**ContCarSim: Continuous Carousel Simulator documentation**

v0, Francesco Destro, October 14 2021

**Index**

[1. Process description 2](#_Toc87449559)

[2 How to use the simulator 4](#_Toc87449560)

[2.1 Inputs 4](#_Toc87449561)

[2.2 Simulation execution 7](#_Toc87449562)

[2.3 Outputs 8](#_Toc87449563)

[2.4 Disturbances and operating modes 11](#_Toc87449564)

[2.5 Setting up a control strategy 11](#_Toc87449565)

[2.6 Simulator scripts and functions: summary 11](#_Toc87449566)

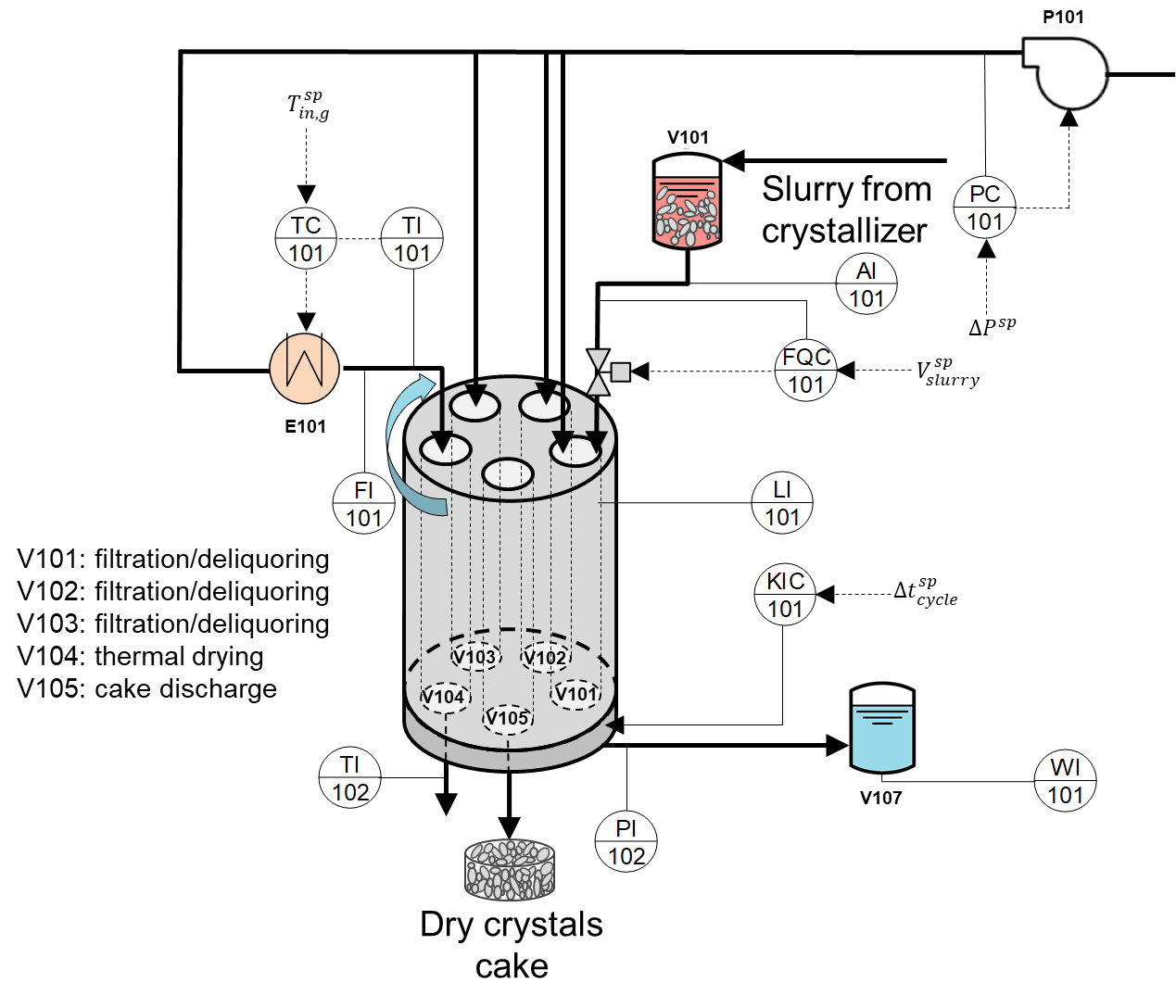
[3 Quality-by-design and quality-by-control challenges 12](#_Toc87449567)

[4 Sample case study 14](#_Toc87449568)

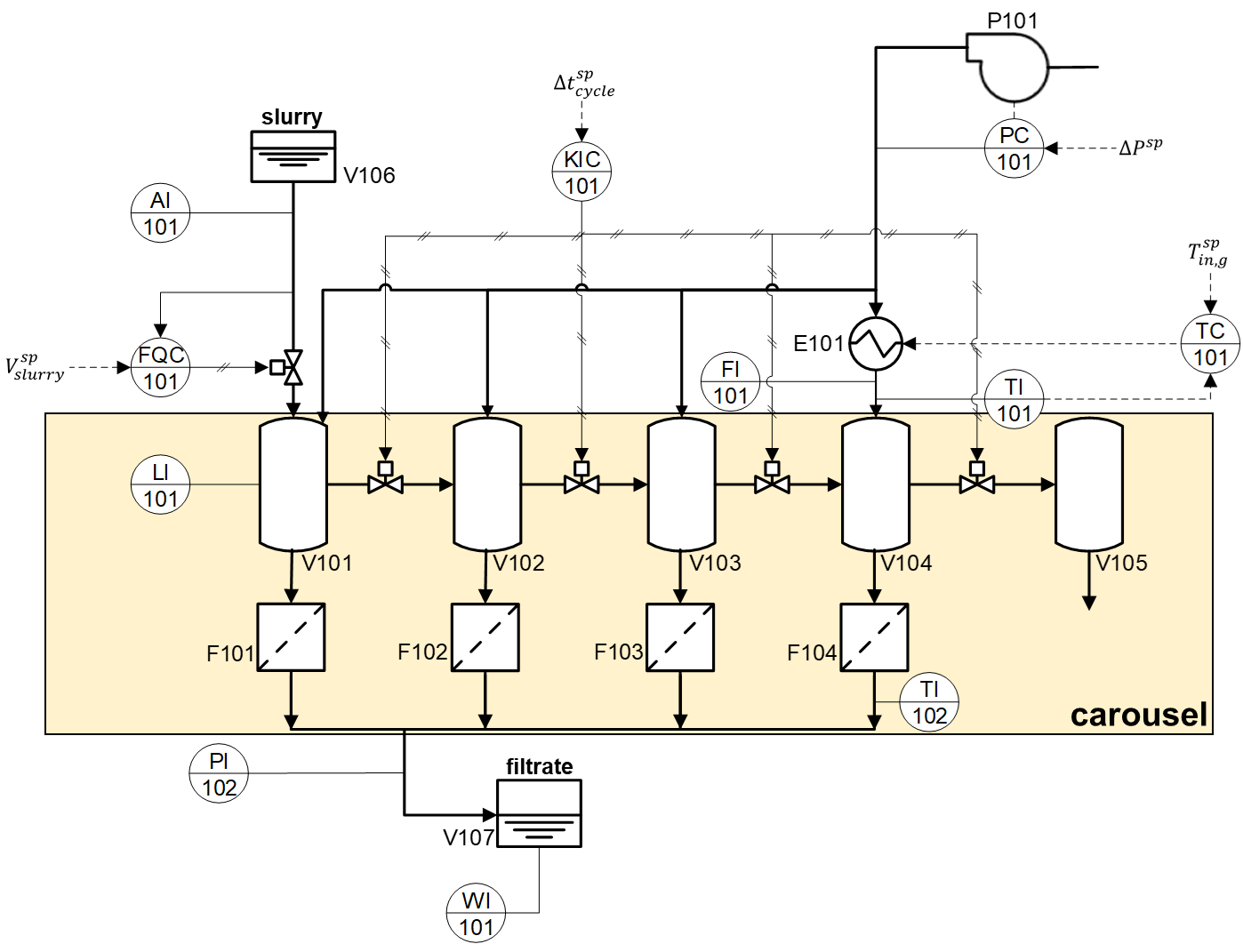
# 1. Process description

The carousel reproduced by the simulator is sketched in Figure 1. A schematic P&ID of the process is provided in Figure 2, with the legend of equipment, sensors and controllers reported in Table 1. The unit can continuously process an inlet slurry stream into a dry crystals cake. The slurry system considered in the simulator is composed by pure paracetamol crystals in a mother liquor composed by pure ethanol. The carousel features five cylindrical ports, each one of 15 mm diameter, which allow a maximum hold-up of 10 mL. The ports are embedded in a main cylindrical body, aligned to five processing stations (Stations 1-5). For illustrative purposes, in Figure 2 the stations are represented as vessels in series (V101-V105), although the actual layout of the carousel is as in Figure 1. Stations 1-4 present a filter mesh at the bottom (F101-104). Station 5 is, instead, open at the bottom for cake discharge, which is enabled by the action of a pneumatic piston (not shown). The pressure gradient for filtration and drying is provided by a compressor (P101), connected to the top section of all the stations, whereas all stations are maintained at atmospheric pressure on the bottom section.

The carousel operates in cyclic mode: processing cycles, during which every port processes batch-wise the material therein contained, are alternated to carousel rotations, during which the ports containing the material being processed are moved to the following station. Carousel rotations are logically represented in the P&ID as material streams, whose flows are controlled by FC-101. The alternating processing cycles and carousels rotations are interrupted when significant mesh fouling is detected: a cleaning-in-place cycle is then triggered. In this case, the material already loaded in the carousel ports is regularly processed during the following cycles, but no more slurry is loaded into

**

**Figure 1.** *Schematic drawing of the carousel for continuous integrated filtration-drying of crystallization slurries mimicked by the simulator. Filter meshes are placed at the bottom of V101-V04 (i.e., processing stations 1–4). Station 5, instead, is open for cake discharge. In physical carousels, controllers FQC-101 and KIC-101 are routines of the programmable logic controllers of the unit.*

****

**Figure 2.** *Logical**P&ID of the carousel illustrated in Figure 1. The equipment legend is reported in Table 1.*

the first station. Hence, at every cycle following cleaning-in-place initiation, an increasing number of ports will be empty. When all ports are empty, all meshes are automatically cleaned by sending a cleaning solvent into the carousel. In the simulator, mesh cleaning is assumed to occur instantaneously.

Stations 1-3 are dedicated to filtration and deliquoring, while in Station 4 thermal drying is carried out. In Station 5, only cake discharge occurs. Slurry processing occurs as follows. The crystallization slurry is fed to Station 1 at the beginning of every cycle, by keeping the valve between the slurry tank (V106) and Station 1 open. After slurry feeding, a subsequent filtration step starts in Station 1, and it continues until filtration ends. If a carousel rotation is triggered before filtration finishes, filtration will continue in Station 2 (and, possibly, in Stations 3 and 4). During filtration, the liquid contained in the slurry is filtered out of the port by the action of the pressure gradient generated by P101, and stored in filtrate collector V106, while the crystals are retained on top of the filter mesh, leading to cake formation. We distinguish between actual filtration, when there is a slurry hold-up on top of the cake being formed, and the subsequent deliquoring, during which the sole remaining liquid is the one retained inside the cake pores. Upon deliquoring, the liquid in the cake pores is mechanically displaced out of the cake by the action of the pressure gradient, until a certain pore saturation equilibrium is achieved. Filtration duration depends on the cake properties and on the pressure gradient itself. Depending on filtration duration, the cake can be partially deliquored in Stations 1-3, or it might even enter Station 4 with some slurry hold-up (drying cannot be properly conducted in this situation, which should be avoided). Thermal drying is performed in Station 4 by flowing a hot air stream through the cake.

**Table 1.** *Legend of Figures 1-2, including unit operations and ancillary equipment.*

|  |  |
| --- | --- |
| **Name** | **Description** |
| *Unit ID* |  |
| F101-F104 | Filter mesh below Stations 1-4 (respectively) |
| P101 | Compressor |
| E101 | Drying air electrical heater |
| V-101 | Carousel Station 1 |
| V-102 | Carousel Station 2 |
| V-103 | Carousel Station 3 |
| V-104 | Carousel Station 4 |
| V-105 | Carousel Station 5 |
| V-106 | Filtrate collector |
| V-107 | Slurry tank |
|  |  |
| *Controllers and sensors* |  |
| AI-101 | Slurry concentration sensor |
| FC-101 | Fed slurry volume controller |
| FI-101 | Flowmeter for drying air entering carousel ports |
| KIC-101 | Carousel rotation controller |
| LI-101 | Camera system measuring volume of fed slurry and cake height |
| PC-101 | Drying air pressure controller |
| PI-102 | Filtrate pressure indicator |
| TC-101 | Drying air inlet temperature controller |
| TI-101 | Thermocouple for drying air inlet temperature |
| TI-102 | Thermocouple for drying air outlet temperature |
| WI-101 | Scale for inferring filtrate flowrate |

# 2 How to use the simulator

Figure 2 shows the logical structure of the simulator, while Figure 3 elucidates the set of scripts and functions that make up the simulator, together with the order and the logics with which they are called.

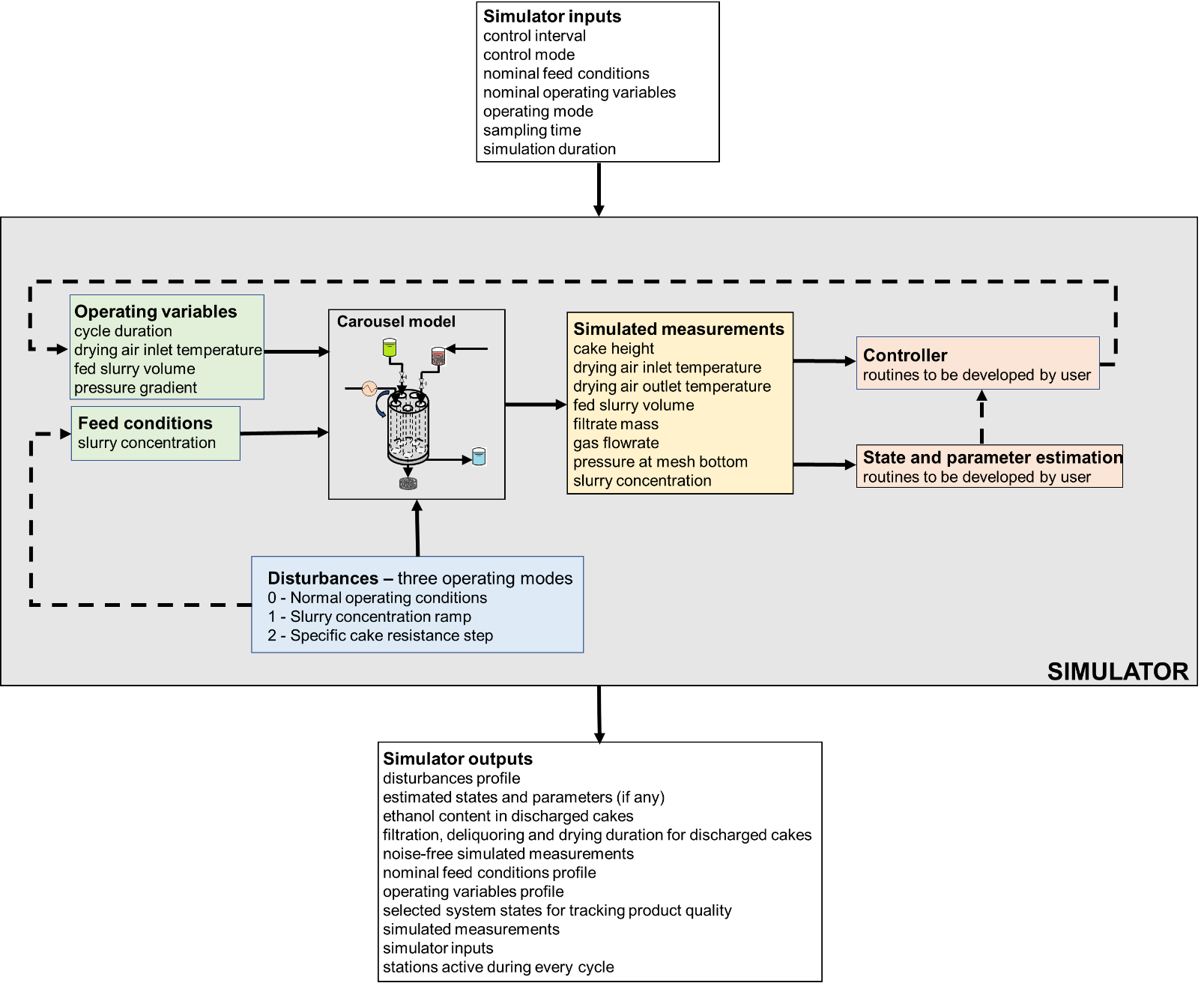
## 2.1 Inputs

The simulator inputs are reported in Table 2, and have to be set up in the script run\_carousel.m.

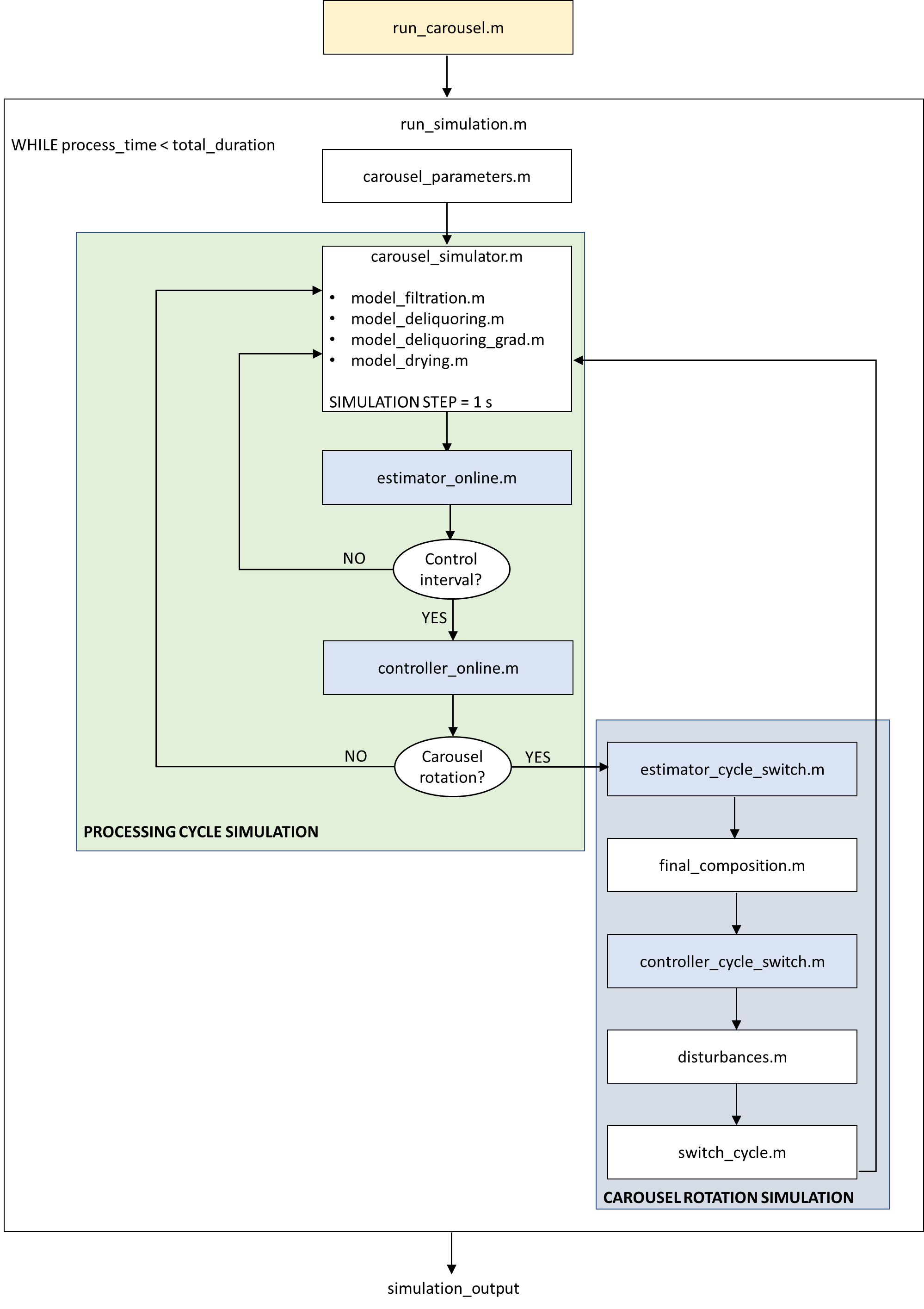
**Table 2.** Simulator inputs.

|  |  |  |  |
| --- | --- | --- | --- |
| **Variable name in** run\_carousel.m | **Variable** | **UOM** | **Admissible values** |
| control\_mode | Flag for selecting control strategy:  0: open-loop  1: closed-loop controller of sample case study (Section 4)  Other modes can be set up by the user | - | 0, 1, other values set up by user |
| operating\_mode | Flag for selecting operating mode: (see Section 2.4)  0: normal operating conditions  1: slurry concentration ramp change  2: specific cake resistance step change | - | 0, 1, 2 |
| total\_duration | Simulation duration | s | [0, +∞) |
| u\_nominal.t\_rot | Nominal cycle duration | s | [5, +∞)\* |
| u\_nominal.V\_slurry | Nominal fed slurry volume | m3 | [5×10-7, 1×10-6] |
| u\_nominal.dP | Nominal pressure gradient | Pa | [1×104, 2×105] |
| u\_nominal.Tinlet\_drying | Nominal drying air inlet temperature | K | [293, 353] |
| cryst\_output.conc\_slurry | Nominal slurry concentration | kg/ m3 | [50, 500] |
| control\_interval | Time interval at which run\_carousel.m is called | s | Multiples of 1 |
| sampling\_time | Sampling time for measurements and states in simulation output | s | Submultiples of 1 |

**\*MUST BE AN INTEGER**



**Figure 2.** *ContCarSim: schematic elucidating the logical simulator structure.*



**Figure 3.** *ContCarSim: computational structure of the simulator, showing the logics and the order with which the functions and scripts of the simulator are called. Yellow: script to be (optionally) edited with desired operating settings and then run. Blue: functions that can be edited for modifying the control strategy.*

## 2.2 Simulation execution

The simulation is initialized by running the script run\_carousel.m, after having set up the desired inputs. The simulation is carried out as shown in Figure 3. Function run\_simulation.m, handling the simulation routine, is automatically called. The parameters of the model are automatically retrieved from function carousel\_parameters.m.

Then, the simulation of the first processing cycle begins. The carousel\_simulator.m function is called for simulating 1 s of carousel operation. The estimator\_online.m function is then executed. By default, estimator\_online.m is an empty function, but it can be modified by the user for setting up parameter and state estimation routines.

Afterwards, carousel operation is simulated again for a duration of 1 s with carousel\_simulator.m, or, if the control interval specified by the user has been achieved, the controller\_online.m function is called, to update the value of the operating variables as specified by the control laws implemented in controller\_online.m. The operating variables are (Figures 1-2):

* The set-point of the inlet drying air temperature ;
* The set-point of the slurry volume fed to the carousel at every cycle ;
* The set-point of the cycle duration ;
* The set-point of the pressure gradient .

Within the default control mode 0, the process is operated at open-loop. Therefore, the set-points of the lower-layer controllers (FQC-101, PC-101, TC-101, and KIC-101) coincide with the nominal values set by the user (Table 2). If closed-loop routines are implemented, the lower-layer controllers set-points are instead adjusted during carousel operation, based on the control laws implemented in controller\_online.m.

The drying air inlet temperature , the pressure gradient , and the cycle duration are assumed to be perfectly controlled, namely the actual responses perfectly track the relevant setpoints. The fed slurry volume is instead subject to Gaussian fluctuations around the set-point, to reflect the behavior of real life carousels, as outlined in Section 2.4. Moreover, for cycles during which V-101 is empty due to the cleaning-in-place routine, is automatically set equal to zero. Function controller\_online.m also contains a sample closed-loop control routine (control mode 1), illustrated in the sample case study (Section 5).

Functions carousel\_simulator.m, estimator\_online.m, and controller\_online.m are subsequently called until the cycle time reaches the current cycle duration At that point, the following functions are called:

* estimator\_cycle\_switch.m: by default an empty function, that can be modified by the user for setting up parameter and state estimation routines;
* final\_composition.m: calculates the composition of the discharged cake, if there is any, and stores other variables that are outputs of the simulator;
* controller\_cycle\_switch.m: updates operating variables following control laws set up by user. The default implementation acts only on , setting it to zero for cycles in which V-101 is empty, and to the nominal fed slurry volume set up in run\_carousel.m in all the other cases;
* disturbances.m: updates the value of the disturbances for the following cycle;
* switch\_cycle.m: handles the carousel rotation routine.

Subsequent processing cycles and carousel rotation routines are simulated, until the set total process duration is reached. At that point, the simulation is terminated, and the simulation output object (simulation\_output.mat) is generated.

## 2.3 Outputs

The structure of simulation\_output.mat is elucidated in Table 3. Note that simulation\_output.mat is generated only if at least one cake has been discharged by the carousel, namely if a large enough total process duration has been specified, compared to the set cycle duration.

The fields of simulation\_output.mat are:

* states, storing the value assumed by selected system states during the simulation;
* measurements, storing the process measurements;
* disturbances, containing the profiles of the disturbances during the simulation;
* manipulated\_vars, storing the profiles of the manipulated variables;
* estimated\_states\_parameters, containing potential estimated states/parameters;
* feed, storing the slurry concentration profile during the simulation;
* settings, containing the settings that were set in run\_carousel.m before initiating the simulation.

**Table 3.** Structure of simulation\_output.mat.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Sub-field** | **Sub-sub-field** | **Variable** | **Description** | **UOM** |
| states | station1  station2  station3 | cake\_x | t | Readings of timer reinitialized at every carousel rotation – vector, step: sampling\_time | s |
|  |  | S | Average cake saturation time profile for cake\_x in  Station 1/2/3 – vector [1×length(t)] | - |
|  |  | w\_EtOH\_cake | Time profile of average ethanol mass fraction in cake for cake\_x in Station 1/2/3 – vector [1×length(t)] | - |
|  | station4 | cake\_x | t | Readings of timer reinitialized at every carousel rotation – vector – step: sampling\_time | s |
|  |  |  | S | Average cake saturation time profile for cake\_x in Station 2/3/4 – vector [1×length(t)] | - |
|  |  |  | w\_EtOH\_cake | Time profile of average ethanol mass fraction in cake for cake\_x in Station 2/3/4 – vector [1×length(t)] | - |
|  |  |  | Tg\_top | Temperature of drying air at top of cake in Station 4 – vector [1 × length(t)] | K |
| measurements |  |  | t\_meas | Readings of timer initialized at process onset - vector: step = sampling\_time | s |
|  |  |  | m\_filt\_WI101 | Readings of WI101 – vector [1 × length(t\_meas)] | kg |
|  |  |  | P\_PI102 | Readings of PI102– vector [1 × length(t\_meas)] | Pa |
|  |  |  | c\_slurry\_AI101 | Readings of AI101– vector [1 × length(t\_meas)] | kg/m3 |
|  |  |  | L\_cake\_LI101 | Readings of LI101– vector [1 × length(t\_meas)] | m |
|  |  |  | V\_slurry\_LI101 | Readings of LI101– vector [1 × length(t\_meas)] | m3 |
|  |  |  | Tg\_in\_TI101 | Readings of TI101– vector [1 × length(t\_meas)] | K |
|  |  |  | Tg\_out\_TI102 | Readings of TI102– vector [1 × length(t\_meas)] | K |
|  |  |  | Vdryer\_FI101 | Readings of FI101– vector [1 × length(t\_meas)] | L/min |
| measurements\_nf | *same structure of* measurements | | | | |
| disturbances |  |  | resistances | Vector [n\_cycles × 4] – element (*i*, *j*) = resistance of mesh in position *j* during processing cycle *i*, for *i* = 1, 2, …, n\_cycles (= total number of simulated cycles) | 1/m |
|  |  |  | c\_slurry | Multiplicative coefficients to nominal slurry concentration – vector [1 × n\_cycles] | - |
|  |  |  | V\_slurry | Multiplicative coefficients to current fed slurry set-point – vector [1 × n\_cycles] | - |
|  |  |  | E | Multiplicative coefficients to nominal cake porosity – vector [1 × n\_cycles] | - |
|  |  |  | hM | Multiplicative coefficients to nominal mass transfer parameter – vector [1 × n\_cycles] | - |
|  |  |  | hT | Multiplicative coefficient to nominal heat transfer parameter – vector [1 × n\_cycles] | - |
| operating\_vars |  |  | t\_vector | Readings of timer initialized at process onset – vector: step = control\_interval | s |
|  |  |  | dP\_vector | Set-point of pressure drop – vector [1 × length(t\_vector)] | Pa |
|  |  |  | Tin\_drying\_vector | Set-point of drying gas temperature – vector [1 × length(t\_vector)] | K |
|  |  |  | n\_cycle\_vector | Cycles counter - vector | - |
|  |  |  | t\_rot\_vector | Cycles duration – vector [1 × length(n\_cycle\_vector)] |  |
|  |  |  | V\_slurry\_vector | Set-point of fed slurry volume – vector [1 × length(n\_cycle\_vector)] | m3 |
| x\_estim |  |  |  |  |  |
| feed |  |  | c\_slurry\_vector | Nominal slurry concentration – vector [1 × n\_cycles] |  |
| cakes\_proc\_times |  |  |  |  |  |
| final\_content |  |  |  |  |  |
| active\_stations |  |  |  |  |  |
| settings |  |  | control\_flag | Scalar | - |
|  |  |  | disturbance\_flag | Scalar | - |
|  |  |  | control\_interval | Time interval at which control routines are called - scalar | s |
|  |  |  | sampling\_time | Sampling time for all sensors – scalar | s |
|  |  |  | total\_duration | Simulation duration – scalar | s |
|  | cryst\_out\_nom |  | conc\_slurry | Nominal slurry concentration in feed – scalar | kg/m3 |
|  |  |  | x | Crystal size distribution – particles diameters | m |
|  |  |  | CSD\_perc | Ru | - |
|  | u\_nom |  | t\_rot | Nominal cycle duration – scalar | s |
|  |  |  | V\_slurry | Nominal fed slurry volume - scalar | m3 |
|  |  |  | dP | Nominal pressure drop – scalar | Pa |
|  |  |  | Tinlet\_drying | Nominal inlet drying gas temperature | K |

## 2.4 Disturbances and operating modes

## 2.5 Setting up a control strategy

Figure 2 shows the logical order with which the scripts and functions of the simulator are called during a simulation. The colored blocks refer to the scripts/functions that can be edited by users.

The simulator is initiated by running the script run\_carousel.m, where the simulation settings can also be modified.

The functions controller\_online.m, controller\_cycle\_switch.m, estimator\_online.m, and estimator\_cycle\_switch.m can be edited for modifying the control strategy of the unit. The default implementation of the simulator consists of open-loop operation.

## 2.6 Simulator scripts and functions: summary

run\_carousel.m Script for initiating carousel simulation

run\_simulation.m Function handling carousel simulation schedule

carousel\_parameters.m Function containing simulation and model parameters

carousel\_simulator.m Function simulating carousel operation using filtration, deliquoring and drying models

estimator\_online.m Function that can be written by the user for online state/parameter estimation

controller\_online.m Function that can be written by the user, containing online control routines

estimator\_cycle\_switch.m Function that can be written by the user for state/parameter estimation routines to be executed at every carousel rotation

final\_composition.m Function executed at the end of every cycle to calculate the composition of the discharged cake, if there is any, and for storing the value of other variables contained in the simulation output

controller\_cycle\_switch.m Function that can be written by the user, containing control routines to be executed at every carousel rotation

disturbances.m Function that sets the value of the disturbances for the following cycle (e.g., filter mesh resistance, Gaussian fluctuations, …)

switch\_cycle.m Function containing carousel rotation simulation routines, such as material transfer from one port to the following one

model\_filtration.m Function simulating filtration (ODE model)

model\_deliquoring.m Function simulating deliquoring with design charts (approximate method called when cake is very small, i.e. with height below 0.3 mm)

model\_deliquoring\_grad.m Function simulating deliquoring (PDE model)

model\_drying.m Function simulating drying (PDE model)

Functions to run for starting the simulation with the desired settings:

run\_carousel.m

Functions to edit for changing the control strategy:

controller\_cycle\_switch.m

controller\_online.m

estimator\_cycle\_switch.m

estimator\_online.m

# 3 Quality-by-design and quality-by-control challenges

The simulator proposed in this study can be used for testing different control strategies within the quality-by-design and quality-by-control frameworks.

The general objective of the process is delivering dry cakes meeting the target quality, namely a residual ethanol content (critical quality attribute) below 0.5%w. This is achieved by setting suitable values for the operating variables, for a given inlet slurry concentration (critical material attribute of the process). Following the quality-by-design jargon, the critical process parameters are identified as and , since they significantly impact the residual ethanol in the discharged cake. and , instead, are identified as the control variables, which affect the product quality to a smaller extent, compared to the critical process parameters. Control routines can be implemented in the simulator for adjusting the desired values of the operating variables (set-point of the relevant controllers). The simulator also features blocks specifically dedicated to the implementation of state and parameter estimation routines. Note that, although the simulator makes the values of multiple outputs available, only the simulated measurements can be used by the developed control and estimation routines (Figure 3), as in real life carousels.

The following specific challenges are envisioned for the simulator:

1. Open-loop operation: determination of operating points delivering the target product quality and description of the design space of the unit;
2. Implementation of state estimators, soft sensors, and real time parameter estimation routines for monitoring key process variables, such as the ethanol content in the cake being dried and the resistance of the filter meshes;
3. Implementation of (model-free and model-based) control routines for automatic adjustment of the control variables (i.e., and ) to meet the target product quality in response to disturbances (in operating modes 0-2) and/or to changes in the inlet slurry concentration. Description of the design space of the unit with such closed-loop routines in place;
4. Implementation of (model-free and model-based) control routines for automatic adjustment of the critical process parameters (i.e., and ), to meet the target product quality in response to disturbances (in operating modes 0-2) and/or to changes in the inlet slurry concentration;
5. Repeating Tasks 3-4 with the additional objective of maximizing the slurry throughput, namely the amount of crystals processed by the carousel in a given timeframe, in operating modes 0-2.
6. For benchmarking purposes, we define the specific objective of implementing closed-loop control routines acting on and to maximize the cumulative mass of the cakes (meeting the target quality) discharged by the carousel in 1 hour of operation in operating modes 0-2, with equal to 250 kg/m3, fixed to 323 K, and fixed to 1E5 Pa. The additional objective of assessing the optimal for maximizing the throughput with the other conditions fixed can also be addressed.

In addition to the listed tasks, the simulator can also be used for generating data for data analytics studies, or for benchmarking fault detection, identification and diagnosis methodologies.

Despite its advanced features, the simulator has some limitations. The assumption of perfect control of the operating variables may not be met in physical carousels, especially for the inlet drying air temperature. Moreover, all simulated measurements are generated without any delay or sensor dynamics. The time needed for the physical rotation of the carousel between cycles is neglected (i.e., the carousel rotation occurs instantaneously in the simulator). Another limitation involves filter mesh fouling. In practice, the actual fouling increase from cycle to cycle depends on the current operating conditions, especially the fed slurry volume and concentration. Since not enough data were available for characterizing the fouling dependence on the operating conditions, the fouling schedule is kept constant in the simulator (Section 2.3.2), and mesh cleaning is assumed to occur instantaneously, when triggered.

Despite these limitations, *ContCarSim* is a realistic simulator that can be used for promoting the adoption of advanced control strategies in pharmaceutical manufacturing on the one hand, and to improve the operation of the novel carousel technology for continuous filtration-drying on the other hand.

# 4 Sample case study

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Sub-field** | **Sub-sub-field** | **Variable** | **Description** | **UOM** |
| states | station1 | batch\_x | t | Readings of timer reinitialized at every carousel rotation – vector – step: sampling\_time | s |
|  |  |  | m\_filt | Filtrate mass time profile for batch\_x in Station 1 – vector [1xlength(t)] | kg |
|  |  |  | S | Average cake saturation time profile for batch\_x in Station 1 – vector [1xlength(t)] | - |
|  |  |  | w\_EtOH\_cake | Time profile of average ethanol mass fraction in cake for batch\_x in Station 1 – vector [1xlength(t)] | - |
|  |  |  | L\_cake | Length of cake of batch\_x – scalar | m |
|  |  |  | c\_slurry | Slurry concentration for batch\_x - scalar | kg/m3 |
|  |  |  | V\_slurry | Loaded slurry volume for batch\_x - scalar | m3 |
|  | pos2/pos3/pos4 | batch\_x | t | Readings of timer reinitialized at every carousel rotation – vector – step: sampling\_time | s |
|  |  |  | m\_filt | Filtrate mass time profile for batch\_x in Station 2/3/4 – vector [1xlength(t)] | kg |
|  |  |  | S | Average cake saturation time profile for batch\_x in Station 2/3/4 – vector [1xlength(t)] | - |
|  |  |  | w\_EtOH\_cake | Time profile of average ethanol mass fraction in cake for batch\_x in Station 2/3/4 – vector [1xlength(t)] | - |
|  | sequence | batch\_x | filtration\_duration | Cumulative filtration duration for batch\_x in Stations 1-4 |  |
|  |  |  | deliquoring\_duration | Cumulative deliquoring duration for batch\_x in Stations 1-4 |  |
|  |  |  | drying\_duration | Cumulative drying duration for batch\_x in Stations 1-4 |  |
|  |  |  | final\_content | Ethanol content in discharged cakes [1 x number discharged cakes] |  |
| measurements |  |  | t\_meas | Readings of timer initialized at process onset - vector: step = sampling\_time | s |
|  |  |  | m\_filt\_WI101 | Readings of WI101 – vector [1xlength(t\_meas)] | kg |
|  |  |  | P\_PI102 | Readings of PI102– vector [1xlength(t\_meas)] | Pa |
|  |  |  | c\_slurry\_AI101 | Readings of AI101– vector [1xlength(t\_meas)] | kg/m3 |
|  |  |  | L\_cake\_LI101 | Readings of LI101– vector [1xlength(t\_meas)] | m |
|  |  |  | V\_slurry\_LI101 | Readings of LI101– vector [1xlength(t\_meas)] | m3 |
|  |  |  | Tg\_top\_TI101 | Readings of TI101– vector [1xlength(t\_meas)] | K |
|  |  |  | Tg\_bot\_TI102 | Readings of TI102– vector [1xlength(t\_meas)] | K |
|  |  |  | Vdryer\_FI101 | Readings of FI101– vector [1xlength(t\_meas)] | L/min |
| measurements\_nf |  |  |  |  |  |
| disturbances |  |  | resistances | Vector [n\_cycles x 4] – element (*i*, *j*) = resistance of mesh in position *j* during processing cycle *i* | 1/m |
|  |  |  | ports\_working | Vector [n\_cycles x 4] –  if port *j* processes material during cycle i, ports\_working(*i*, *j*) = 1. Otherwise, ports\_working(*i*, *j*) = 0. | - |
|  |  |  | c\_slurry | Gaussian multiplicative disturbances to nominal slurry concentration –  Vector [1 x n\_cycles] | - |
|  |  |  | V\_slurry | Gaussian multiplicative disturbances to current fed slurry set-point –  Vector [1 x n\_cycles] | - |
|  |  |  | E | Gaussian multiplicative disturbances to nominal cake porosity –  Vector [1 x n\_cycles] | - |
|  |  |  | hM | Gaussian multiplicative disturbances to nominal mass transfer coefficient –  Vector [1 x n\_cycles] | - |
|  |  |  | hT | Gaussian multiplicative disturbances to nominal heat transfer coefficient –  Vector [1 x n\_cycles] | - |
| operating\_vars |  |  | t\_vector | Readings of timer initialized at process onset – vector: step = control\_interval | s |
|  |  |  | dP\_vector | Set-point of pressure drop – vector [1 x length(t\_vector)] | Pa |
|  |  |  | Tin\_drying\_vector | Set-point of drying gas temperature – vector [1 x length(t\_vector)] | K |
|  |  |  | n\_cycle\_vector | Cycles counter - vector | - |
|  |  |  | t\_rot\_vector | Cycles duration – vector [1 x length(n\_cycle\_vector)] |  |
|  |  |  | V\_slurry\_vector | Set-point of fed slurry volume – vector [1 x length(n\_cycle\_vector)] | m3 |
| estimated\_states\_parameters  *structure defined by user in estimator\_online.m and estimator\_cycle\_switch.m* | | | | | |
| feed |  |  | c\_slurry\_vector | Nominal slurry concentration –  vector [1 x n\_cycles] |  |
| cakes\_processing\_times |  |  |  |  |  |
| final\_content |  |  |  |  |  |
| active\_stations |  |  |  |  |  |
| settings |  |  | control\_flag | Scalar | - |
|  |  |  | disturbance\_flag | Scalar | - |
|  |  |  | control\_interval | Time interval at which control routines are called - scalar | s |
|  |  |  | sampling\_time | Sampling time for all sensors – scalar | s |
|  |  |  | total\_duration | Simulation duration – scalar | s |
|  | cryst\_output\_nominal |  | conc\_slurry | Nominal slurry concentration in feed – scalar | kg/m3 |
|  |  |  | x | Crystal size distribution – particles diameters | m |
|  |  |  | CSD\_perc | Ru | - |
|  | u\_nominal |  | t\_rot | Nominal cycle duration – scalar | s |
|  |  |  | V\_slurry | Nominal fed slurry volume - scalar | m3 |
|  |  |  | dP | Nominal pressure drop – scalar | Pa |
|  |  |  | Tinlet\_drying | Nominal inlet drying gas temperature | K |