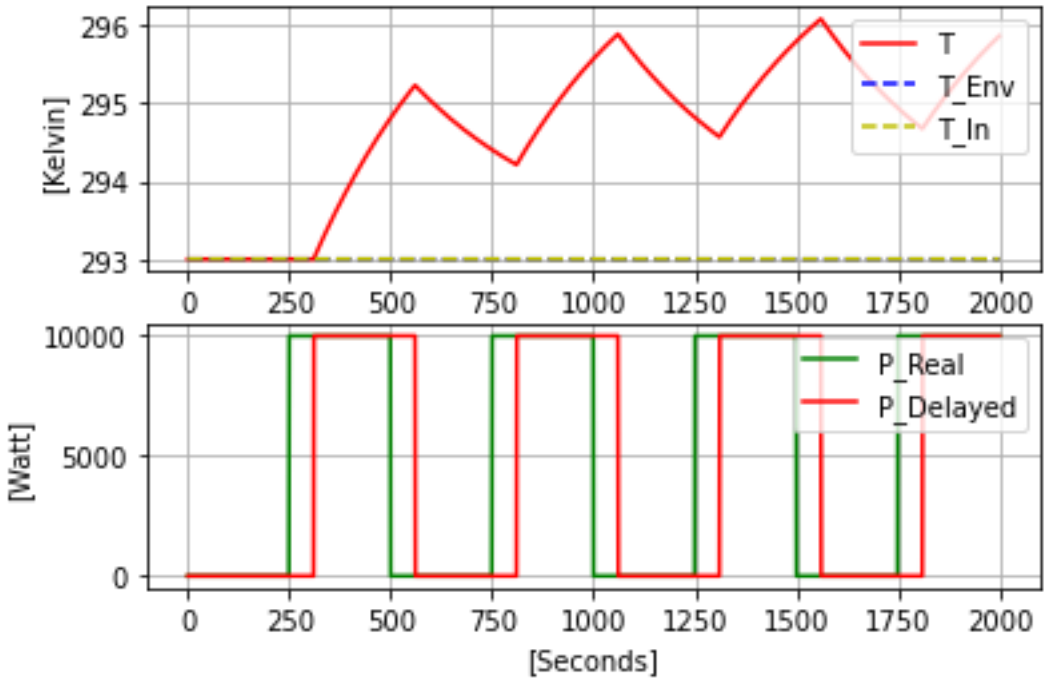

Figure: 3

Figure 3 shows two plots; first plot shows the temperatures versus the time plot where **T_Env** and **T_in** are kept constant to 20 degree Celsius or 273K (all degree celcius are converted to kelvin for this model simulation) and **T** rises from initial 293K to 295.1 and falls again to 294.2K for 0KW input power, hence it changes with step power input of 0-10 KW; the second plot shows the step **power** input of 10 KW from 0 to 2000 seconds with time delay of 60 seconds which means the power is set to 10KW at 250 seconds but actually is set only at 310 seconds.

2. Simulation

The simulator behaves correctly both dynamically and in steady-state condition. Step change of 10 KW was applied as power and the time delay was kept 60 seconds with 1 second of Time constant as previously defined in the model.

3. Stability of the simulator

Selecting time step of 100 seconds.

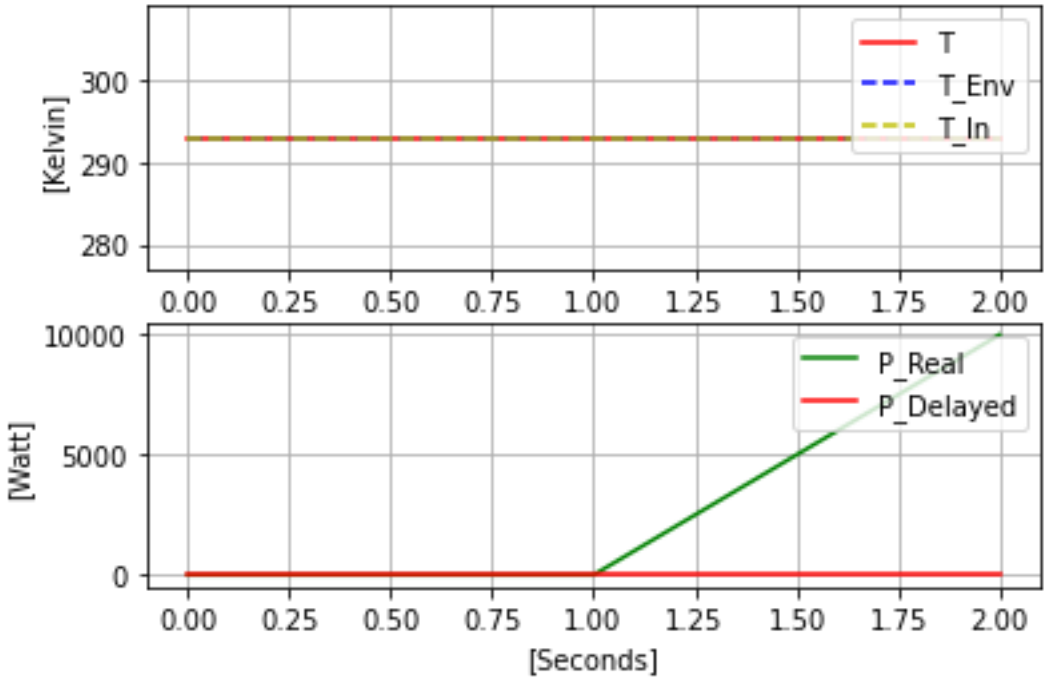

Figure: 4

Figure 4 shows that all the temperature values are same with little or no change and the step input of power is also inaccurate, hence it is clear that selecting too large time scale leads to numerically inaccurate, and possibly unstable simulator.