System Design Laboratory

Hybrid Automaton

Daniele Nicoletti
`VR472590`
Giacomo Canella
`VR481630`

a.a.: 2021/2022

Contents

0.1 Introduction

In this exercise we are asked to extend the example seen in class. The system we need to extend is composed by 2 tanks (a main tank and a reserve one) connected to each other.
There is the possibility to control the return flow (the one in the connection that goes from reserve tank to the main tank) by acting on a "valve" variable.
Along with the tanks we have an additional automaton that is the one representing the weather.

This system needs to be extended in the following ways:

- Model depletion of both tanks (i.e., water level cannot go below zero);
- Model overflow of both tanks (i.e., water level cannot go over a given constant, say 2 meters; for the pool), assuming the backup tank is the same as the pool
- Modify the constants in the current system in a way that overflow occurs;
- Introduce an automaton that monitors the amount of wasted water due to overflow (hint: can be the integration of the instantaneous precipitation (summarized with the r variable) along the time spent in the overflow condition);
- Modify the constants in all automata to cause both depletion and overflow (hint: work with the duration in the sunny and stormy locations), when the controller is disabled, i.e., the valve aperture a variable in the equations of the tanks automaton is instead a constant with a chosen fixed value between 0 and 1;
- Re-enable the controller and see for which of its parameters, if any, both depletion and overflow can be avoided;
- Try out a proportional controller in place of the hysteretic controller (hint: adapt the pro- portional controller shown in Lecture 7, while discussing dominance checking).

0.1.1 Starting system presentation

Before exposing out solution we will like to briefly introduce the automatons that compose the starting system.

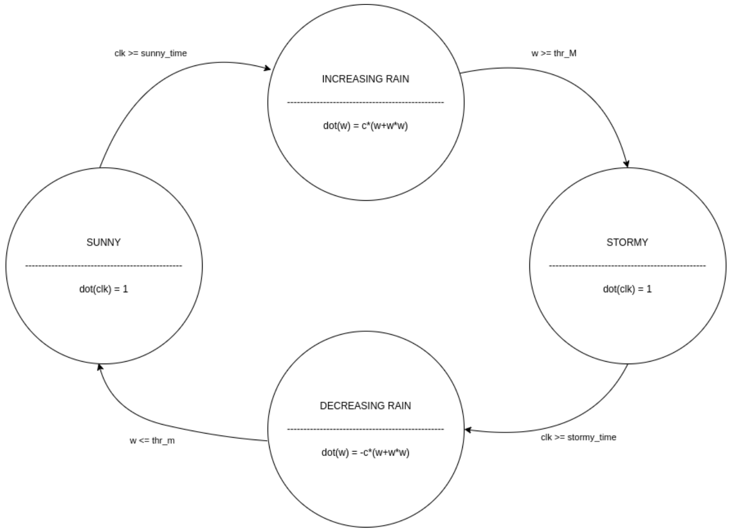
Tanks automaton

The automaton representing the tanks has been modeled in class by using a single location which represent the tanks in their normal operational state. In this location the equations for the two tanks are modeled as:

- "$dot(z) = a * beta * d - alpha * z + r * gamma$" , for the main tank;
- "$dot(d) = -a * beta * d + alpha * z$" , for the reserve tank.

a being the variable that control the return valve opening/closure;
z,d being the variable representing the level of the the two tanks;
r being the variable representing the rain/sunny behaviour;
beta,gamma being the constant representing respectively the magnitude of the flow from the reserve to the main tank and the second representing the magnitude of the raining/drying caused by the weather.

Weather



The automaton describing the weather has been modeled using 4 location:

- **Sunny**;
- **Increasing rain**;
- **Stormy**;
- **Decreasing rain**.

In this automaton a **"w"** variable is used to calculate the behaviour of the weather in the following way:

- **if w $\geq$ of a certain threshold, the weather will transition from increasing rain to stormy**;
- **if w $\leq$ of a certain threshold, the weather will transition from decreasing rain to sunny** ;

The equations for "w" are:

- $dot(w) = -c * (w + w * w)$, for the **decreasing rain location**;
- $dot(w) = c * (w + w * w)$, for the **increasing rain location**.

c being the constant representing the magnitude at which the weather changes.

Regarding the **Sunny,Stormy** locations, their duration is regulated by a clock. This allows to decide the amount of time spent in sunny/stormy location, which means having the maximum amount of rain/drought ($r = 1$ or $r = -1$).
r is the variable representing how much rain/drought is being given by the weather.
It is comprised in [-1,1] (-1 = maximum drying, 1 = maximum rain).

0.2 Depletion modeling

The Depletion for the tanks automaton has been modeled by adding 1 location and 2 transitions to the original automaton.

0.2.1 Location

The location added (**Depleted**) represent the state of depletion and the equations for this location wants to resemble the following idea:
"When the automaton is in the depleted state, the level of water in the main and in the reserve tank are fixed to zero ".

This, in terms of equations translates into:

- $let(z) = thr\_dry$;
- $let(d) = thr\_dry$.

Where **thr_dry** is a constant with value = 0.

0.2.2 Guards

The Guards are the following:

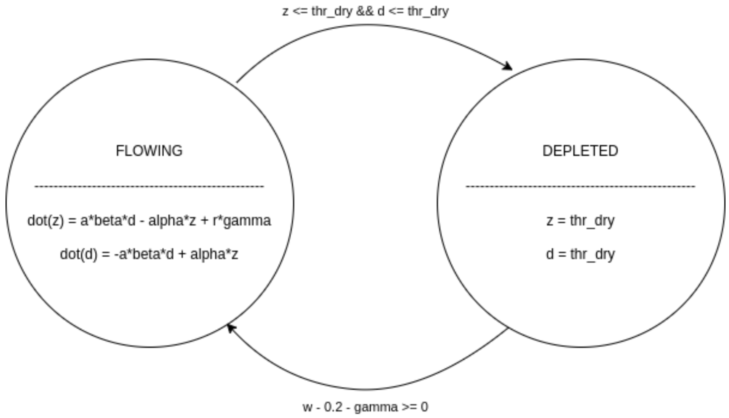
- $z <= thr\_dry$ && $d <= thr\_dry$, for entering the **Depleted** location from the **Flowing**.
- $r * gamma >= 0\_dec$, for exiting the **Depleted** location and enter the **Flowing**. Which is as soon as the rain start pouring, we will have water in the system again.

0.2.3 Events

The events added for modeling the depletion are:

- **drought** : for the Flowing to Depleted transition;
- **flood** : for the Depleted to Flowing transition.

Here is the automaton:



0.3 Overflow modeling

The Overflow for the tanks automaton has been modeled by adding 1 location and 2 transitions to the original automaton.

0.3.1 Location

The location added (**Overflow**) represent the state of overflow and the equations for this location wants to resemble the following idea:
"When the automaton is in overflow, the level of z is fixed to a certain threshold while the amount d can still increase because the flow from main to reserve tank is always open".

So the equations for the **Overflow** location are:

- $let(z) = thr\_ovf$;
- $dot(d) = -a * beta * d + alpha * z$

The transitions added are:

- From **Flowing** to **Overflow**;
- From **Overflow** to **Flowing**.

0.3.2 Guards

The Guards are the following:

- $z > thr\_ovf$, for the transition to the **Overflow** location.
- $a * beta * d + r * gamma <= alpha * z$, for the transition from **Overflow** to the **Flowing** location. Which means that the outgoing flow from the pool is greater that the ingoing flow of water.

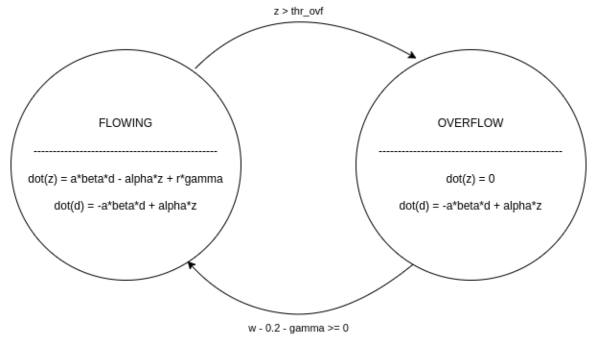
thr_ovf being a constant that determines the maximum amount of water that can be contained in z before going in overflow (in out system is set to 2.0) .

0.3.3 Events

The events added for modeling the overflow are:

- **overflowing** : for the Flowing to Overflow transition;
- **stabilizing** : for the Overflow to Flowing transition.

The automaton diagram is the following:



0.4 Overflow calculator modeling

To model the calculation of the spilled water during the overflow of the main tank we created another automaton.

0.4.1 Locations

The calculator automaton is composed by 2 locations:

- **active**;
- **idle**.

0.4.2 Guards

The guard are the following:

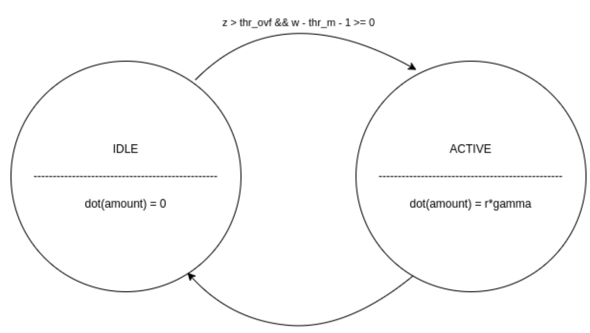
- $z >= thr\_ovf w - thr\_m - 1 >= 0\_dec$,which means **"if z is in overflow and it is still raining"**. This is for the transition from **Idle** to **Active**;
- $z < thr_o vf || w - thr_m - 1 < 0_d ec$, for the transition from **Active** to **Idle**.

0.4.3 events

The events for this automaton are:

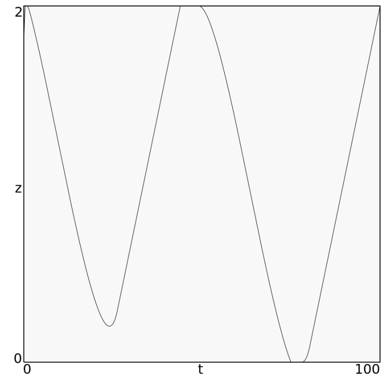
- **over** that is used to enter the active location;
- **under** that is used to enter the idle location.

The automaton diagram is the following:

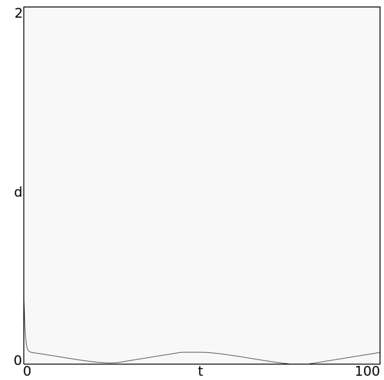


0.5 Cause Overflow and Depletion

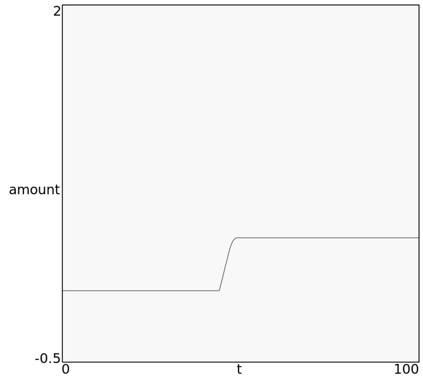
Here is the system under the conditions of overflow and depletion:
Z behaviour:



here is shown the Overflow(level of z over threshold) and Depletion(level of z to zero).
D behaviour:



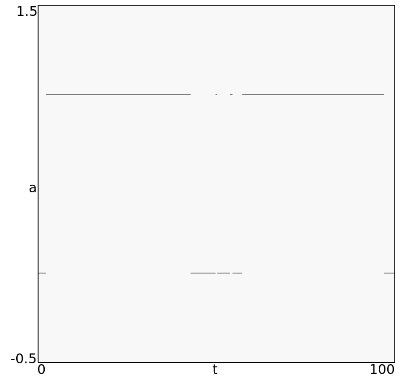
Amount of water wasted



To have both overflow and depletion conditions we acted on the length of the sunny/stormy period: now sunny time is 5 and stormy time is 20.

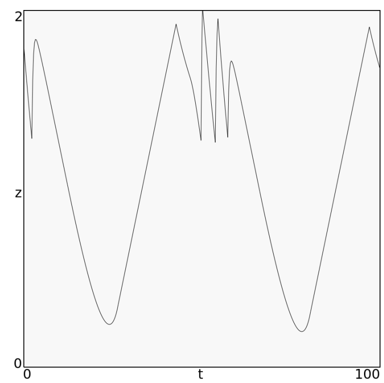
0.6 System under hysteresis Controller

The hysteresis controller, as seen in class, is a controller that acts by fully opening or fully closing the valve depending on certain thresholds.
Its behaviour is the following: (here is shown the value of **a** varible, representing the aperture of the valve).

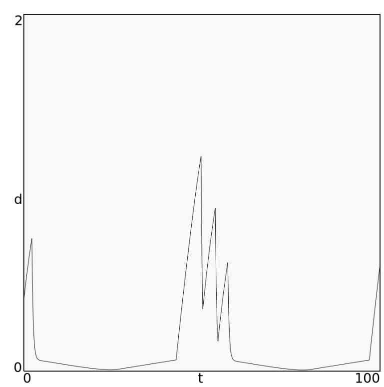


With this controller the behaviour of the system changes in the following ways:

Z:

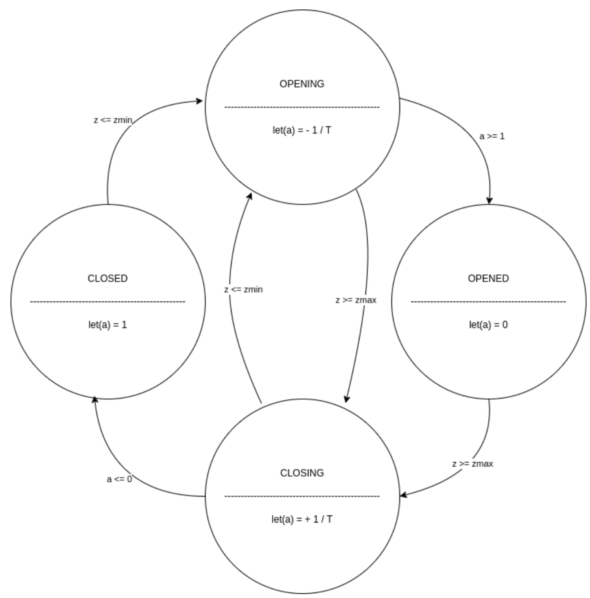


D:



0.7 System under Proportional Controller

The proportional controller, also seen in class, tries to apply a more intelligent control by adding 2 additional location between the **closed** and **opened**.



0.7.1 Locations

This automaton have 4 locations:

- **closed**
- **opening**
- **closing**
- **opened**

The **opening** and**closing** locations are characterized by the following equations:

- $dot(a) = +1/T$, for the **opening** location;
- $dot(a) = -1/T$, for the **closing** location.

Where T is a constant that regulates how fast the valve is closing/opening.

0.7.2 Guards

The guards are the following:

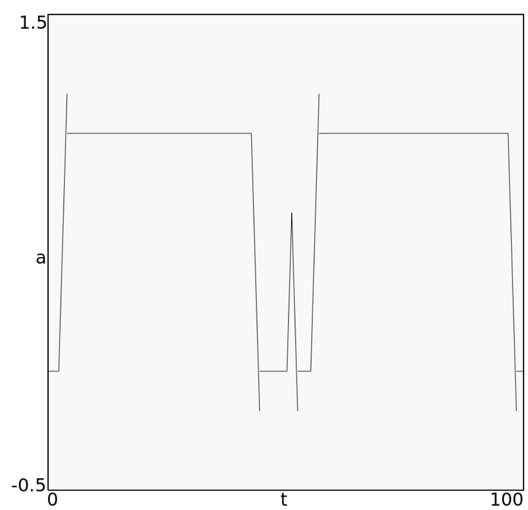
- $z \leq zmin$, passing from **closed** to **opening**;
- $a \geq 1$, passing from **opening** to **opened**;
- $z \geq zmax$, passing from **opened** to **closing**;
- $a \leq 0$, passing from **closing** to **closed**;
- $z \geq zmax$, passing from **opening** to **closing**;
- $z \leq zmin$, passing from **closing** to **opening**.

0.7.3 Events

The events for this automaton are:

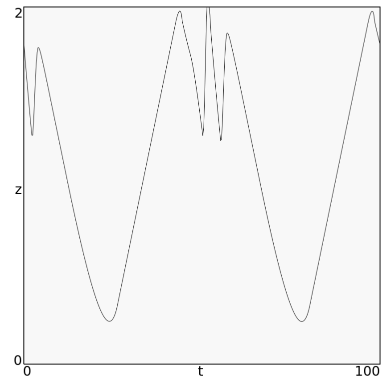
- stop_opening used transitioning from **opening** to **opened**;
- stop_closing used transitioning from **closing** to **closed**;
- can_open used transitioning from **closed** to **opening** and from **closing** to **opening**;
- can_close used transitioning from **opened** to **closing** and from **opening** to **closing**.

Its behaviour is the following: (here is shown the value of **a** varible, representing the aperture of the valve).

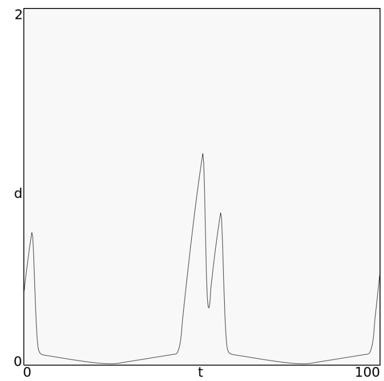


With this controller the behaviour of the system changes in the following ways:

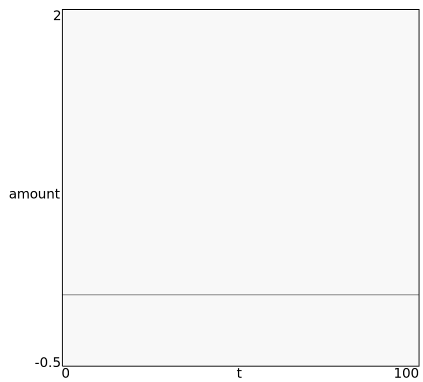
Z:



D:



Amount of water wasted:



As shown here, even if z reaches its maximum threshold, no water is wasted because its amount is kept precisely at the maximum level without ever going beyond, thanks to the controller.