Discrete Event-based Modeling of Conveyors for Dry Bulk Material

Peter Junglas[1*], Lukas Schmedes[2]

[1]PHWT-Institut, PHWT Vechta/Diepholz, Thüringer Straße 3, 49356 Diepholz, Germany;
[*]*peter@peter-junglas.de*

[2] GRIMME Landmaschinenfabrik GmbH & Co. KG, Hunteburger Straße 32, 49401 Damme, Germany

SNE 33(3), 2023, 133-140, DOI: 10.11128/sne.33.tn.10655
Selected ASIM SST 2022 Postconf. Publication: 2023-08-01;
Received Rev. Improved: 2023-09-21; Accepted: 2023-09-25
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna
ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. The simulation of transport processes, though inherently continuous, is often done in a discrete-event simulation environment. In the case of conveyors for dry bulk material, this can lead to modeling difficulties, especially regarding the coupling of two conveyors with different velocities. We will present a modeling approach solving such problems, describe an implementation in SimEvents and present results of systematic tests.

Introduction

In the simulation of production and logistic processes, the modeling of materials handling is of paramount importance. Though the detailed description of a transport process uses continuous functions of time, such as positions or mass flows, in the context of a complex simulation it is often simplified and modeled using only discrete events. But this reduction of complexity can lead to problems, because:

> It is often important to model such entity transfer accurately since studies have shown that delays and inefficiencies in operations might be caused more by the need just to move things around rather than in actually doing the work. [1, p.345]

A simple conveyor belt moving discrete unit loads with constant velocity generally can be modeled easily enough. But building an adequate model for the transport of dry bulk material with a wide range of granularity and changing velocities of one belt or between belts is much more difficult.

Modifying the belt velocity is a standard method to adapt to a varying input mass flow. This can be used to utilise the full capacity of the conveyor at a lower speed, which will often decrease the power consumption [2], or to speed up in order to shorten the transportation time. On the other hand, when transporting damageable goods like apples or potatoes, one could try to slow down the conveyor to guarantee a high quality of the goods.

In the following we will describe a discrete-event model of a conveyor for dry bulk material, which has a control input to change the velocity. A special focus will be on the coupling of conveyors running with different velocities, since this leads to modeling problems in a discrete environment. Finally the model will be implemented in SimEvents from Mathworks [3] and tested systematically.

The acceleration or deceleration of a highly loaded conveyor creates considerable tension in the belt, leading to local stretching or even breaking of the belt [4]. In this study we will neglect this effect and treat the belt as a rigid body with the same velocity everywhere.

1 Modeling of a Single Conveyor

A discrete-event model of a conveyor has to implement the delays of the incoming entities given by the conveyor length l and the velocity v. In addition it has to store the positions of all entities at the current (discrete) time in order to cope with entities of varying size l_E or with a time-dependent velocity (cf. Figure 1).

Such a component exists in many commercial discrete simulation environments such as Arena [1], SimEvents [3] or PlantSimulation [5]. The length of the entities can either be defined as a fixed parameter or as an entity-specific attribute. The various programs have different additional features like a minimal distance between entities, accumulation of entities in case

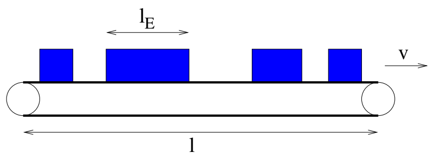

Figure 1: Simple conveyor model.

of blocking or incorporating acceleration phases at the beginning – all of which will not be considered in the following. Another difference is the exact procedure, how an entity enters (or leaves) the conveyor. For concreteness we will define that a (right-going) entity starts at position $x = 0$ with its left edge coinciding with the left edge of the conveyor, and leaves at position $x = l$, when its left edge coincides with the right edge of the conveyor.

For the transportation of dry bulk material the situation is a bit more complicated, because there are no easily identifiable entities and the transport process is inherently continuous: Due to a usually non-uniform production process and the inhomogeneity of the material the conveyor is filled with an incoming mass flow $\dot{m}_{in}(t)$, which leads to a line load $\lambda := \frac{\partial m}{\partial x}$ given by

$$\lambda(t,x) = \frac{1}{v}\dot{m}_{in}(t - \frac{x}{v}) \tag{1}$$

for a constant velocity v (cf. Figure 2).

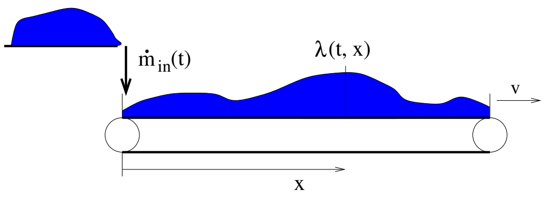

Figure 2: Conveyor model for bulk dry material.

The granularity of dry bulk material varies widely in practical applications, ranging from almost microscopic particles (powder) over small or medium sized particles (rice, apples) to large lumps (ore). The modeling methods used vary accordingly. Two extreme methods are described and compared in [6]: the microscopic description, where the movement of each particle is described separately, and a macroscopic representation, using a mass density and differential equations describing the conservation laws.

Both methods are computationally intensive, therefore in [7] a “mesoscopic” approach has been utilised, which is well suited for medium sized granularity and can be easily incorporated into standard discrete modeling environments. Basically, the continuous line load is replaced by discrete entities E_i with mass m_i given by

$$m_i = \int_{(i-1)\Delta t}^{i\Delta t} \dot{m}_{in}(t)\,dt, \quad i \in \mathbb{N}^+$$

for an arbitrary fixed time interval Δt. Using (1) one gets for constant velocity v:

$$\begin{aligned} m_i &= v\int_{(i-1)\Delta t}^{i\Delta t} \lambda(i\Delta t, v(i\Delta t - t))\,dt \\ &= \int_0^{v\Delta t} \lambda(i\Delta t, x)\,dx \end{aligned}$$

This shows that the conveyor is divided into compartments of equal length $l_E = v\Delta t$. For simplicity, one often chooses $l_E = l/N$ for $N \in \mathbb{N}^+$, so that entity E_i enters the conveyor at time $i\Delta t$, i. e. when its compartment is filled, and leaves at $(N+i)\Delta t$.

An alternative approach could be to use compartments of equal mass instead of equal length. This would lead to a more complex timing of events, which makes the interpretation of results more involved. Furthermore, some simulation environments (e. g. Arena) use conveyor components with a fixed cell size, which would make the implementation of this approach quite ugly.

2 Coupling of Conveyors with Different Velocities

We will now analyse how one can combine two single conveyors with lengths l_1, l_2 and velocities v_1, v_2. A mathematical description of the continuous process with additional input and output reservoirs at the end of each conveyor has been given in [8]. Considering only the case of constant (but different) velocities, one has:

$$\begin{aligned} \lambda_1(t,x) &= \frac{1}{v_1}\dot{m}_{in,1}(t - \frac{x}{v_1}) \\ \lambda_2(t,x) &= \frac{1}{v_2}\dot{m}_{in,2}(t - \frac{x}{v_2}) \end{aligned}$$

Connecting the conveyors directly, the output of conveyor 1 is the input of conveyor 2, therefore:

$$\begin{aligned} \dot{m}_{in,2}(t) &= v_1 \lambda_1(t, l_1) \\ \Rightarrow \qquad \lambda_2(t,x) &= \frac{v_1}{v_2}\lambda_1(t - \frac{x}{v_2}, l_1) \end{aligned}$$

For a continuous model one simply changes the line load λ_2 by the factor v_1/v_2. This happens automatically in real life, when the bulk material gets thinned out or condensed on the second conveyor – as long as the capacity of the second belt is not exceeded.

But our discrete model runs into problems with the timing of the entities: Conveyor 1 sends entities at time intervals $\Delta t_1 = l_{E,1}/v_1$ to the entrance of conveyor 2, which in turn delivers entities at its output at generally different time intervals $\Delta t_2 = l_{E,2}/v_2$. Therefore one cannot maintain the idea of an entity defined as a fixed set of particles with given mass. Instead one identifies an entity with the content of a given compartment on a conveyor. Such a compartment is created and filled at the entrance of a conveyor, and emptied and destroyed at its exit.

The remaining task is now to compute the mass of such a newly created entity. According to the ratio

$$k := \frac{\Delta t_2}{\Delta t_1} = \frac{l_{E,2}}{l_{E,1}} \cdot \frac{v_1}{v_2}$$

one needs different strategies, how to cope with this problem. Such strategies should fulfil two requirements:

- mass conservation, i. e. incoming and outgoing masses should balance on a short time scale,
- homogeneity, i. e. the output mass distribution should closely follow the input mass values. For a constant incoming distribution this means, that the outgoing values shouldn't vary much.

If k is integer, one simply adds up the masses of k incoming entities to create an outgoing one, whereas if $1/k$ is integer, one distributes the mass of one incoming entity among $1/k$ outgoing entities. In all other cases one has to account for the unbalanced timing of input and output entities. Since the problem appears only at the connection of the two conveyors, we can concentrate on the second conveyor with its incoming values $m_{in,i}$ at times $i\Delta t_1$ and the corresponding outgoing values $m_{out,j}$ at times $j\Delta t_2$, disregarding the delay time of the second conveyor.

If $k > 1$ one can apply a simple collection strategy using a virtual bin, which accumulates all incoming masses into m_{acc}. A new output entity then empties the bin and gets the total accumulated mass. Table 1 shows how this works in an example with equal entity lengths $l_{E,1} = l_{E,2} = 1\,\text{m}$, velocities $v_1 = 2.5\,\text{m/s}$, $v_2 = 1\,\text{m/s}$ and constant incoming masses $m_i = 1\,\text{kg}$.

i	j	t	m_{in}	m_{acc}	m_{out}
1	-	0.4	1	1	-
2	-	0.8	1	2	-
-	1	1.0	-	0	2
3	-	1.2	1	1	-
4	-	1.6	1	2	-
5	2	2.0	1	0	3
6	-	2.4	1	1	-

Table 1: Times and masses for example 1 ($k = 2.5$).

For $k < 1$ one has to use a partition strategy instead. The following strategy "A" defines the mass of a partition as

$$m_p = k\, m_{in}$$

each time a new entity enters, and sets outgoing entities accordingly. This simple scheme leads to a problem due to the timing, as can be seen in Table 2, which uses $v_1 = 1\,\text{m/s}$, $v_2 = 1/0.35\,\text{m/s}$: At $t = 1.75$ there is not enough mass available for the outgoing entity E_5. But this can be cured easily by setting

$$m_{out} = \min(m_p, m_{acc})$$

Unfortunately strategy A has a serious drawback, as Table 3 shows using $v_1 = 1\,\text{m/s}$, $v_2 = 1/0.8\,\text{m/s}$: Though the mean ratio of input entities to output entities is 0.8, for a while the actual ratio is 1. Therefore the accumulated mass m_{acc}, which is just the difference between total input and total output mass, grows.

i	j	t	m_{in}	m_p	m_{acc}	m_{out}
1	-	1.0	1	0.35	1	-
-	3	1.05	-		0.65	0.35
-	4	1.4	-		0.3	0.35
-	5	1.75	-		0	0.30
2	-	2.0	1	0.35	1	-

Table 2: Times and masses for example 2 ($k = 0.35$).

This is in conflict with the primary goal of mass conservation on a short time scale. Even worse: When the following input entities are empty (i. e. $m_{in} = 0$), m_p is set to 0 and the accumulated mass stays in the internal bin.

i	j	t	m_in	m_p	m_acc	m_out
1	-	1.0	1	0.8	1	-
-	2	1.6	-		0.2	0.8
2	-	2.0	1	0.8	1.2	-
-	3	2.4	-		0.4	0.8
3	-	3.0	1	0.8	1.4	-
-	4	3.2	-		0.6	0.8

Table 3: Times and masses for example 3 ($k = 0.8$), strategy A.

Strategy "B" tries to solve this problem by changing the value of the partition mass to

$$m_p = k\, m_{acc},$$

where the value is only computed, when an input entity arrives. This will distribute the surplus value of m_{acc} among the next outgoing entities, thereby reducing the total mass imbalance, as can be seen in Table 4 for the values of example 3.

i	j	t	m_in	m_p	m_acc	m_out
1	-	1.0	1	0.800	1.000	-
-	2	1.6	-		0.200	0.800
2	-	2.0	1	0.960	1.200	-
-	3	2.4	-		0.240	0.960
3	-	3.0	1	0.992	1.240	-
-	4	3.2	-		0.248	0.992

Table 4: Times and masses for example 3 ($k = 0.8$), strategy B.

We will finally provide a mathematical description of the distribution strategy B to clarify possible open points and to guide the implementation. Starting point are the two positive time intervals Δt_1, $\Delta t_2 = k\,\Delta t_1$ between arrival or departure of entities at the virtual connecting bin and the positive end time t_{end} of the simulation. We now define the sets

$$\begin{aligned} T_{in} &= \{i\Delta t_1 \,|\, i \in \mathbb{N}^+\} \cap [0, t_{end}] \\ T_{out} &= \{j\Delta t_2 \,|\, j \in \mathbb{N}^+\} \cap [0, t_{end}] \end{aligned}$$

The function $m_{in}(t)$ is given for $t \in T_{in}$ (by a production process) and constant elsewhere.

The functions m_{acc}, m_p and m_{out} will be defined on $T_{in} \cup T_{out}$, they are constant elsewhere. For simplicity we denote

$$f(t^-) := f(t - \varepsilon) \quad (\varepsilon > 0 \text{ sufficiently small}),$$

where "sufficiently small" means "smaller than the size of any open time interval from $T_{in} \cup T_{out}$". We now start with

$$m_{acc}(0) = 0\, kg$$

and define:

$$m_p(t) = \begin{cases} k\,(m_{acc}(t^-) + m_{in}(t)) & | \, t \in T_{in} \\ \text{const.} & | \, \text{otherwise} \end{cases}$$

$$m_{out}(t) = \begin{cases} \min(m_p(t), m_{acc}(t^-)) & | \, t \in T_{out} \setminus T_{in} \\ \min(m_p(t), m_{acc}(t^-) + m_{in}(t)) & | \, t \in T_{in} \cap T_{out} \\ \text{const.} & | \, \text{otherwise} \end{cases}$$

$$m_{acc}(t) = \begin{cases} m_{acc}(t^-) + m_{in}(t) & | \, t \in T_{in} \setminus T_{out} \\ m_{acc}(t^-) - m_{out}(t) & | \, t \in T_{out} \setminus T_{in} \\ m_{acc}(t^-) + m_{in}(t) - m_{out}(t) & | \, t \in T_{in} \cap T_{out} \\ \text{const.} & | \, \text{otherwise} \end{cases}$$

One easily checks that these definitions reproduce the collection strategy and partition strategy B. Strategy A is a bit simpler and can be easily defined in a similar way.

3 Implementation in SimEvents

SimEvents [9] is a blockset for the Simulink environment from Mathworks [10] that enables discrete event modeling. It uses the transaction-based approach, which describes entities that are handled by fixed components. It contains the usual components like an entity generator, a server, a queue and several routing blocks. As stated above, a basic `Conveyor System` component is available that transports discrete entities of given length. Many components include so-called "action"-functions, which are called at the entry or exit of an entity, and can be defined using Simulink function blocks.

The conveyor for dry bulk material (cf. Figure 3) is defined as a component with an input and output port for the entities, inputs for the incoming and outgoing

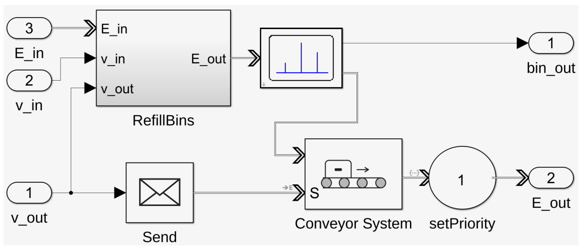

Figure 3: Conveyor model.

velocities and an additional output to display the entities leaving the internal bin. The length l_E of the compartments and the total length l are provided as parameters. Incoming and outgoing entities have attributes describing their length and their mass. The block uses the predefined `C onveyor System` and a component `RefillBins` that handles the adaptation of the different velocities.

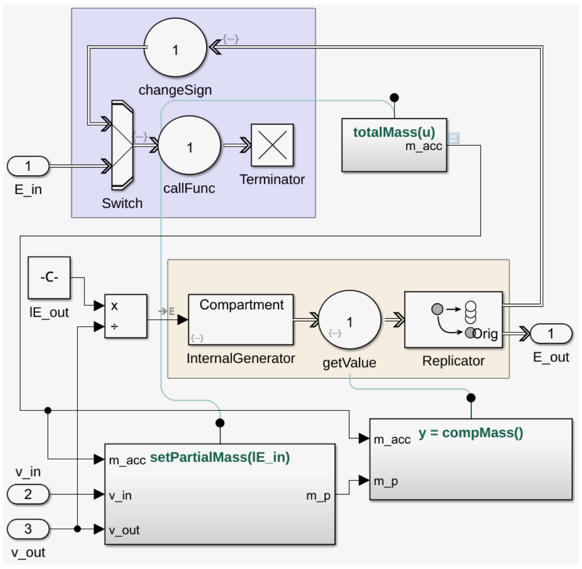

Figure 4: Implementation of the `RefillBins` component.

The central block `RefillBins` (cf. Figure 4) implements the formulae described in section 2 that define strategy B. Incoming entities are routed through a server, which calls the Simulink functions `totalMass` to compute m_{acc} and `setPartialMass` to compute m_p, and destroyed afterwards. An internal generator creates new entities at times in T_{out} and sends them to a server that sets the mass attribute to m_{out}, which is computed with the function `compMass`. Before an entity leaves the block, a copy is created and sent back to the input server, so that its mass can be subtracted from m_{acc}. The alternative strategy A can be implemented easily in an analogous way.

As usual for transaction-based modeling, one has to make sure that concurrent events are handled in the correct order to make things work. If $t \in T_{in} \cap T_{out}$, this means that the incoming entity has to be processed before the internally created one to guarantee the correct computation of m_p and m_{out}. For this purpose entities enter the conveyor with a high priority (low value), while the internal generator creates entities with low priority, which is raised, when an entity leaves the conveyor.

A more subtle timing problem has lead to the inclusion of the server `getValue` behind the internal generator: In principle the call of the function `compMass` could have been done immediately inside the generator. But then the order of the mass computation and the processing of a concurrent incoming entity are not defined! The priority only affects events and messages, not internal function calls.

4 Test Results

To compare the performance of the strategies A and B in detail, a set of tests have been carried out that concentrate on two key figures: the mean value over time $\overline{m_{acc}}$ of the internally accumulated mass, which shows the short-time mass conservation, and the standard deviation σ_{out} of the output mass, which measures the homogeneity of the outgoing mass distribution.

All tests use constant entity lengths $l_{E,1} = l_{E,2} = 1\,\mathrm{m}$ and outgoing velocity $v_2 = 1\,\mathrm{m/s}$. The input velocity is given as $v_1 = k v_2$, where different values of k and different input mass distributions will be analysed. All results are compiled in Table 5 and are referenced by their number in the following.

The first group (1 – 9) consists of tests with constant input mass $m_i = 1\,\mathrm{kg}$ and varying k. For k or $1/k$ integer, optimal procedures have been given in section 2, which lead to a constant output mass, i. e. $\sigma = 0$. For these cases the average value of m_{acc} can easily be computed to be

$$\begin{aligned} \overline{m_{acc}} &= \frac{n-1}{2n} & \big|\, n \equiv 1/k \in \mathbb{N}^+ \\ \overline{m_{acc}} &= \frac{n-1}{2} & \big|\, n \equiv k \in \mathbb{N}^+ \end{aligned}$$

Test results 1 – 4 show that both implementations reproduce these values, minor differences are due to a short initial period.

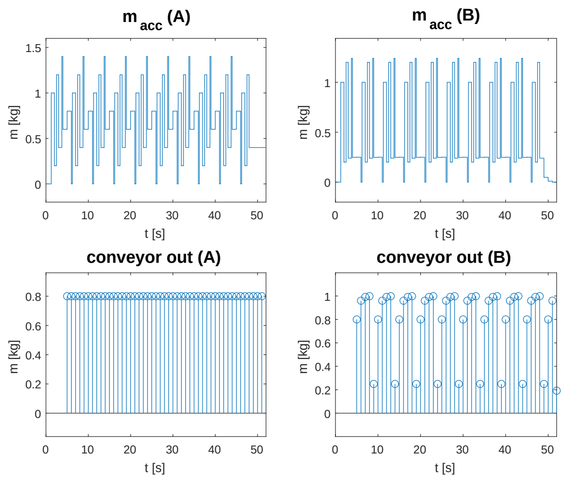

Figure 5: Test result 6, constant input, k = 0.8.

Results for other values of k are shown in tests 5 – 9, among them the examples from section 2. The plots in Figure 5 display the function $m_{acc}(t)$ and the conveyor output over time for both strategies, they reproduce the results for $k = 0.8$ from Tables 3 and 4. The figures from Table 5, no. 6, show that the mass balance of strategy B is better by a factor of 1.5 than that of strategy A.

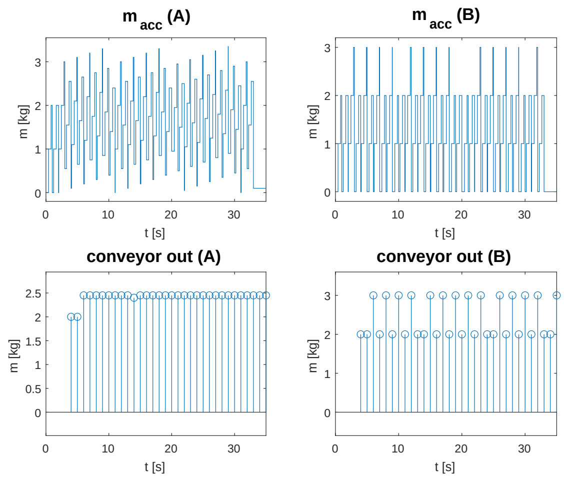

Figure 6: Test result 8, constant input, k = 2.45.

Very instructive is the comparison of tests 8 and 9: Changing k from 2.5 to 2.45 leads to a much worse local mass balance, especially for strategy A.

The reason for this behaviour can be seen in Figure 6: The accumulated mass rises slowly over fast cycles of 2 s, but is reset with a longer period of 20 s. A look at Table 1 shows that for $k = 2.5$ a much shorter period of 5 s appears, so that m_{acc} can't grow as much. The length of the period is given by the representation $k = p/q$ with coprime natural numbers p, q:

$$k = \frac{p}{q} = \frac{\Delta t_2}{\Delta t_1} \quad \Rightarrow \quad q\Delta t_2 = p\Delta t_1,$$

where $\Delta t_2 = 1\,\mathrm{s}$ in all our tests. A rational k with a large denominator therefore leads to a long period, which can possibly produce a long time accumulation and a bad mass balance. The problem is less severe for strategy B, since it gets rid of short time accumulations as fast as possible.

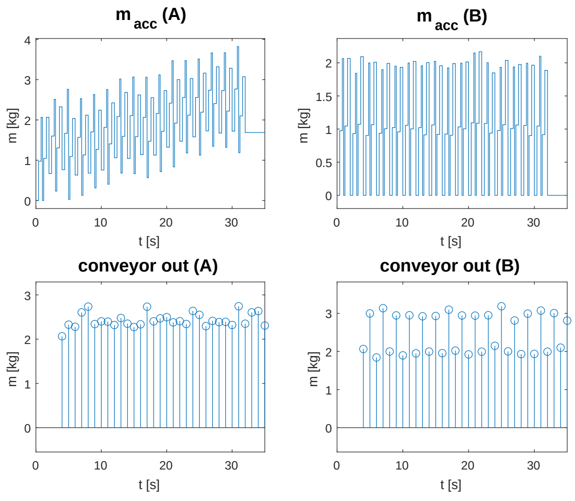

Figure 7: Test result 14, uniform input, k = 2.5.

In real applications the input flow is usually not constant, at least it fluctuates due to the granularity of the material. To model this, the next tests (10 – 14) take mass input values $m_i(t)$ using a uniform distribution on the interval [0.9, 1.1] kg, which has a standard deviation of $\sigma_{in} = 0.058$ kg.

The corresponding results in Table 5 are generally similar to the previous ones, but the standard deviations seem to be too small, they are sometimes smaller than σ_{in}. This is due to two effects: Firstly, the mean value of the output mass is not 1, but k, and σ_{out} has to be scaled accordingly. Secondly, the internal accumulation process smoothes the incoming values, thereby reducing the standard deviation.

A striking result is the mass balance of strategy A in test 14 (k = 2.5), which is much larger than expected. Figure 7 shows that the accumulation of rest masses, which was limited before due to the periodic behaviour, now grows apparently unbounded.

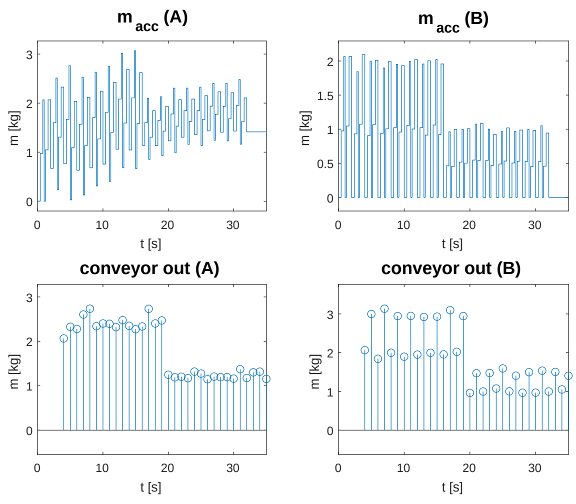

Figure 8: Test result 19, falling input, k = 2.5.

For the last tests a macroscopic change has been added to the small scale fluctuations: In tests 15 – 19 the mean value of the input mass is reduced by a factor two in the middle of the measurement, in tests 20 – 24 it is doubled.

If one takes the changing mass scale into account, the results are similar to the last ones. It is interesting to note, what happens to the previous problematic case $k = 2.5$: As can be seen in Figure 8 the accumulation of "residual mass" continues, even if the mean value drops. The same happens, if the mean value rises (cf. Table 5, no. 24).

5 Conclusion

We have presented a simple discrete event-based model of a conveyor system for the transport of dry bulk material, which allows for the coupling of conveyors with different velocities. Two strategies have been compared to cope with the timing problems, where strategy B is much better, if the short time mass balance is of highest concern, while strategy A provides a better homogeneity of the outgoing masses.

No.	k	$\overline{m_{acc}}(A)$ [kg]	$\overline{m_{acc}}(B)$ [kg]	$\sigma_{out}(A)$ [kg]	$\sigma_{out}(B)$ [kg]
1	0.33	0.3300	0.3300	0.0000	0.0000
2	0.14	0.4243	0.4243	0.0000	0.0000
3	3.00	1.0000	1.0000	0.0000	0.0000
4	7.00	3.0000	3.0000	0.0000	0.0000
5	0.35	0.5315	0.4460	0.0145	0.0394
6	0.80	0.6644	0.4555	0.0000	0.2930
7	1.20	0.8429	0.5000	0.0672	0.3966
8	2.45	1.5315	1.1300	0.1106	0.5040
9	2.50	1.1714	0.9143	0.0884	0.5080
10	0.35	0.5331	0.4440	0.0241	0.0386
11	0.80	0.6825	0.4499	0.0455	0.2933
12	1.20	0.8351	0.4933	0.0867	0.3948
13	2.45	1.3826	1.1141	0.1690	0.4864
14	2.50	1.8598	0.9123	0.1552	0.5136
15	0.35	0.4111	0.3237	0.0875	0.0929
16	0.80	0.6621	0.3391	0.2048	0.3075
17	1.20	0.6865	0.3629	0.2910	0.4185
18	2.45	1.1876	0.8426	0.5983	0.6979
19	2.50	1.5818	0.6838	0.6132	0.7500
20	0.35	0.8212	0.6846	0.1764	0.1809
21	0.80	0.9939	0.6715	0.4006	0.5983
22	1.20	1.3305	0.7239	0.6296	0.9080
23	2.45	2.0568	1.6572	1.2454	1.4854
24	2.50	2.4205	1.3695	1.2898	1.5003

Table 5: Test results comparing mass conservation and homogeneity of both strategies.

While discretisation of continuous systems is important to reduce computation times drastically, it creates problems of its own. To solve them, a precise mathematical description is of uttermost importance, not only to precisely define the model, but also to guide and thereby simplify the implementation process. A typical implementation problem, which had to be solved here, was to ensure the correct ordering of concurrent events. While using priorities is a standard way to cope with it, one had to dig deeply into internal features of SimEvents to come up with a final solution. Since such details vary between different simulation environments [11], a precise (mathematical!) definition of the exact behaviour of SimEvents would have been helpful.

Though the model has shown its principle validity in a series of tests, the real proof of its usefulness would be seen in the integration with a controller.

The coupling with a continuous controller should work in principle, but for several reasons – practical as well as theoretical –, a discrete controller with a finite set of velocities would be more adequate in a lot of applications [12, 13].

Whether the simple strategies proposed here are useful in such a context, or whether one needs more complex strategies, which are adapted to the controller algorithm, is a question for future research.

Acknowledgement

The authors like to thank GRIMME Landmaschinenfabrik GmbH & Co. KG for the permission to publish results of internal research.

References

[1] Kelton WD, Sadowski R, N. Zupick. *Simulation with Arena*. New York: McGraw-Hill, 6th ed. 2015.

[2] Hiltermann J, Lodewijks G, Schott DL, Rijsenbrij J, Dekkers J, Pang Y. A methodology to predict power savings of troughed belt conveyors by speed control. *Particulate science and technology*. 2011;29(1):14–27.

[3] The MathWorks. *SimEvents: Model and simulate discrete-event systems*. `www.mathworks.com/products/simevents.html`.

[4] He D, Pang Y, Lodewijks G. Belt conveyor dynamics in transient operation for speed control. *International Journal of Civil, Environmental, Structural, Construction and Architectural Engineering*. 2016; 10(7):828–833.

[5] Bangsow S. *Tecnomatix Plant Simulation - Modeling and Programming by Means of Examples*. Cham: Springer, 2nd ed. 2020.

[6] Göttlich S, Hoher S, Schindler P, Schleper V, Verl A. Modeling, simulation and validation of material flow on conveyor belts. *Applied mathematical modelling*. 2014; 38(13):3295–3313.

[7] Fioroni MM, Franzese LAG, Zanin CE, Furia J, de Toledo Perfetti L, Leonardo D, da Silva NL. Simulation of continuous behavior using discrete tools: Ore conveyor transport. In: *2007 Winter Simulation Conference*. IEEE. 2007; pp. 1655–1662.

[8] Pihnastyi O, Kozhevnikov G, Khodusov V. Conveyor model with input and output accumulating bunker. In: *2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT)*. IEEE. 2020; pp. 253–258.

[9] Li W, Mani R, Mosterman PJ. Extensible discrete-event simulation framework in SimEvents. In: *Proc. 2016 Winter Simulation Conference*. Arlington: IEEE. 2016; pp. 943–954.

[10] The MathWorks. *Simulink: Simulation and Model-Based Design*. `https://www.mathworks.com/products/simulink/`.

[11] Schriber TJ, Brunner DT, Smith JS. How discrete-event simulation software works and why it matters. In: *Proceedings of the 2012 Winter Simulation Conference (WSC)*. IEEE. 2012; pp. 1–15.

[12] Pang Y, Lodewijks G, Schott DL. Fuzzy Controlled Energy Saving Solution for Large-Scale Belt Conveying Systems. In: *Energy, Environment and Sustainable Development*, vol. 260 of *Applied Mechanics and Materials*. Trans Tech Publications Ltd. 2013; pp. 59–64.

[13] Reutov AA. Simulation of load traffic and steeped speed control of conveyor. In: *IOP Conference Series: Earth and Environmental Science*, vol. 87. IOP Publishing. 2017; p. 082041.