Delft University of Technology

Delft Center for Systems and Control

Technical report 09-024

Centralized, decentralized, and distributed model predictive control for route choice in automated baggage handling systems*

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

If you want to cite this report, please use the following reference instead:
A.N. Tarău, B. De Schutter, and J. Hellendoorn, "Centralized, decentralized, and distributed model predictive control for route choice in automated baggage handling systems," *Control Engineering and Applied Informatics,* Special Issue on Distributed Control in Networked Systems, vol. 11, no. 3, pp. 24–31, 2009.

Delft Center for Systems and Control
Delft University of Technology
Mekelweg 2, 2628 CD Delft
The Netherlands
phone: +31-15-278.24.73 (secretary)
URL: `https://www.dcsc.tudelft.nl`

*This report can also be downloaded via `https://pub.deschutter.info/abs/09_024.html`

Centralized, decentralized, and distributed model predictive control for route choice in automated baggage handling systems

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: In this paper we develop and compare efficient predictive control methods for routing individual vehicles which ensure automatic transportation of bags in a baggage handling system of an airport. In particular we consider centralized, decentralized, and distributed model predictive control (MPC). To assess the performance of the proposed control approaches, we consider a simple benchmark case study, in which the methods are compared for several scenarios. The results indicate that the best performance of the system is obtained when using centralized MPC. However, centralized MPC becomes intractable when the number of junctions is large due to the high computational effort this method requires. Decentralized and distributed MPC offer a balanced trade-off between computation time and optimality.

Keywords: Baggage handling systems, route choice control, model predictive control.

1. INTRODUCTION

The increasing need for cost efficiency of the air transport industry and the rise of low-cost carriers require a cost effective operation of the airports. The state-of-the-art technology used by baggage handling systems at airports to transport the bags in an automated way incorporates scanners that scan the labels on each piece of luggage, baggage screening equipment for security scanning, networks of conveyors equipped with junctions that route the bags through the system, and destination coded vehicles (DCVs). As illustrated in Figure 1, a DCV is a metal cart with a plastic tub on top. These carts transport the bags at high speed on a network of tracks.

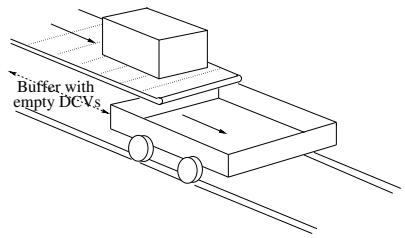


Fig. 1. Loading a DCV.

In this paper we consider a DCV-based baggage handling system. Higher-level control problems for such a system are route assignment for each DCV (and implicitly the switch control of each junction), line balancing (i.e. route assignment for each empty DCV such that all the loading stations have enough empty DCVs at any time instant), and prevention of buffer overflows. The velocity control of each DCV is a medium-level control problem. The medium-level controller on board of each DCV ensures a minimum safe distance between DCVs and also hold DCVs at switching points, if required. We assume that the velocity of each DCV is always at its maximum, $v^{\max}$, unless overruled by the local on-board collision avoidance controller. Finally, the low-level control problems are coordination and synchronization when loading a bag onto a DCV (in order to avoid damaging the bags or blocking the system), and when unloading it to its end point. Note that we assume the low-level controllers already present in the system.

In this paper we focus on the higher-level control problem of optimally routing DCVs on the network of tracks so that all the bags to be handled arrive at their end points within given time windows. Currently, the networks are simple, the DCVs being routed through the system using routing schemes based on preferred routes. These routing schemes can be adapted to respond on the occurrence of predefined events. In the research we conduct we consider more complex networks. Also, we do not consider predefined preferred routes. Instead we develop advanced control methods to determine the optimal routing in case of dynamic demand.

In the literature, the route assignment problem has been addressed by e.g. Gang et al. (1996), Kaufman et al. (1998). But, in our case we do not deal with a shortest-path or shortest-time problem, since, due to the airport's logistics, we need the bags at their end points within given time windows. The route choice problem for a DCV-based baggage handling system has been analyzed by e.g. Fay (2005) where an analogy to data transmissions via internet is proposed, and by e.g. Hallenborg and Demazeau (2006) where a multi-agent approach has been developed. However, this multi-agent system deals with major challenges due to the extensive communication required. Therefore, the goal of our work is to develop and compare efficient control approaches (viz. predictive control methods

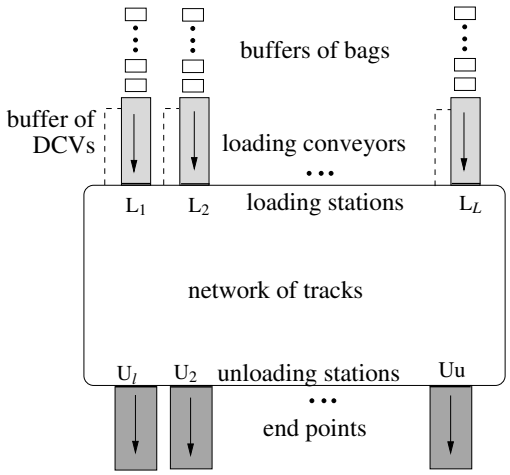


Fig. 2. Baggage handling system using DCVs.

and heuristic approaches) for route choice control of each DCV transporting a bag through the track network. These control approached are developed in a centralized, a decentralized, and a distributed manner. The control approach is said to be decentralized if the local control actions are computed without any communication or coordination between the local controllers, while the control approach is said to be distributed if additional communication and coordination between neighboring controllers is involved, see e.g. Siljak (1991) and Weiss (1999).

The paper is organized as follows. We start by giving a brief process description of a DCV-based baggage handling system (Section 2). Next, in Section 3, we determine a continuous-time event-driven model of the system. Afterwards, in Section 4, the operation constraints and the global performance index are elaborated. In Section 5, we propose advanced control methods for computing the route of each DCV in a centralized, a decentralized, and a distributed manner. The analysis of the simulation results and the comparison of the proposed control methods are presented in Section 6. Finally, in Section 7, conclusions are drawn and the directions for future research are presented.

2. OPERATION OF THE SYSTEM

Consider the general DCV-based baggage handling system sketched in Figure 2. This baggage handling system operates as follows: given a demand of bags (identified by their unique code) together with their arrival times at the loading stations, and the network of tracks, the route of each DCV (from a given loading station to the corresponding unloading station) has to be computed subject to the operational and safety constraints presented in Section 4, such that all the bags to be handled arrive at their end points within given time windows. The bags unloaded outside their end points' time window are then penalized as presented in Section 4.2.

We consider a system with L loading stations L_1, L_2, …, L_L and U unloading stations U_1, U_2, …, U_U as depicted in Figure 2. Let us index the bags loaded onto DCVs at station L_l with $\ell \in \{1,\ldots,L\}$ as $b_{\ell,1},...,b_{\ell,N_l}$ with N_ℓ the number of bags that will be loaded at station L_ℓ during the entire simulation period. Then let $t^{\text{arrival}}_{\ell,i}$ denote the time instant when bag $b_{\ell,i}$ actually arrives at loading station L_ℓ ($t^{\text{arrival}}_{\ell,i} < t^{\text{arrival}}_{\ell,i+1}$ for $i = 1,\ldots,N_\ell - 1$). Then we define the L-tuple $\mathscr{T} = (\mathbf{t}^{\text{arrival}}_1, \mathbf{t}^{\text{arrival}}_2, \ldots, \mathbf{t}^{\text{arrival}}_L)$ that comprises the vectors of bag arrival times $\mathbf{t}^{\text{arrival}}_\ell = [t^{\text{arrival}}_{\ell,1} \ldots t^{\text{arrival}}_{\ell,N_\ell}]^{\mathrm{T}}$ with $\ell \in \{1,2,\ldots,L\}$. We also consider that the track network has S junctions S_1, S_2, …, S_S.

3. MODEL

In this section we present the simplifying assumptions and the continuous-time event-driven model to be used in order to determine the optimal route choice for DCVs in a baggage handling system.

3.1 Assumptions

Later on we will use the model for on-line model-based control. So, in order to obtain a balanced trade-off between a detailed model that requires large computation time and a fast simulation we make the following assumptions:

- A_1: a sufficient number of DCVs are present in the system so that when a bag is at the loading station there is a DCV ready for transporting it,
- A_2: the network of tracks has single-direction tracks,
- A_3: we assume each loading station to have only one outgoing link and each unloading station to have only one incoming link,
- A_4: each junction S_s with $s \in \{1,2,\ldots,S\}$ has maximum 2 incoming links and 2 outgoing links, both indexed by $l \in \{0,1\}$ as sketched in Figure 3. If S_s has 2 incoming links then it also has a switch going into the junction (called switch-in hereafter). If S_s has 2 outgoing links then it has also a switch going out of the junction (called switch-out hereafter). Note that a junction can have only switch-in, only switch-out, or both switch-in and switch-out.
- A_5: a route switch at a junction can be performed in a negligible time span,
- A_6: the speed of a DCV is piecewise constant,
- A_7: the end points have capacity large enough that no buffer overflow can occur,
- A_8: the destinations to which the bags have to be transported are allocated to the end points when the process starts.

Since we consider the line balancing problem solved, these assumptions are reasonable and give a good approximation of the real baggage handling system.

3.2 Event-based model

Later on the model of the DCV-based baggage handling system will be used for on-line model-based control. So, in order to obtain a fast simulation, we write the model as an event-driven one consisting of a continuous part describing the movement of the individual vehicles transporting bags through the network, and of the following discrete events: loading a new bag into the system, unloading a bag that arrives at its end point, updating the position of the switch switch-in, and updating the position

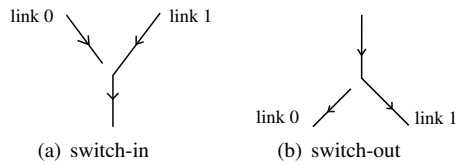


Fig. 3. Incoming and outgoing links at a junction.

of a switch-out at a junction, and updating the velocity of a DCV.

Let X be the number of bags that the baggage handling system has to handle and let X^{crt} be the total number of bags that entered the track network up to the current time instant $t^{\text{crt}} \leq t_0 + T^{\max}$ with t_0 the initial simulation time and $T^{\max}$ the maximum simulation period. Also, let DCV_i denote the DCV that transports the ith bag that entered the track network up to the current time instant, $i \leq X^{\text{crt}}$. Note that in case that 2 or more bags are loaded onto DCVs at the same time instant t, we order the DCVs according to the index of the loading stations (DCV_i will then denote the DCV transporting the bag loaded at the loading station with the smallest index, DCV_{i+1} will denote the DCV transporting the bag loaded at the loading station with the next smallest index, and so on).

The state of the DCV-based baggage handling system consists of the just-crossed junction, and the next-to-be-crossed junction for each DCV, their speed and their position on the link that the DCVs travel, and the position of the switch-in and switch-out at each junction. Then the model of the baggage handling system is given by the algorithm below.

Algorithm 1. Model of the baggage handling system

1: $t^{\text{crt}} \leftarrow t_0$
2: **while** $t^{\text{crt}} \leq t_0 + T^{\max}$ **do**
3: **for** $\ell = 1$ to L **do**
4: $\tau_\ell^{\text{load}} \leftarrow$ time that will pass until the next loading event of L_ℓ
5: **end for**
6: **for** $m = 1$ to U **do**
7: $\tau_m^{\text{unload}} \leftarrow$ time that will pass until the next unloading event of U_m
8: **end for**
9: **for** $s = 1$ to S **do**
10: $\tau_s^{\text{sw\_in}} \leftarrow$ time that will pass until the next switch-in event at S_s
11: $\tau_s^{\text{sw\_out}} \leftarrow$ time that will pass until the next switch-out event at S_s
12: **end for**
13: **for** $i = 1$ to X^{crt} **do**
14: $\tau_i^{\text{v\_update}} \leftarrow$ time that will pass until the next velocity-update event of DCV_i
15: **end for**
16: $\tau^{\min} \leftarrow \min\big(\min_{\ell=1,2,\ldots,L} \tau_\ell^{\text{load}}, \min_{m=1,2,\ldots,U} \tau_m^{\text{unload}}, \min_{s=1,2,\ldots,S} \tau_s^{\text{sw\_in}}, \min_{s=1,2,\ldots,S} \tau_s^{\text{sw\_out}}, \min_{i=1,\ldots,X^{\text{crt}}} \tau_i^{\text{v\_update}}\big)$
17: $t^{\text{crt}} \leftarrow t^{\text{crt}} + \tau^{\min}$
18: take action (i.e. load, unload, switch-in update, switch-out update, velocity-update)
19: update the state of the system
20: **end while**

If multiple events occur at the same time, then we take all these events into account when updating the state of the system (i.e. the position and the speed of DCVs, and the position of switch-in and switch-out at junctions) at step 19.

According to the model, for each bag that has to be handled, we compute the time instants when the bag enters and exits the track network. Let t_i^{load} denote the time instant when the ith bag that entered the track network is loaded onto a DCV and let t_i^{unload} denote the time instant when the same bag is unloaded at its end point. Consequently, we denote the model of the baggage handling system as $\mathbf{t} = \mathscr{M}(\mathscr{T}, x(t_0), \mathbf{u}, \mathbf{v})$, where:

- $\mathbf{t} = [t_1^{\text{load}} \ldots t_X^{\text{load}}\, t_1^{\text{unload}} \ldots t_X^{\text{unload}}]^{\text{T}}$ with X be the number of bags that the system has to handle,
- $\mathscr{T} = (\mathbf{t}_1^{\text{arrival}}, \mathbf{t}_2^{\text{arrival}}, \ldots, \mathbf{t}_L^{\text{arrival}})$ defined in Section 2,
- $x(t_0)$ is the initial state of the system with t_0 the initial simulation time,
- $\mathbf{u}$ is the route control sequence,
- $\mathbf{v}$ is the velocity sequence for each DCV.

4. CONSTRAINTS AND CONTROL OBJECTIVE

In this section we present the safety and operational constraints of a DCV-based baggage handling system, together with the control objective to be used when comparing the proposed control methods.

4.1 Operational constraints

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

C_1: a DCV can transport only one bag at the time,
C_2: a bag can be loaded onto a DCV only if there is an empty DCV under the loading station. This means that if there is a traffic jam at a loading station, then no loading event can occur at that loading station.
C_3: a switch at a junction changes its position after minimum τ_{x} time units in order to avoid the quickly and repeatedly movement back and forth of the switch which may lead to mechanical damage,
C_4: the speed of each DCV is bounded between 0 and $v^{\max}$.

These constraints are denoted by $\mathscr{C}(\tau_{\text{x}}, v^{\max}) \leq 0$.

4.2 Control objective

Since the baggage handling system performs successfully if all the bags are transported to their end point before a given time instant, from a central point of view, the primary objective is the minimization of the overdue time. A secondary objective is the minimization of the additional storage time at the end point. This objective is required due to the intense utilization of the end points in a busy airport. Hence, one way to construct the objective function J_i^{pen} corresponding to the bag with index i, $i \in \{1, 2, \ldots, X\}$, is to penalize the overdue time and the additional storage time. Accordingly, we define the following penalty for bag index i:

$$J_i^{\text{pen}}(t_i^{\text{unload}}) = \sigma_i \max(0, t_i^{\text{unload}} - t_i^{\text{end}}) + \lambda_1 \max(0, t_i^{\text{end}} - \tau_i^{\text{open}} - t_i^{\text{unload}}) \quad (1)$$

where

- t_i^{end} is the time instant when the end point closes and the bags are loaded onto the plane.
- τ_i^{open} is the maximum possible length of the time window for which the end point corresponding to bag index i is open for that specific flight.
- σ_i is the static priority of bag index i (the flight priority), $0 < \sigma_i \leq \sigma_i^{\max}$ with $\sigma^{\max} > 1$ the maximum priority that can be assigned to a flight.
- $\lambda_1 < 1$ is a nonnegative weighting parameter that expresses the penalty for the additional storage time.

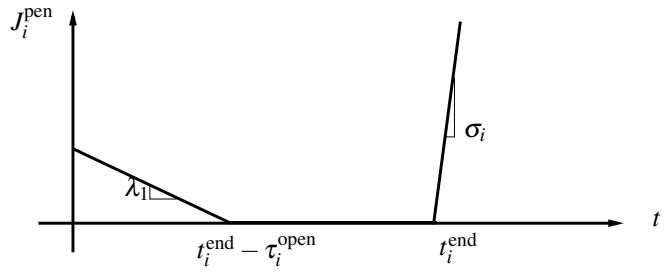


Fig. 4. Objective function J_i^{pen}.

However, the above performance function has some flat parts, which yield difficulties for many optimization algorithms. Therefore, in order to get some additional gradient we also include the dwell time. This results in:

$$J_i(t_i^{\text{unload}}) = J_i^{\text{pen}}(t_i^{\text{unload}}) + \lambda_2(t_i^{\text{unload}} - t_i^{\text{load}}) \tag{2}$$

where λ_2 is a small weight factor ($0 < \lambda_2 \ll 1$).

The final objective function to be used when comparing the proposed control approaches is given by:

$$J^{\text{tot}} = \sum_{i=1}^{X} J_i^{\text{pen}}(t_i^{\text{unload}}) \tag{3}$$

Note that the objective function $J_i^{\text{pen}}(t_i^{\text{unload}})$ depends on the unloading time of bag index i at its end point, and implicitly it depends on the routes of all the bags to be handled.

5. CONTROL APPROACHES

In this section we propose centralized, decentralized, and distributed model predictive control to determine the route of each DCV transporting a bag.

5.1 Centralized model predictive control

Model predictive control (MPC) is an on-line model-based predictive control design method (see e.g Maciejowski (2002), Allgöwer et al. (1999), Camacho and Bordons (1995)). In the basic MPC approach, given an horizon N, at step k, the future control sequence $u(k+1), u(k+2), \ldots, u(k+N)$ is computed by solving a discrete-time optimization problem over a period $[t_k, t_k + \tau_s N]$, where $t_k = t_0 + k\tau_s$ with τ_s the sampling time, so that a cost criterion is optimized subject to the operational constraints. MPC uses a receding horizon approach. So, 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 t_{k+1} is solved using this new information. In this way, also a feedback mechanism is introduced.

We define now a variant of MPC, where k is not a time index, but a bag index. In this context bag step k denotes the time instant when the kth bag entered the track network. Also, the horizon N corresponds to the number of bags that we let enter the track network after bag step k. Computing the control $u(k+j)$, with $j \in \{1,2,\ldots,N\}$ consists in determining the route of DCV_{k+j}. Assume that there is a fixed number R of possible routes from a loading station to an unloading station. The R routes are indexed $1,2,\ldots,R$. Let $r(i) \in \{1,2,\ldots,R\}$ denote the route of DCV_i. We assume that, at bag step k the route is selected once for each DCV without being adjusted after the decision has been made. Now let $\mathbf{r}(k)$ denote the future route sequence for the next N bags entering the network after bag step k, $\mathbf{r}(k) = [r(k+1)\, r(k+2) \ldots r(k+N)]^{\text{T}}$.

The total performance function of the centralized MPC is defined as $J_{k,N}^{\text{CMPC}}(\mathbf{r}(k)) = \sum_{i=1}^{k+N} J_i(\hat{t}_i^{\text{unload}})$ where $\hat{t}_i^{\text{unload}}$ is the estimated arrival time of DCV_i depending on the routes of the first $k+N$ bags that entered the network. Accordingly, the MPC optimization problem at bag step k is defined as follows:

$$\begin{aligned} &\min_{\mathbf{r}(k)} J_{k,N}^{\text{CMPC}}(\mathbf{r}(k)) \\ &\text{subject to} \\ &\qquad \mathbf{t} = \mathscr{M}(\mathscr{T}, x(t_k), \mathbf{r}(k), \mathbf{v}) \\ &\qquad \mathscr{C}(v^{\max}, \tau_x) \leq 0 \end{aligned}$$

Centralized MPC can compute on-line the route of each DCV in the network, but it requires large computational efforts as will be illustrated in Section 6. Therefore, we will also propose decentralized and distributed control approaches that offer a trade-off between the optimality of the performance of the controlled system and the time required to compute the solution.

5.2 Decentralized model predictive control

In decentralized model predictive route choice control we consider each junction separately, as a local system. For all junctions we will then define similar local MPC problems.

Local system Each local system consists of a junction, its incoming and its outgoing links. Let us now consider the most complex case, where junction S_s with $s \in \{1,2,\ldots,S\}$ has both a switch-in and a switch-out. Moreover, S_s is not directly connected to an unloading station. For the sake of simplicity of notation, in the remainder of this subsection, we will not explicitly indicate the subscript s for variables that refer to junction S_s. Next we index [1] the bags that successively cross junction S_s during the entire simulation period as $b_1, b_2, \ldots, b_{N^{\text{bags}}}$, where N^{bags} is the number of bags that cross S_s during the simulation period.

Local control measures In decentralized route choice control we compute the positions of the switch-in and switch-out of junction S_s for each bag that crosses S_s. For all the other junctions, the same procedure is applied.

Recall from Section 5.1 that we use a variant of MPC with a bag index. So, in this approach, the local control is updated at every time instant when some bag has just entered an incoming link of junction S_s. Let t^{crt} be such a time instant. Then for junction S_s we determine bag index k such that $t_k^{\text{cross}} \leq t^{\text{crt}} < t_{k+1}^{\text{cross}}$, where t_k^{cross} is defined as the time instant when bag b_k has just crossed the junction. If no bag has crossed the junction yet, we set $k = 0$.

Let $N^{\max}$ be the maximum prediction horizon for a local MPC problem and n_l^{horizon} the number of DCVs traveling at time instant t^{crt} on link l going into S_s. Then, the local optimization is performed over the next $N = \min\left(N^{\max}, \sum_{l=0}^{1} n_l^{\text{horizon}}\right)$ bags that will pass junction S_s after bag index k. By solving this local optimization problem we compute the control sequence $\mathbf{u}(k) = [u^{\text{sw\_in}}(k+1) \ldots u^{\text{sw\_in}}(k+N)\, u^{\text{sw\_out}}(k+1) \ldots u^{\text{sw\_out}}(k+N)]^{\text{T}}$ corresponding to the next N bags $b_{k+1}, b_{k+2}, \ldots, b_{k+N}$ that will cross the junction. The control decisions $u^{\text{sw\_in}}(k+1), \ldots, u^{\text{sw\_in}}(k+N)$ of the switch into S_s determine the order

[1] This order depends on the evolution of the position of the switch-in at junction S_s.

in which the bags cross the junction and the time instants at which the bags $b_{k+1},\ldots,b_{k+N}$ enter S_s. The control decisions $u^{\text{sw\_out}}(k+1),\ldots,u^{\text{sw\_out}}(k+N)$ determine the next junction towards which the bag $b_{k+1},\ldots,b_{k+N}$ will travel.

Local objective function When solving the local MPC optimization problem for junction S_s, we will use a local objective function $J_{k,N}^{\text{DMPC}}$. The local objective function is computed via a simulation of the local system for the next N bags that will cross the junction, being defined as follows:

$$J_{k,N}^{\text{DMPC}}(\mathbf{u}(k)) = \sum_{j=1}^{\min(N,N_s^{\text{cross}})} J_{k+j}(\hat{t}_{k+j}^{\text{unload},*}) + \lambda^{\text{pen}}(N - N_s^{\text{cross}})$$

where N_s^{cross} is the number of DCVs that actually crossed junction S_s during the prediction period, $\hat{t}_{k+j}^{\text{unload},*}$ is the predicted unloading time instant of bag b_{k+j}, and λ^{pen} is a nonnegative weighting parameter. The variables N_s^{cross} and $\hat{t}_{k+j}^{\text{unload},*}$ are determined by simulating the prediction model presented next for a given control sequence $\mathbf{u}(k)$.

Local prediction model The local prediction model at bag index k is an event-driven model for the local system over an horizon of N bags. So, according to **Algorithm 1**, for the next N bags to cross S_s, given the current state of the local system, we compute the period $\tau^{\min}$ until the next event will occur in the local system (loading if S_s is connected to loading stations, unloading if S_s is connected to unloading stations, switching at S_s, updating the speed of a DCV running through the local system), we shift the current time with $\tau^{\min}$, take the appropriate action, and update the state of the local system.

Recall that we consider S_s to be connected via its outgoing links to junctions that are not unloading stations. Hence, we have to estimate the time when each of the next N bags to cross S_s will reach their end point. To this aim, we first consider a fixed release rate during the prediction period for each outgoing link $l \in \{0,1\}$ of S_s. Let ζ_l be the fixed release rate at time instant t^{crt}.

Next we present how we calculate ζ_l given the state of the local system at t^{crt}. Let τ^{rate} be the length of the time window over which we compute the link release rate. The variable τ^{rate} can be derived using empirical data. If $t^{\text{crt}} < \tau^{\text{rate}}$ we consider $\zeta_l = \zeta^{\max}$ with $\zeta^{\max}$ the maximum number of DCVs per time unit that can cross a junction using maximum speed. If $t^{\text{crt}} \geq \tau^{\text{rate}}$, let n_l^{rate} denote the number of DCVs that left the outgoing link l within the time window $[t^{\text{crt}} - \tau^{\text{rate}}, t^{\text{crt}}]$. Then, if $n_l^{\text{rate}} > 0$ the fixed release rate of link l out of S_s to be used during the entire prediction period is given by $\zeta_l = \dfrac{n_l^{\text{rate}}}{\tau^{\text{rate}}}$, while if $n_l^{\text{rate}} = 0$ we set $\zeta_l = \varepsilon$ with $0 < \varepsilon \ll 1$.

Now we want to determine the arrival time of bag b_{k+j} with $j \in \{1,\ldots,N\}$ at its end point. Let S_l^{next} denote the junction that bag b_{k+j} will cross next, where $l = u^{\text{sw\_out}}(k+j)$ and let S_{k+j}^{dest} be the end point of bag b_{k+j}. Then, for each possible route $r \in \mathscr{R}_{l,k+j}^{\text{next}}$, where $\mathscr{R}_{l,k+j}^{\text{next}}$ is the set of routes from S_l^{next} to S_{k+j}^{dest}, we predict the time when bag b_{k+j} will arrive at S_{k+j}^{dest} via route r as follows:

$$\hat{t}_{l,r,k+j}^{\text{unload}} = t_{k+j}^{\text{cross}} + \hat{\tau}_{l,k+j}^{\text{link}} + \hat{\tau}_r^{\text{route}} \tag{4}$$

where

- t_{k+j}^{cross} is the time instant (computed by the local prediction model) at which bag b_{k+j} crosses S_s.
- $\hat{\tau}_{l,k+j}^{\text{link}}$ is the time we predict [2] that bag b_{k+j} spends on link l out of S_s. For this prediction we take:

$$\hat{\tau}_{l,k+j}^{\text{link}} = \begin{cases} \max\left(\dfrac{d_l^{\text{link}}}{v^{\max}}, \dfrac{n_{l,k+j}}{\zeta_l}\right) & \text{if link } l \text{ is not jammed} \\ \max\left(\dfrac{d_l^{\text{link}}}{v^{\text{jam}}}, \dfrac{n_{l,k+j}}{\zeta_l}\right) & \text{if link } l \text{ is jammed} \end{cases}$$

  where d_l^{link} is the length of link l out of S_s, $n_{l,k+j}$ is the number of DCVs on link l at time instant t_{k+j}^{cross}, and v^{jam} is the speed to be used in case of jam, $v^{\text{jam}} \ll 1$. We consider link l to be jammed only if $Q_l \geq \alpha Q_l^{\max}$ where Q_l is the capacity of link l at time instant t_{k+j}^{cross}, $Q_l^{\max}$ is its maximum capacity, and α is a weighting parameter determined based on empirical data, $0 < \alpha < 1$.
- $\hat{\tau}_r^{\text{route}}$ is the average travel time on route $r \in \mathscr{R}_{l,k+j}^{\text{next}}$ for an average speed determined based on empirical data.

Then the optimal predicted unloading time instant is defined as follows:

$$\hat{t}_{k+j}^{\text{unload},*} = \underset{\{\hat{t}_{l,r,k+j}^{\text{unload}} | r \in \mathscr{R}_{l,k+j}^{\text{next}}\}}{\arg\min} J_{k+j}(\hat{t}_{l,r,k+j}^{\text{unload}})$$

Local optimization problem So, the MPC optimization problem at junction S_s and bag step k is defined as follows:

$$\begin{aligned} &\min_{\mathbf{u}(k)} J_{k,N}^{\text{DMPC}}(\mathbf{u}(k)) \\ &\text{subject to} \\ &\qquad \mathbf{t} = \mathscr{M}^{\text{local}}(\mathscr{T}, x(t_k), \mathbf{u}(k), \mathbf{v}(k)) \\ &\qquad \mathscr{C}(\tau_{\mathbf{x}}, v^{\max}) \leq 0 \end{aligned}$$

where $\mathscr{M}^{\text{local}}(\mathscr{T}, x(t_k), \mathbf{u}(k), \mathbf{v}(k))$ describes the local dynamics of junction S_s with its incoming and outgoing links, with $\mathbf{x}$ the state of the local system and $\mathbf{v}(k)$ the velocity sequence for each DCV in the local system.

After computing the optimal control, only $u^{\text{sw\_in}}(k+1)$ and $u^{\text{sw\_out}}(k+1)$ are applied. Next the state of the system is updated. At bag step $k+1$, a new optimization will be then solved over the next N bags.

The main advantage of decentralized MPC consists in a smaller computation time than the one needed when using centralized control due to the fact that we now compute for each junction, independently, the solution of a smaller and simplified optimization problem.

5.3 Distributed model predictive control

One may increase the performance of the *decentralized* control proposed above by implementing a *distributed* approach that uses additional communication and coordination between neighboring junctions. Data will be communicated between consecutive levels of influence. A level of influence κ consists of junctions for which we compute the local control independently. Let us now assign levels of influence to each junction in the network. We assign influence level 1 to each junction in the network connected via a direct link to a loading station. Next, we consider all junctions connected by a link to some

[2] If S_s would be directly connected to an unloading station, then $\hat{\tau}_{l,k+j}^{\text{link}} = \frac{d_l^{\text{link}}}{v^{\max}}$ if S_l^{next} is S_{k+j}^{dest}, and $\hat{\tau}_{l,k+j}^{\text{link}} = \tau^{\max}$ if S_l^{next} is S_{k+j}^{dest} with $\tau^{\max}$ a large nonnegative scalar.

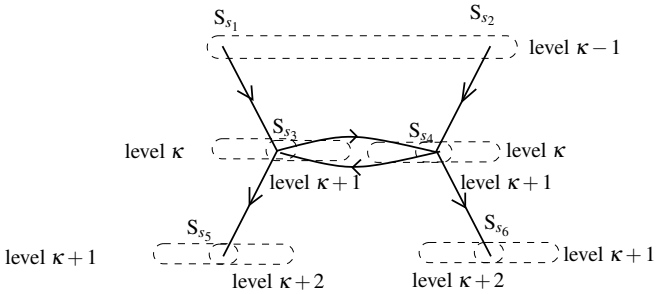


Fig. 5. Levels of influence.

junction with influence level 1, and we assign influence level 2 to them. In that way we recursively assign an influence level to each junction with the constraint that at most 2 influence levels are assigned to a given junction [3] (see Figure 5 where $\{s_1, s_2, s_3, s_4, s_5, s_6\} \subseteq \{1, 2, \ldots, S\}$).

In this section we consider that the communication of the future actions is performed downstream. This means that the local controller of each junction on influence level $\kappa = 1$ solves the local optimal control problem has described in Section 5.2. After computing the optimal switch control sequence, each junction with influence level κ communicates to its neighboring junctions at level $\kappa+1$ which bags (out of all the bags over which we make the prediction for the corresponding junction with influence level κ) will enter the incoming link of the junction at level $\kappa+1$ and at which time instant. Next, we iteratively consider the junctions at levels $\kappa = 2, 3$, etc. until level of influence $\mathscr{K}$, were $\mathscr{K}$ is the largest level of downstream influence assigned in the network. Then, for each junction on influence level $\kappa > 1$, we compute a local solution to the local MPC problem as presented next.

Assume S_s with $s \in \{1, \ldots, S\}$ to have assigned influence level $\kappa > 1$. For the sake of simplicity of notation, in the remainder of this subsection, we will not explicitly indicate the subscript s for variables that refer to junction S_s. Let S_l^{prev} denote the neighboring junction of S_s connected via the incoming link l of S_s (accordingly, S_l^{prev} has assigned influence level $\kappa-1$). Then, we compute a local solution for S_s to the local MPC problem over an horizon of

$$N = \min\left(N^{\max}, \sum_{l=0}^{1}\left(n_l^{\text{horizon}} + \sum_{\ell=0}^{1} N_l\right)\right)$$

bags where N_l is the horizon of the local MPC problem at S_l^{prev}.

Note that in this approach $\mathscr{M}^{\text{local}}(\mathscr{T}, x(t_k), \mathbf{u}(k), \mathbf{v}(k))$ describes the local dynamics of junction S_s with its incoming and outgoing links and additional data from neighboring junctions (if any).

The computation of the local control is performed according to the following algorithm:

Algorithm 2. Distributed computation of local control

1: **for** $\kappa = 1$ to $\mathscr{K}$ **do**
2: compute independently local switching sequences for influence level κ taking into account the control on influence level $\kappa - 1$
3: **end for**

[3] The constraint that at most $\kappa_{\max}$ influence levels are assigned to a junction influences the computational complexity.

Every time some bag has crossed some junction we update the local control of junctions in the network as follows. Assume that some bag has just crossed junction S_s at level κ. Then, we update the control of S_l^{next} at level $\kappa+1$, $S_{l,n}^{\text{next}}$ at level $\kappa+2$, and so on until level $\mathscr{K}$, where $S_{l,n}^{\text{next}}$ is the junction connected to S_l^{next} via the outgoing link $n \in \{0, 1\}$ of S_l^{next}.

Note that the controllers of the junctions on level κ have to wait for the completion of the computation of the switching sequences of the controllers on the previous level before starting to compute their future control action. Therefore, when comparing with decentralized MPC, such distributed MPC may improve the performance of the system, but at the cost of higher computation time due to the required synchronization in computing the control actions.

5.4 Optimization methods

When using centralized MPC, at each bag step k, the future route sequence $r(k+1), r(k+2), \ldots, r(k+N)$ is computed over an horizon of N bags so that $J_{k,N}^{\text{CMPC,tot}}(\mathbf{r}(k))$ is minimized subject to the system's dynamics and the operational constraints. So, the control has an integer representation. Therefore, to solve the optimization problem P_1 one could use e.g. *genetic* algorithms, *simulated annealing*, or *tabu search* (see e.g. Reeves and Rowe (2002), Dowsland (1993), Glover and Laguna (1997)).

Recall that when using decentralized or distributed MPC, the control variables for switch-in and switch-out at junction S_s, represent the positions 0 or 1 that the switch-in and switch-out of S_s should have when the DCV carrying bag i will pass the junction. Hence, also in these cases, the control variable has an integer representation. In order to solve the optimization problem P_2 one can use integer optimization once more.

6. CASE STUDY

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

6.1 Set-up

We consider the network of tracks depicted in Figure 6 with 6 loading stations, 1 unloading station, and 10 junctions. We have considered this network because on the one hand it is simple, allowing an intuitive understanding of and insight in the operation of the system and the results of the control, and because on the other hand, it also contains all the relevant elements of a real set-up.

We assume that the velocity of each DCV varies between 0 m/s and $v^{\max} = 20$ m/s, being controlled by on-board collision avoidance controllers. The lengths of the track segments are indicated in Figure 6.

In order to faster assess the efficiency of our control method we assume that we do not start with an empty network but with a network already populated by DCVs transporting bags.

6.2 Scenarios

For the calibration of the weighting parameters we have defined 27 scenarios, each consisting of a stream of 120 bags.

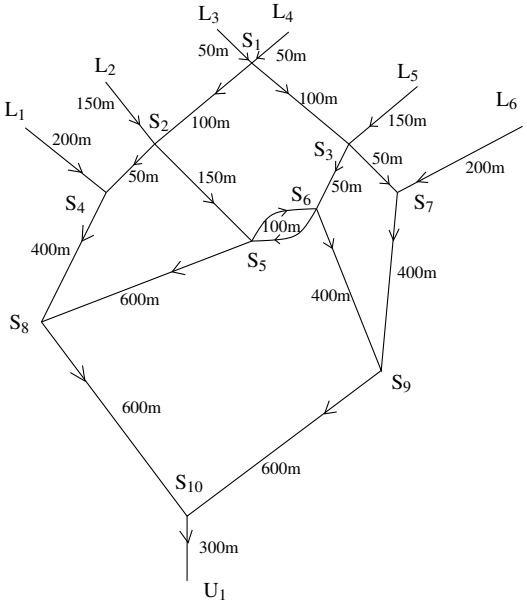


Fig. 6. Case study for a DCV-based baggage handling system.

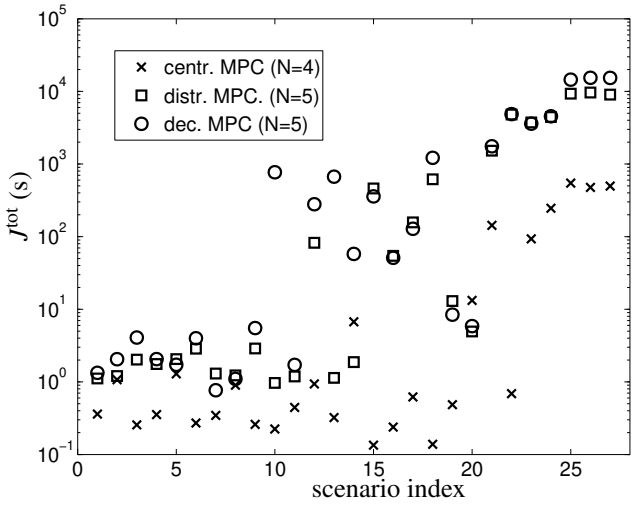
We have also considered different classes of demand profiles at each loading station and different initial states of the system where 60 DCVs evenly distributed on links are already transporting bags in the network, running from loading stations $L_1, L_2, \ldots, L_6$ to junctions S_4, S_2, S_1, S_3, S_7, from S_1 to S_2, and from S_1 to S_3. Their position at t_0 and their static priorities (σ_i) are assigned randomly. In scenarios $1, \ldots, 6$ it is considered that all the bags have to be loaded onto the same plane. In scenarios $7, \ldots, 27$, we consider that the group of bags transported by DCVs through the network before t_0 have to be loaded onto plane A. The rest of the bags have to be loaded onto plane B. Moreover, plane A departs earlier than plane B. Also, in scenarios $1, \ldots, 18$ we analyze the performance of the baggage handling system when the last bag that enters the system can arrive in time at the corresponding end point if the DCV has an average speed of 12 m/s, while in scenarios $19, \ldots, 27$, we examine the situation where the transportation of the bags is very tight (the last bag that enters the system can only arrive in time at the corresponding end point if the shortest path is used and its DCV is continuously running with maximum speed).

6.3 Results

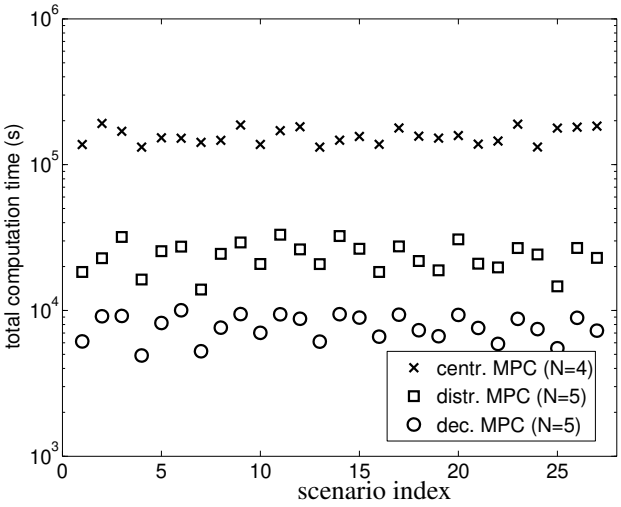
In order to solve the optimization problems of centralized, decentralized, and distributed MPC we have used the *genetic* algorithm of the Matlab optimization toolbox *Genetic Algorithm and Direct Search* implemented via the function `ga` with multiple runs since simulations show that this optimization technique gives good performance, with the shortest computation time. Note that we have used the function `ga` with its default options for *bitstring* population.

Based on simulations we now compare, for the given scenarios, the proposed control methods [4] . In Figure 7 we plot the results obtained when using centralized, decentralized, and respectively distributed MPC. Note that the lower the performance index J^{tot} is, the better the performance of the baggage handling system is.

[4] Recall that when comparing the proposed control approaches we compute the closed loop performance index given by (3).



(a) closed loop performance index



(b) computation time

Fig. 7. Comparison of the proposed control approaches.

Clearly the best performance of the system is obtained when using centralized switch control. However, centralized control becomes intractable in practice when the number of junctions is large due to the large computation time [5] required. The simulations indicate that both decentralized MPC and distributed MPC offer a balanced trade-off between computation time and optimality. However, the results confirm that the communication of the intended control action between neighboring junction may increase the performance of the system, but at the cost of bigger computational effort.

7. CONCLUSIONS AND FUTURE WORK

In this paper we have considered the baggage handling process in large airports using destination coded vehicles (DCVs) running at high speeds on a "mini" railway network. A fast event-driven model of the continuous-time baggage handling process has been determined. In particular we consider the route choice control problem for each DCV transporting bags on the track network. In order to optimize the performance of the system, we have compared three predictive control methods that can be used to route the DCVs through the network. These approaches

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

are centralized, decentralized, and distributed model predictive control (MPC).

The results show that the best performance of the system is obtained using centralized control. Moreover, centralized MPC is not tractable in practice due to the large computational effort that this method requires. Decentralized and distributed MPC offer a balanced trade-off between the optimality and the time required to compute the route for each DCV.

In future work we will analyze more variants of distributed control, where e.g. we combine the downstream optimization with the upstream coordination, and assess the scalability and benefits that can be obtained by using such distributed control. We will also apply the proposed approaches to real-life case studies.

ACKNOWLEDGEMENTS

This research is supported by the VIDI project "Multi-Agent Control of Large-Scale Hybrid Systems" (DWV.6188) 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, by the Delft Research Centre Next Generation Infrastructures, and by the European 7th framework STREP project "Hierarchical and Distributed Model Predictive Control (HD-MPC)" (contract number INFSO-ICT-223854).

REFERENCES

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

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

Dowsland, K. (1993). Simulated annealing. In C. Reeves (ed.), *Modern heuristic techniques for combinatorial problems*, chapter 2, 20–69. John Wiley & Sons, Inc., New York, USA.

Fay, A. (2005). Decentralized control strategies for transportation systems. In *Proceedings of the International Conference on Control and Automation*, 898–903. Budapest, Hungary.

Gang, H., Shang, J., and Vargas, L. (1996). A neural network model for the free-ranging AGV route-planning problem. *Journal of Intelligent Manufacturing*, 7(3), 217–227.

Glover, F. and Laguna, F. (1997). *Tabu Search*. Kluwer Academic Publishers, Norwell, Massachusetts, USA.

Hallenborg, K. and Demazeau, Y. (2006). Dynamical control in large-scale material handling systems through agent technology. In *Proceedings of the International Conference on Intelligent Agent Technology*, 637–645. Hong Kong, China.

Kaufman, D., Nonis, J., and Smith, R. (1998). A mixed integer linear programming model for dynamic route guidance. *Transportation Research Part B: Methodological*, 32(6), 431–440.

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

Reeves, C. and Rowe, J. (2002). *Genetic Algorithms - Principles and Perspectives: A Guide to GA Theory*. Kluwer Academic Publishers, Norwell, Massachusetts, USA.

Siljak, D. (1991). *Decentralized Control of Complex Systems*. Academic Press, INC, San Diego, California.

Weiss, G. (1999). *Multiagent Systems*. The MIT Press Cambridge, London, UK.