Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

Throughput optimization of automated flats sorting machines

A.N. Tarău * **B. De Schutter** *,** **J. Hellendoorn** *

* *Delft Center for Systems and Control*
Delft University of Technology, Mekelweg 2, 2628 CD Delft, The Netherlands {a.n.tarau, j.hellendoorn}@tudelft.nl
** *Marine and Transport Technology Department*
Delft University of Technology, The Netherlands, b@deschutter.info

Abstract: Large letters, journals, magazines, plastic wrapped mail items of A4 size are called flats. In order to be able to handle the large volumes of flats that have to be processed state-of-the-art post sorting centers are equipped with dedicated flats sorting machines. The throughput of a flats sorting machine is crucial when dealing with a continually increasing number of items to be sorted in a certain time. But, the throughput is limited by the mechanical constraints. We propose to optimize the efficiency of this sorting system by making several design changes and implementing advanced model-based control methods such as optimal control and model predictive control. In this paper we determine an event-based model of the flats sorting system. The considered control methods are compared for several scenarios. Results indicate that using the proposed approaches the throughput can be increased with up to 52.62%, the computation of the optimal velocity being performed in real-time.

1. INTRODUCTION

During the last decades the volume of magazines, catalogs, plastic wrapped mail items that have to be processed in a certain period by post sorting centers has increased considerably. Nowadays, the focus is on quality, reliability, and throughput maximization. The throughput of a post sorting machine is defined as the number of sorted mail items divided by the time needed to sort them, being expressed as e.g. number of items/hour. The throughput is limited by the mechanical constraints and also by the performance of the reading devices. In this paper we consider large letters (A4 size envelopes) with more or less information printed on them (sender and destination address, advertisement messages, stamps and mail class service information). Such mail items are named flats. Small parcels with a maximum thickness of about 80 mm are also considered flats.

The procedure performed by a flats sorting machine consists of two processes: preparing the flats and sorting them. During the preparation phase, the flats are faced in the same way, and the stamp used for postage is voided. Finally, the address and postal code are located and the necessary information is extracted and printed on the flat in form of a bar code. Conveyor systems transport the flats during their preparation with constant speed. When the mail item leaves the preparation phase, it is fed into a transport box (cassette) of the sorting phase by the feeding device, as illustrated in Figure 1. The transport box carries the flat and deposits it into a destination bin according to the bar code. The flats for which the necessary information could not be extracted, are stored in a special bin. This is how, currently, most of the flats sorting machines are working.

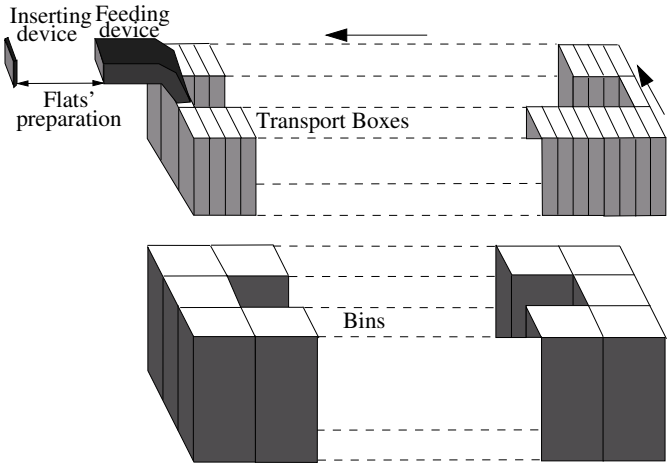

Fig. 1. Sorting part of a flats sorting machine.

The main control problems of this system consist of setting the inserting rate of the sorting machine see e.g. Lohmann [1996, 1997], positioning of the transport box when inserting the flat, and synchronization of transport boxes and bins when dropping a flat in its corresponding destination bin. At a higher level of control an important problem is how to allocate the destinations to the bins.

We investigate approaches to increase the throughput of the flats sorting machine. This can be achieved first by making minor design changes such as adding a second feeding device and also by maneuvering the bin system. Afterwards, advanced model-based control methods have to be implemented in order to assure the optimal maneuvers. A fast event-based model is determined. The control approaches considered in this paper are optimal control and model predictive control.

The paper is organized as follows. In Section 2, the proposed new set-up of the flats sorting machine is described. The simplifying assumptions and the continuous-time event-driven model to be used are presented in Section 3. In Section 4, several control approaches are proposed for setting the velocity of the system transporting the bins. Their analysis and comparison are given in Section 5. Finally, in Section 6, conclusions are drawn and possible future directions are presented.

2. AUTOMATED FLATS PROCESSING

In order to increase the throughput of the flats sorting machine, we propose a new set-up illustrated in Figure 2.

The preparation of the flats is identical to the one described in Section 1. But, in order to simplify the previous explanation, instead of the inserting device and the flats' preparation phase, a buffer of flats the codes of which are known in advance is further considered.

The plant is augmented by adding a second feeding device which also increases the throughput. However, by increasing the number of feeders only, one does not necessarily obtain the maximum possible throughput. Therefore, in the new set-up, the bottom system transporting the bins is also able to move to the left, to the right, or not at all. The top system transporting the boxes moves as usual, with a constant speed. The reason for this is to increase the number of empty transport cassettes and, hence, the availability of the transport boxes.

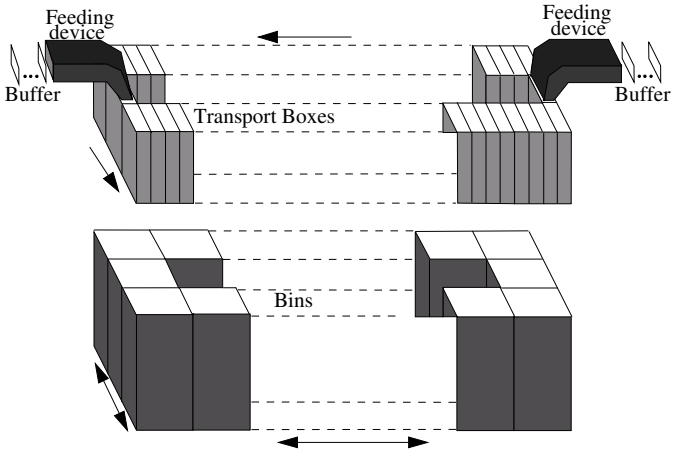

Fig. 2. New set-up for the flats sorting machine.

Consequently, a new control problem arises: continually adjusting the speed of the bottom system, so that the throughput is maximized.

One might ask why not increasing the throughput by adding more extra feeding devices. If more feeders are added to the sorting system the costs and the structural complexity of the system increase considerably. As a consequence, the gains do not outweigh the costs of the third or fourth feeder.

3. ASSUMPTIONS AND CHOSEN MODEL

We consider the simplified process depicted in Figure 2. Two FIFO (First In First Out) buffers of flats are fed into the system.

The required assumptions for this model are the following:

(1) the top system moves with a constant speed.
(2) the speed of the bottom system is piecewise constant.
(3) the flats sorting machine has two feeders positioned symmetrically.
(4) each feeding device has a finite buffer, with codes that are known in advance. This allows the computation in advance of the optimal speed and its continuous adjustment.
(5) when using a flats sorting machine with two feeders, we consider the stream of codes $S = [s(1)\ s(2)\ \cdots\ s(f)]^{\mathrm{T}}$ where f is its length, split in two new streams $S_1 = [s(1)\ s(2)\ \cdots\ s(l)]^{\mathrm{T}}$ and $S_2 = [s(l+1)\ s(l+2)\ \cdots\ s(f)]^{\mathrm{T}}$ with $l = \lfloor \frac{f}{2} \rfloor$, where $\lfloor x \rfloor$ denotes the largest integer less than or equal to x.
(6) both buffers have the same maximum capacity $b_{\max}$.
(7) the insertion of the buffer's next flat is performed in a negligible time span, when an empty box is in front of the feeding device.
(8) a full bin is replaced with a new one in a negligible time span.
(9) the dropping process is performed in a negligible time span, when a box that contains a flat for destination X is positioned above the bin with the same code.

There are four events that can occur:

- inserting a new flat into the sorting section of the system using the first feeding device.
- inserting a new flat using the second feeding device.
- dropping the flats that meet the corresponding bin.
- updating the speed of the bottom system.

The model of the flats sorting system is an event-driven system consisting of a continuous part, movement of the transport boxes and bins, and of the discrete events listed above. The following situation has been assumed: given a velocity sequence $V = [v(0)\ v(1)\ \cdots\ v(N)]^{\mathrm{T}}$ and the sequence of time interval lengths $\Delta = [\delta(0)\ \delta(1)\ \cdots\ \delta(N)]^{\mathrm{T}}$, on each time interval $[t_i, t_{i+1})$, $i = 0, 1, \cdots N$, with $t_{i+1} = t_i + \delta(i)$ and t_0 the initial time, the velocity of the bottom system equals $v(i)$ as illustrated in Figure 3.

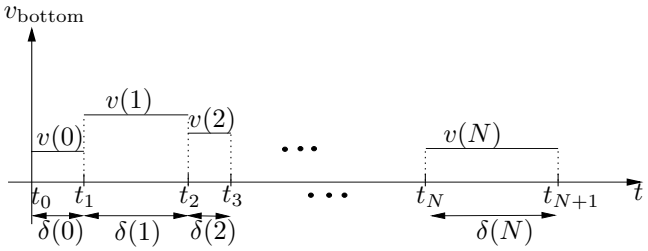

Fig. 3. Speed evolution of the bin system.

The model is given by the algorithm below, where the feeding devices are denoted by F1 and F2.

Algorithm 1. Flats sorting

1: $i \leftarrow 0$
2: $\Delta \leftarrow [\delta(0)\ \delta(1)\ \cdots\ \delta(N)]^{\mathrm{T}}$
3: **while** there are flats to be sorted **do**
4: $t_{\mathrm{F1}} \leftarrow$ time that will pass until the first empty box arrives in front of F1

```
5:    t_F2 ← time that will pass until the first empty
            box arrives in front of F2
6:    t_d ← time that will pass until the first next
            dropping event
7:    t_min ← min(t_F1, t_F2, t_d, δ(i))
8:    update the state of the system
9:    δ(i) ← δ(i) − t_min
10:   if δ(i) = 0 then
11:     update the speed of the bottom system
12:     i ← i + 1
13:   end if
14: end while
```

Hence, while there are still flats to be sorted and the machine is not stopped, the following computations are performed. The time intervals until the first feeding event using F1, feeding using F2, dropping, and updating the speed of the bottom system would occur are determined. The minimum of these time intervals, $t_{\min}$, is chosen and the state of the system is updated i.e. the position of the bottom system is shifted with $v_{\text{bottom,current}} \cdot t_{\min}$, and correspondingly, the position of the top system is shifted with $v_{\text{top}} \cdot t_{\min}$. The variables $v_{\text{bottom,current}}$ and v_{top} represent the current velocity of the bottom, and the velocity of the top system respectively. The actions are taken accordingly to the minimum time computed. More than one action can be performed in the same time and also, more than one flat can be dropped simultaneously. The length $\delta(i)$ of the time interval is decreased with $t_{\min}$. When $\delta(i)$ becomes 0, the speed of the bottom system has to be updated and the next interval for which the speed of the bottom system stays constant is considered.

The operational constraints derived from the mechanical and design limitations of the machine are the following:

- the velocities of the bottom and top system are bounded.
- a box can transport only one flat.
- although the independent actions of feeding and/or dropping of different flats are performed instantaneously, the insertion and dropping of the same flat requires a nonzero amount of time.

4. CONTROL APPROACHES

In order to increase the throughput of the considered flats sorting system, we propose several control approaches to set the speed of the bottom system such as optimal control with variable speed, optimal control with piecewise constant speed, optimal control with constant speed, and model predictive control.

4.1 Optimal control with variable speed

Several methods for solving dynamic optimization problems have been developed. The optimal control problem consists of finding the time varying control law $u(\cdot)$ for a given system such that a performance index J is optimized while satisfying the operational constraints imposed by the model, see e.g. Lewis [1986].

In this paper the function J represents the throughput of the flats sorting machine, while $u(\cdot)$ is the speed of the bottom system that has to be continually adjusted. The optimal control problem is defined as follows:

P1: $\max_{u} J(u(\cdot))$
subject to
the model of the system
operational constraints

where $u : [0, T] \to \mathbb{R}$ with $T \geq 0$ the time needed to sort the entire stream of flats that enter the system in one sorting round.

However, continually adjusting the velocity of the bottom system of the considered flats sorting machine in such a way to maximize the throughput is not feasible due to the extremely high computational time required.

4.2 Optimal control with piecewise constant speed

One way to simplify the problem P1 is to divide the interval $[0, T]$ in $M+1$ periods of length T_{s}, such that $(M+1) \cdot T_{\text{s}} = T$. Defining the piecewise constant control law $u_{\text{pwct}} : \{0, 1, \cdots, M\} \to \mathbb{R}$ and the time instants $t_k = k \cdot T_{\text{s}}$ for $k = 0, 1, \cdots, M+1$, the piecewise constant speed $u(t) = u_{\text{pwct}}(k)$ for $t_k \leq t < t_{k+1}$ and $k = 0, 1, \cdots, M$ that maximizes J has to be computed.

For Algorithm 1, the length of the time interval $\delta(k)$ for $k = 0, 1, \cdots, M$ is then defined as $\delta(k) = t_{k+1} - t_k$. Accordingly, the optimal control problem is defined as follows:

P2: $\max_{U_{\text{pwct}}} J(U_{\text{pwct}})$
subject to
the model of the system
operational constraints

where $U_{\text{pwct}} = [u_{\text{pwct}}(0)\ u_{\text{pwct}}(1)\ \cdots\ u_{\text{pwct}}(M)]^{\text{T}}$.

Another way to simplify the optimal problem P1 is to divide the sorting period $[0, T]$ into $N+1$ time intervals of variable length $\delta(0), \delta(1), \cdots, \delta(N)$ such that $\sum_{j=0}^{N} \delta(j) = T$. Defining the piecewise constant control law $u_{\text{pwct}} : \{0, 1, \cdots, N\} \to \mathbb{R}$, the piecewise constant speed that maximizes J, $u(t) = u_{\text{pwct}}(k)$ for $\sum_{j=0}^{k-1} \delta(j) \leq t < \sum_{j=0}^{k} \delta(j)$ with $\sum_{j=0}^{-1} \delta(j) = 0$ has to be computed. The optimal control problem is defined as follows:

P3: $\max_{U_{\text{pwct}}, \Delta} J(U_{\text{pwct}}, \Delta)$
subject to
the model of the system
operational constraints

where $U_{\text{pwct}} = [u_{\text{pwct}}(0)\ u_{\text{pwct}}(1)\ \cdots\ u_{\text{pwct}}(N)]^{\text{T}}$ is the control sequence and $\Delta = [\delta(0)\ \delta(1)\ \cdots\ \delta(N)]^{\text{T}}$ is the corresponding sequence of variable time interval lengths.

In both cases, P2 and P3, the throughput increases with a smaller T_{s} or an increasing N until the best achievable throughput is reached. However, this comes with the cost of a higher computation time.

4.3 Optimal control with constant speed

Now we consider the simplest case of P2 and P3. For the entire stream of flats entering the system in one sorting round, the constant speed u_{ct} that maximizes the

throughput is computed. This optimal control problem can be defined as follows:

P4: $\max_{u_{\rm ct}} J(u_{\rm ct})$
subject to
the model of the system
operational constraints

where $u_{\rm ct}$ is the constant velocity of the bottom system.

The throughput obtained by solving P4 is in general smaller than the one obtained by using optimal control with variable speed, but the computation time also decreases.

4.4 Model predictive control

In order to make a trade-off between the optimality of the throughput and the time required to compute the optimal velocity sequence of the bottom system, model predictive control (MPC) is introduced.

Model predictive control is an on-line control design method that uses the receding horizon principle, see e.g. Allgöwer et al. [1999], Camacho and Bordons [1995], Maciejowski [2002].

In this approach, given the prediction horizon $N_{\rm p}$ and the control horizon $N_{\rm c}$ with $N_{\rm c} \leq N_{\rm p}$, at time step k, the future control sequence $u(k|k), \cdots, u(k+N_{\rm c}-1|k)$ is computed by solving a discrete-time optimization problem over a given period $[k, k+N_{\rm p}]$ so that a cost criterion J is optimized subject to constraints on the inputs and outputs. The input signal is assumed to become constant beyond the control horizon i.e. $u(k+j|k) = u(k+N_{\rm c}-1|k)$ for $j \geq N_{\rm c}$. After computing the optimal control sequence, only the first control sample is implemented, and subsequently the horizon is shifted. Next, the new state of the system is measured or estimated, and a new optimization problem at time $k+1$ is solved using this new information. In this way, a feedback mechanism is introduced.

The optimization problem can be solved by using a modified version of optimal control with piecewise constant speed on constant time intervals where at time step k, $u_{\rm pwct}(j) = u(j|k)$ for $j = 0, 1, \cdots, N_{\rm c}-1$.

P5: $\max_{U_{\rm pwct}} J(U_{\rm pwct})$
subject to
the model of the system
operational constraints
control horizon constraint

where $U_{\rm pwct} = [u_{\rm pwct}(0)\ u_{\rm pwct}(1)\ \cdots\ u_{\rm pwct}(N_{\rm c}-1)]^{\rm T}$.

This way, at time step k, the control sequence $U_{\rm pwct}$ that maximizes the throughput over the period $[k, k+N_{\rm p}]$ is computed. The prediction horizon presented above, is determined by using the following procedure[1]. Assume that we want to sort in advance b flats. For $T_{\rm s} \cdot N_{\rm c}$ seconds the velocity sequence $U_{\rm pwct}$ is used, each velocity $u_{\rm pwct}(j)$, with $j = 0, 1, \cdots, N_{\rm c}-1$ being applied on a sampling period of length $T_{\rm s}$, sorting $b_1 \leq b$ flats. For sorting the rest of $b - b_1$ flats, the velocity of the bottom system is kept constant, equal to $u_{\rm pwct}(N_{\rm c}-1)$. Consequently, the time T needed to sort the b flats divided by $T_{\rm s}$ determines the prediction horizon as follows: $N_{\rm p} = \lceil \frac{T}{T_{\rm s}} \rceil$ where $\lceil x \rceil$ denotes the smallest integer greater than or equal to x.

The main advantage of MPC is given by a smaller computation time than the one obtained when using optimal control with variable speed. Even more, the velocity of the bottom system may be computed on-line. However, this happens at the cost of a suboptimal throughput.

[1] In this variant of MPC the prediction horizon corresponds to the number of flats to be sorted in advance. This is required in order to be able to compare the obtained throughput values for different control inputs in a correct way.

5. SIMULATION RESULTS

In this section we compare the proposed control methods based on simulation examples.

5.1 Scenarios

Recall that the velocity of the top system is constant. We assume it to be equal to 1 m/s, while the velocity of the bottom system varies between -2 m/s and 2 m/s. It is also assumed that the width of the bin is four times the width of the box. For the examples in this section we consider $N_{\rm bin} = 10$ bins, respectively $N_{\rm box} = 40$ boxes, while the length of the stream of flats that enter the sorting system equals 2000.

We consider several scenarios, where the stream consists of:

(1) perfectly ordered codes (i.e. with the same order as the order of codes allocated to the bins). Since the width of the bin is four times the width of the box, by perfectly ordered flats we mean e.g. $1,1,1,1,2,2,2,2,\cdots,N_{\rm bin},N_{\rm bin},N_{\rm bin},N_{\rm bin},1,1,1,1,\cdots$. The order of the $N_{\rm bin}$ bins passing under the first feeding device when the bottom system moves to the right is in this case $1,2,3,\cdots,N_{\rm bin}$.
(2) alternating sequences of random, and respectively ordered codes e.g. $s_1, s_2, \cdots, s_l$ where if s_i is a sequence of random codes, $1 \leq i < l$, then s_{i+1} is a sequence of ordered codes. The length of each sequence s_i for $i = 1, 2, \cdots, l$ is also randomly chosen. This scenario has been chosen due to the fact that the mail may be presorted.
(3) completely random codes.

For the scenarios listed above, the throughput of the flats sorting machine where the bottom system of the flats sorting machine is static, has been listed in Table 1.

	$J_{\rm opt}$ (flats/s)	
Scenario	**1 feeder**	**2 feeders**
1	10.000	13.157
2	9.891	13.404
3	9.837	13.012

Table 1. Throughput for the considered scenarios, when the bottom system of the sorting machine is static.

5.2 Optimal control with constant speed

In this section we compute the constant speed of the bottom system that optimizes the throughput for all the flats that enter the system in one sorting round.

In order to determine the optimal speed, one may use the Matlab functions *fmincon* incorporated in the Optimization Toolbox, see e.g. Han [1977], or *patternsearch* and *ga*, incorporated in the Genetic Algorithm and Direct Search Toolbox see e.g. Lewis and Torczon [2000], Goldberg [1989].

The *fmincon* function finds a local minimum of a smooth function based on gradient methods, while *patternsearch* and *ga* determine a local minimum of a non-smooth objective function. Many variations can be noticed while plotting throughput versus velocity of the bottom system by discretizing the velocity with the sampling step of e.g. 0.01 m/s as illustrated in Figure 4.

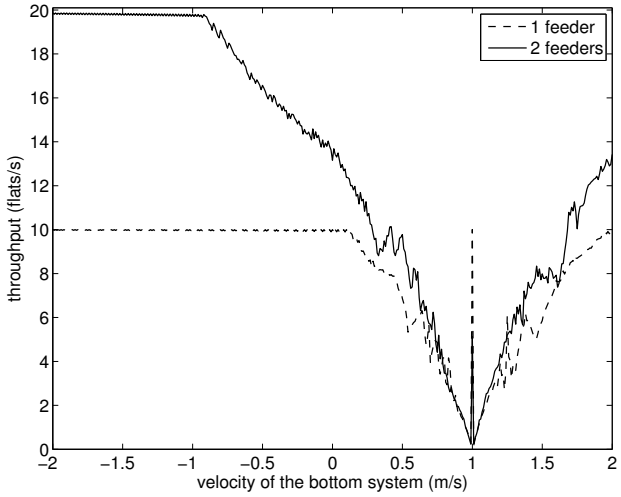

Fig. 4. Throughput vs. velocity for scenario 1 when the length of the initial stream is 2000.

Also, the amplitude of the variations increases when the length of the initial stream decreases. Therefore, multiple initial points for *patternsearch* and *fmincon* and multiple runs for *ga* have to be used. By comparing the best throughput attained for each of the three optimization routines and the corresponding computation time, we have decided to further use the *patternsearch* function in solving the optimization problems.

The results obtained when using optimal control with constant speed are shown in Table 2.

Scenario	1 feeder			2 feeders		
	J_{opt} (flats/s)	v_{opt} (m/s)	improvement (%)	J_{opt} (flats/s)	v_{opt} (m/s)	improvement (%)
1	10.005	1	0.05	19.905	−1.96	51.28
2	9.972	−1.88	0.8	19.879	−1.97	48.30
3	9.979	−1.87	1.44	19.848	−1.85	52.53

Table 2. Comparison of maximal throughput and optimal velocity for the considered scenarios, using optimal control with constant speed.

Since the improvement obtained by using the proposed set-up with only one feeder is not substantial, only the sorting machine with two feeders is further considered.

5.3 Optimal control with piecewise constant speed

After sampling the sorting time, the *piecewise* constant speed of the bottom system that optimizes the throughput for the entire stream is computed. The throughput increases with the length N or M of the control sequence until the best achievable throughput is approached. This is 19.908 flats/s for the first scenario, 19.885 flats/s for the second scenario, and 19.861 flats/s for the third one.

Also, simulations show that, when solving P2 and P3 for the control sequence U_{pwct} with the same length P, the throughput gain obtained with P3 does not outweigh the computation time for $P \geq 4$ since the result is the same within an accuracy of 10^{-3}, but at the cost of more than double computation time.

5.4 Model predictive control

When applying MPC, the smaller T_s is chosen the bigger N_c has to be set in order to maximize the throughput. We first consider improving the performance and afterwards, the computation effort is taken into account.

To obtain the best throughput a maximum prediction horizon is selected, while N_c is set equal to N_p. Therefore, the time T needed to sort the entire stream of flats using optimal control with constant speed is computed. Accordingly, the prediction horizon is set to $\lceil \frac{T}{T_s} \rceil$, while $N_c = N_p$. Various lengths T_s of the time period have been considered. Based on simulation results, it has been noticed that for $T_s < 1\,\text{s}$ the resulting values of the throughput stop increasing. This procedure also gives high performance, but is not feasible due to the computation effort.

Simulations indicate that applying MPC with a prediction horizon determined by a buffer of 80 flats known in advance, $N_c = N_p$, and $T_s = 1\,\text{s}$ gives already the throughput in within 1% deviation of the best throughput achieved when applying optimal control with piecewise constant speed. The results are: 19.895 flats/s for the first scenario, 19.854 flats/s for the second scenario, and 19.827 flats/s for the third one. Nevertheless, high computation time is still required.

The computation time can be lowered by using the Matlab function *patternsearch* with less initial points and by adjusting its search options. Also, the smaller the control horizon becomes, the more the time needed to compute the control sequence $u_{\text{pwct}}(0), \cdots, u_{\text{pwct}}(N_c - 1)$ decreases. This time can be reduced up to a maximum of 2.5 s for $N_c = 1$, up to 5 s for $N_c = 2$, up to 8 s for $N_c = 3$ etc. [2]. According to the MPC procedure, after computing the optimal control sequence, only the first control is applied and the horizon is shifted. But, depending on the choice of T_s, the computation time of the control sequence U_{pwct} may be larger than the sampling period. This problem can be circumvented either by using larger

[2] The simulations were performed on a 3.0 GHz P4 with 1 GB RAM.

T_s or by applying not only the first control $u_{\text{pwct}}(0)$ after solving the optimization problem, but several controls $u_{\text{pwct}}(0), \cdots, u_{\text{pwct}}(j)$, where $j \leq N_c - 1$ such that $T_s \cdot (j+1)$ is greater than the computation time, and shift the horizon accordingly. This method produces real-time, but suboptimal results e.g. 19.861 flats/s for the first scenario, 19.842 flats/s for the second scenario, and 19.860 flats/s for the third one ($N_c = 1$ and $T_s = 3$ s).

5.5 Discussion

Based on simulations, a summary of the obtained results is presented in Table 3.

Control method	CPU time (s)	relative performance (%)
opt. ctrl. with variable speeds and time intervals	10^5	100
opt. ctrl. with variable speeds on constant time intervals	$2 \cdot 10^4$	100
opt. ctrl. with constant speed	80	99.96
MPC	10^3	99.86
real-time MPC	45	99.84
static bottom system	0	66.33

Table 3. Comparison of maximal throughput and computation time for the proposed control methods.

It has been assumed that the maximum achievable throughput is obtained by using the optimal control with piecewise constant speed on time intervals of variable length. The performance of the other approaches was computed relative to this maximum. But, for each of the control methods to be compared, the throughput corresponding to the chosen scenarios varies. Therefore, the average throughput is used in calculating the relative performance. The computational time is also averaged and rounded.

Simulation results show that applying MPC gives a good trade-off between the computation time and the maximum achievable throughput. Also, the optimal control approach is not feasible, in the sense that in reality the entire stream of flats that enters the system in one sorting round is not known in advance. Only a finite buffer b of codes is known beforehand, b depending on the maximum time allowed to prepare the flats for sorting. Therefore, MPC is suitable in determining the optimal velocity sequence of the bottom system. Also, variants of MPC described in Section 5.4 allow real-time control, still assuring over 99.7% of the maximum throughput for a minimum buffer of 40 flats known in advance.

6. CONCLUSIONS AND FUTURE RESEARCH

In this paper we have presented a short description of how flats sorting machines currently work, have proposed a new set-up by making minor design changes i.e. adding an extra feeding device and moving the bottom bin system, and have implemented advanced control techniques so as to assure the optimal maneuvers. We have also presented an event-driven model of the process. The considered control approaches are optimal control and model predictive control.

Results indicate that the most appropriate control method to set the bottom system's velocity of the proposed flats sorting machine is model predictive control. This conclusion is drawn based on the following reasons: the method is suitable for real flats sorting machine where only a finite buffer of flats is known in advance. The performance is influenced by the prediction horizon and the control horizon. For a minimum buffer of 40 flats known in advance, the MPC procedure assures over 99.8% of the best throughput, respectively over 99.7% when using the real-time MPC variant.

Based on the results in Table 1 and Table 2 one may note that only by maneuvering the bottom system an improvement of the throughput up to 1.44% is achieved, but when adding a second feeding device, the throughput is increased with up to 52.53%. This opens future lines of research, such as analyzing how the number of feeders, their position, and the velocity of the top system influences the throughput of the automated flats sorting machine. Also, we will perform sensitivity analysis of the proposed control methods and conduct experiments in a real physical setting.

ACKNOWLEDGEMENTS

This research is supported by the VIDI project "Multi-Agent Control of Large-Scale Hybrid Systems" of the Dutch Technology Foundation STW, Applied Science division of NWO and the Technology Programme of the Dutch Ministry of Economic Affairs, by the BSIK project "Next Generation Infrastructures" (NGI), by the Transport Research Centre Delft, and by the Delft Research Centre Next Generation Infrastructures.

REFERENCES

F. Allgöwer, T.A. Badgwell, S.J. Qin, J.B. Rawlings, and S.J. Wright. Nonlinear predictive control and moving horizon estimation – an introductory overview. In *Advances in Control: Highlights of ECC'99*, pages 391–449. Springer, London, UK, 1999.

E.F. Camacho and C. Bordons. *Model Predictive Control in the Process Industry.* Springer-Verlag, Berlin, Germany, 1995.

D.E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley, Boston, Massachusetts, USA, 1989.

S.P. Han. A globally convergent method for nonlinear programming. *SIAM Journal on Optimization*, 22(3): 297–309, 1977.

F.L. Lewis. *Optimal Control.* John Wiley & Sons, New York, USA, 1986.

R.M. Lewis and V. Torczon. Pattern search methods for linearly constrained minimization. *SIAM Journal on Optimization*, 10(3):917–941, 2000.

B. Lohmann. Throughput control for a transport process and an application in postal automation machines. *Control Engineering Practice*, 4(11):1503–1509, 1996.

B. Lohmann. An application in postal automation: Two ways of modeling a transport process. In *Proceedings of 2nd MATHMOD*, pages 449–453, Vienna, Austria, February 1997.

J.M. Maciejowski. *Predictive Control with Constraints.* Prentice Hall, Harlow, UK, 2002.