Technical report 09-012

Receding horizon approaches for route choice control of automated baggage handling systems[*]

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

If you want to cite this report, please use the following reference instead:
A.N. Tarău, B. De Schutter, and H. Hellendoorn, "Receding horizon approaches for route choice control of automated baggage handling systems," *Proceedings of the European Control Conference 2009*, Budapest, Hungary, pp. 2978–2983, Aug. 2009.

[*]This report can also be downloaded via `https://pub.deschutter.info/abs/09_012.html`

Receding horizon approaches for route choice control of automated baggage handling systems

Alina N. Tarău, Bart De Schutter, and Hans Hellendoorn

Abstract— **State-of-the-art baggage handling systems transport luggage in an automated way using destination coded vehicles (DCVs). These vehicles transport the bags at high speeds on a "mini" railway network. Currently, the networks are simple and the performance of the system is limited. In the research we conduct, more complex networks are considered. In order to optimize the performance of the system we compare several predictive control methods that can be used to route the DCVs through the track network. More specifically, 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.**

I. Introduction

The baggage handling system (BHS) of an airport is one of the most important factors that determine the airport's efficiency and reliability. The first objective of a BHS is to transport all the checked-in or transfer bags to the corresponding end points[1] before the planes have to be loaded. However, due to the airports' logistics, an end point is allocated to a plane at a given time before the departure of the plane. Hence, the BHS performs optimally if each of the bags to be handled arrives at its given end point in a specific time window.

Currently, the fastest way to transport the luggage is to use destination coded vehicles (DCVs). A DCV is a metal cart with a plastic tub on top. These carts transport the bags at high speeds on a "mini" railway network.

Controllers that solve the low-level control problems (such as coordination and synchronization when loading the bags onto a DCV and when unloading a DCV, or velocity control for each DCV such that collisions are avoided) are already present in the system. In this paper we consider higher-level control problems the route choice control of DCVs such that the performance of the system is optimized. Since we need the bags at their end point within a given time window, we cannot use solutions found in literature for solving the shortest-path or shortest-time problems (see e.g. [1], [2]). Also, we do not solve the problem of scheduling and routing automated guided vehicles as in e.g. [3]. In this paper we compare advanced control methods that can be used to solve our routing problem viz. centralized, decentralized, and distributed model predictive control.

The paper is organized as follows. In Section II, a continuous-time event-driven model of the system is presented. Afterwards, in Section III, the global performance index is elaborated. In Section IV, we propose advanced control methods for computing the route of each DCV in a centralized, a decentralized[2], and a distributed[3] manner. The analysis of the simulation results and the comparison of the proposed control methods are elaborated in Section V. Finally, in Section VI, conclusions are drawn and the directions for future research are presented.

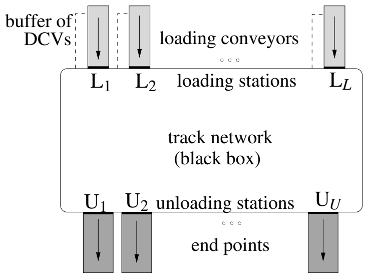


Fig. 1. Baggage handling system using DCVs.

II. Event-driven model

To illustrate our approaches we consider a DCV-based BHS as sketched in Figure 1. This system operates as follows: given a dynamic demand of bags (identified by their unique code) and a buffer of empty DCVs for each loading station, together with the network of tracks, the optimal route of each DCV (from a given loading station to the corresponding unloading station) has to be computed subject to operational and safety constraints such that the performance of the system is optimized.

We consider a BHS with L loading stations and U unloading stations as depicted in Figure 1. Accordingly, we define the L-tuple $\mathscr{T} = (\mathbf{t}_{\text{arrival},1}, \mathbf{t}_{\text{arrival},2}, \ldots, \mathbf{t}_{\text{arrival},L})$ that

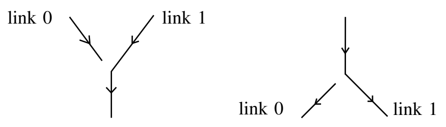
All authors are with Faculty of Mechanical, Maritime and Materials Engineering, Delft Center for Systems and Control, Delft University of Technology, Mekelweg 2, 2628 CD Delft, The Netherlands a.n.tarau@tudelft.nl, b@deschutter.info, j.hellendoorn@tudelft.nl

Bart De Schutter is also with the Marine and Transport Technology department of Delft University of Technology.

[1]An end point is the place where the bags are lined up, waiting to be loaded onto the plane.

[2]If the local control actions are computed without any communication or coordination between the local controllers, the control approach is said to be decentralized.

[3]If the local control actions are computed considering also communication and coordination between neighboring controllers, then the control approach is said to be distributed.



(a) switch-in (b) switch-out

Fig. 2. Incoming and outgoing links at a junction. The switch-in and switch-out are positioned on link 1.

comprises the sequences of bag arrival times $\mathbf{t}_{\text{arrival},\ell} = [t_{\text{arrival},\ell,1} \cdots t_{\text{arrival},\ell,N_\ell}]^{\text{T}}$ with $\ell \in \{1,2,\dots,L\}$ and N_ℓ the number of bags that will arrive at loading station L_ℓ during the entire simulation period. We also consider that the track network has S junctions S_1, S_2,..., S_S. Note that without loss of generality we can assume that each junction has maximum 2 incoming links and maximum 2 outgoing links, both indexed by $l \in \{0,1\}$ as sketched in Figure 2. Each junction has a switch going into the junction (called switch-in hereafter) and a switch going out of the junction (called switch-out hereafter).

There are five types of events that can occur:

- loading a new bag into the system
- unloading a bag that arrives at its end point
- updating the position of the switch-in
- updating the position of the switch-out
- updating the speed of a DCV.

The model of the BHS is an event-driven one consisting of a continuous part describing the movement of the individual vehicles transporting the bags through the network, and of the discrete events listed above.

Let X be the number of bags that BHS has to handle and X_{crt} the 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. Let DCV_i denote the DCV that transports the ith bag that entered the track network up to the current time instant, $i \leq X_{\text{crt}}$. Assuming that the velocity of each DCV is piecewise constant, the model of the BHS is given by the algorithm below.

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 $t_{\text{load},i}$ denote the time instant when the ith bag that entered the track network is loaded onto a DCV and let $t_{\text{unload},i}$ denote the time instant when the same bag is unloaded at the corresponding end point. Consequently, we denote the model of the BHS as $\mathbf{t} = \mathscr{M}(\mathscr{T}, x(t_0), \mathbf{u})$, where $\mathbf{t} = [t_{\text{load},1} \cdots t_{\text{load},X}\ t_{\text{unload},1} \cdots t_{\text{unload},X}]^{\text{T}}$, $x(t_0)$ is the initial state of the system (i.e. position of switches and position and speed of DCVs at time instant t_0), and $\mathbf{u}$ is the route control sequence.

Algorithm 1. Model of the BHS

```
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:        $\tau_{\text{load},\ell} \leftarrow$ time that will pass until the next loading event of $\text{L}_\ell$
5:    end for
6:    for $\ell = 1$ to $U$ do
7:        $\tau_{\text{unload},\ell} \leftarrow$ time that will pass until the next unloading event of $\text{U}_\ell$
8:    end for
9:    for $s = 1$ to $S$ do
10:       $\tau_{\text{switch\_in},s} \leftarrow$ time that will pass until the next switch-in[4] event at $\text{S}_s$
11:       $\tau_{\text{switch\_out},s} \leftarrow$ time that will pass until the next switch-out[5] event at $\text{S}_s$
12:   end for
13:   for $i = 1$ to $X_{\text{crt}}$ do
14:       $\tau_{\text{speed\_update},i} \leftarrow$ time that will pass until the next speed-update event of $\text{DCV}_i$
15:   end for
16:   $\tau_{\min} \leftarrow \min( \min_{\ell=1,\dots,L} \tau_{\text{load},\ell}, \min_{\ell=1,\dots,U} \tau_{\text{unload},\ell}, \min_{s=1,\dots,S} \tau_{\text{switch\_in},s}, \min_{s=1,\dots,S} \tau_{\text{switch\_out},s}, \min_{i=1,2,\dots,X_{\text{crt}}} \tau_{\text{speed\_update},i})$
17:   $t_{\text{crt}} \leftarrow t_{\text{crt}} + \tau_{\min}$
18:   take action (i.e. load, unload, switch-in update, switch-out update, speed-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.

The system obeys to operational constraints derived from the mechanical and design limitations of the system such as:

C_1: the speed of each DCV is bounded between 0 and $v_{\max}$

C_2: the position of a switch at a junction can change after minimum $\tau_{\text{x}} > 0$ time units in order to avoid chattering

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

III. Global performance index

In this section is defined the global performance index J that will be used in this paper.

Since the BHS performs successfully if all the bags are transported to the corresponding 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 $J_{\text{pen},i}$ corresponding to the ith bag that entered the network, $i \in \{1,2,\dots,X\}$, is to penalize the overdue time and the additional storage time. Accordingly, as sketched in Figure 3, we define the following penalty for DCV_i:

$$J_{\text{pen},i}(t) = \sigma_i \max(0, t - t_{\text{load\_plane},i}) + \lambda_1 \max(0, t_{\text{load\_plane},i} - \tau_{\text{max\_storage},i} - t) \quad (1)$$

where $t_{\text{load\_plane},i}$ is the time instant when the end point closes and the bags are loaded onto the plane, σ_i is the static priority of the bag on DCV_i (the flight priority), and $\tau_{\text{max\_storage},i}$

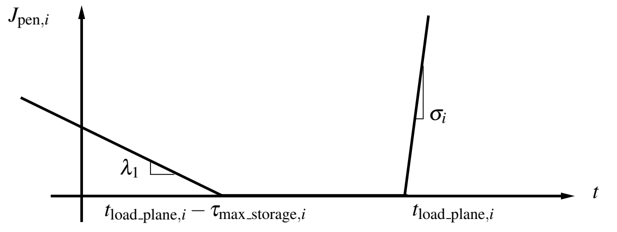


Fig. 3. Objective function $J_{\text{pen},i}$.

is the maximum possible time window for which the end point corresponding to DCV$_i$ is open for that specific flight. The weighting parameter $\lambda_1 \leq 1$ represents the relative cost between buying additional storage space at the end points and the cost of customers that have their baggage delayed.

In order to minimize the energy consumption we also include the dwell time. Then we obtain the following objective function:

$$J_i(t) = J_{\text{pen},i}(t) + \lambda_2(t - t_{\text{load},i}) \tag{2}$$

where λ_2 is a small weight factor ($\lambda_2 \ll \lambda_1$).

The final performance index is given by $J_{\text{tot}} = \sum_{i=1}^{X} J_i(t_{\text{unload},i})$. Note that the objective function $J_i(t_{\text{unload},i})$ depends on the arrival time of DCV$_i$ at its corresponding end point, and implicitly it depends on the routes of all the X bags to be handled.

IV. Control approaches

In this section we determine the route of each DCV transporting a bag. 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. These collision avoidance controllers ensure a minimum safe distance between DCVs and also hold DCVs at switching points, if required.

A. Centralized model predictive control

Model predictive control (MPC) is an on-line model-based predictive control design method (see e.g [4], [5], [6]). In the basic MPC approach, given a 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 $[t_k, t_k + T_{\text{s}}N]$, where $t_k = t_0 + kT_{\text{s}}$ with T_{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 $J_{\text{CMPC},k,N}(\mathbf{r}(k)) = \sum_{i=1}^{k+N} J_i(\hat{t}_{\text{unload},i})$ where $\hat{t}_{\text{unload},i}$ 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:

$$\begin{aligned}
\text{P}_1\!: \quad & \min_{\mathbf{r}(k)} J_{\text{CMPC},k,N}(\mathbf{r}(k)) \\
& \text{subject to} \\
& \quad \mathbf{t} = \mathscr{M}(\mathscr{T}, x(t_k), \mathbf{r}(k)) \\
& \quad \mathscr{C}(v_{\text{max}}, \tau_{\text{x}}) \leq 0
\end{aligned}$$

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 V. 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.

B. Decentralized model predictive control

In decentralized model predictive route choice control we consider local subsystems, each consisting of a junction S$_s$ with $s \in \{1, 2, \ldots, S\}$, its incoming and its outgoing links. 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$_s$.

Our prediction model is a simulation of the local system. In the prediction model we index the bags that successively cross S$_s$ during the entire simulation period $[t_0, t_0 + T_{\text{max}}]$ as $b_1, b_2, \ldots, b_{N_{\text{bags}}}$, where N_{bags} is the number of bags that cross S$_s$ during $[t_0, t_0 + T_{\text{max}}]$. The optimization is performed at every junction S$_s$ over the next $N \leq N_{\text{bags}}$ bags that pass the junction. Note that when we solve the local optimization problem we consider only the bags that travel on the incoming links of S$_s$ at the current time instant.

Every time a bag has crossed the junction we update the local control. So, assume that bag b_k has crossed S$_s$. The time instant at which this happens is denoted by $t_{\text{cross}\,\text{S}_s, b_k}$. For the sake of simplicity of notation, we will not explicitly include the time argument when specifying the control laws and related variables since they always refer to $t_{\text{cross}\,\text{S}_s, b_k}$. Next we compute the control sequence[6] $\mathbf{u}(k) = [u_{\text{sw\_in}}(k+1) \ldots u_{\text{sw\_in}}(k+N)\, u_{\text{sw\_out}}(k+1) \ldots u_{\text{sw\_out}}(k+N)]^{\text{T}}$ corresponding to the next N bags $b_{k+1}, b_{k+2}, \ldots, b_{k+N}$ that will

[6]For junctions with only one incoming link we have $\mathbf{u}(k) = [u_{\text{sw\_out}}(k+1) \ldots u_{\text{sw\_out}}(k+N)]^{\text{T}}$, while for junctions with only one outgoing link we have $\mathbf{u}(k) = [u_{\text{sw\_in}}(k+1) \ldots u_{\text{sw\_in}}(k+N)]^{\text{T}}$.

cross the junction by solving an optimization problem. 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 in which the bags cross the junction and the corresponding time when the bags $b_{k+1},\ldots,b_{k+N}$ enter S_s. The control decision $u_{\text{sw\_out}}(k+j)$ for $j=1,2,\ldots,N$ of the switch out of S_s influences the route that bag b_{k+j} will take.

When solving the local MPC optimization problem, we will use a local performance index $J_{\text{DMPC},k,N}$. The local performance index is computed via a simulation of the local system for the next N bags that will cross the junction. This goes as follows. At bag step $k\geq 0$, the initial state of the local system consists of the positions of the DCVs traveling in the local system and the positions of the switch-in and switch-out of S_s at the time instant when bag k crossed the junction (if $k=0$ we consider the initial state of the local system at time instant t_0). Next, at bag step k, we compute the release rate of each outgoing link of S_s. The computation of the release rate is required due to the fact that we use a local simulation as prediction. Recall that the outgoing links of S_s are indexed by $l\in\{0,1\}$. Then let n_l denote the number of DCVs that left the outgoing link l within the time window $[t_{\text{cross}\,\mathrm{S}_s,b_k}-\tau_{\text{q}},t_{\text{cross}\,\mathrm{S}_s,b_k}]$, of length τ_{q} time units. Then the fixed release rate of link l for the prediction model at bag step k is given by $\zeta_{l,k}=\dfrac{n_l}{\tau_{\text{q}}}$. However, for links that connect S_s with unloading stations, the release rate is by definition unbounded. The control of the switch-out $u_{\text{sw\_out}}(k+j)$ with $j\in\{1,2,\ldots,N\}$ represents the position of switch-out when bag b_{k+j} will cross S_s, determining the next junction towards which bag b_{k+j} will travel. Let us call this junction $\mathrm{S}_{\text{next},k+j}$ (note that in fact $\mathrm{S}_{\text{next},k+j}$ is a function of $u_{\text{sw\_out}}(k+j)$). For each possible route $r\in\mathscr{R}_{\text{next},k+j}$, where $\mathscr{R}_{\text{next},k+j}$ is the set of routes from $\mathrm{S}_{\text{next},k+j}$ to the corresponding end point of bag b_{k+j}, we estimate the time that the DCV transporting bag b_{k+j} needs to reach its end point via route r as follows:

$$\hat{t}_{\text{unload},k+j,r}=t_{\text{leave local system},k+j}+\tau_{\text{approx},r}$$

where $t_{\text{leave local system},k+j}$ is the time instant (predicted by the local simulation model) at which bag b_{k+j} leaves the link $\mathrm{S}_s\to\mathrm{S}_{\text{next},k+j}$ for the release rate $\zeta_{u_{\text{sw\_out}}(k+j),k}$, and $\tau_{\text{approx},r}$ is the static time period that the DCV carrying bag b_{k+j} would need to travel the route $r\in\mathscr{R}_{\text{next},k+j}$ with $v_{\max}$. Then the local objective function $J_{\text{DMPC},k,N}(\mathbf{u}(k))$ is defined as:

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

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

So, in decentralized route choice MPC, at step k, where k is the number of DCVs passing junction S_s, the future control sequence of the switches $\mathbf{u}(k)$ is computed by solving an optimization problem over a horizon of N bags traveling on the incoming links of S_s so that the local performance function $J_{\text{DMPC},k,N}(\mathbf{u}(k))$ is optimized subject to the operational constraints. Accordingly, the MPC optimization problem at junction S_s and bag step k is defined as follows:

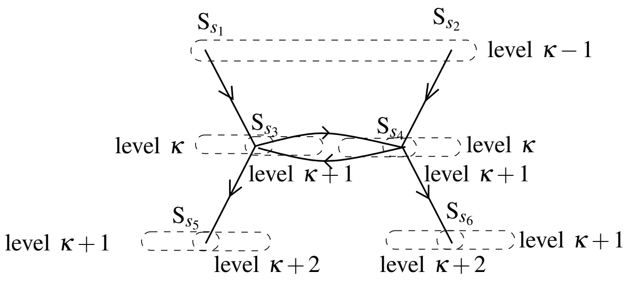


Fig. 4. Levels of parallel computation.

$$\begin{aligned}
\mathrm{P}_2:\quad & \min_{\mathbf{u}(k)} J_{\text{DMPC},k,N}(\mathbf{u}(k))\\
& \text{subject to}\\
& \qquad \mathbf{t}=\mathscr{M}((\mathscr{T},x(t_k),\mathbf{u}(k)))\text{ when considering only}\\
& \qquad\qquad \mathrm{S}_s\text{ with its incoming and outgoing links}\\
& \qquad \mathscr{C}(v_{\max},\tau_{\text{x}})\leq 0
\end{aligned}$$

After computing the future 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 in parallel the solution of a smaller and simplified optimization problem.

C. 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 in parallel. 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 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[7] (see Figure 4 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 P_2 as described in Section IV-B. Furthermore, for each junction on the same influence level $\kappa>1$, the intended switch control sequence of the junctions on influence level $\kappa-1$ is communicated. So, the junctions on influence level κ use as additional information the expected arrival time of the bags sent from influence level $\kappa-1$. Then we compute for each junction on influence level κ the local solution of P_2 over a horizon

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

of N bags traveling on the incoming links of the junction or coming from the neighboring junctions on influence level $\kappa - 1$.

The computation of the local control is performed according to the following algorithm where $\mathscr{K}$ is the largest level of influence assigned in the network.

Algorithm 2. Distributed computation of local control

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

Every time a bag crosses a junction we update the local control of all junctions. Recall 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 and iteration in computing the control actions.

D. 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 by solving P_1 over a horizon of N bags so that $J_{\text{tot CMPC},N}(\mathbf{r}(k))$ 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* [7], [8], [9].

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.

V. Case study

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

A. Set-up

We consider the network of tracks depicted in Figure 5 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 5.

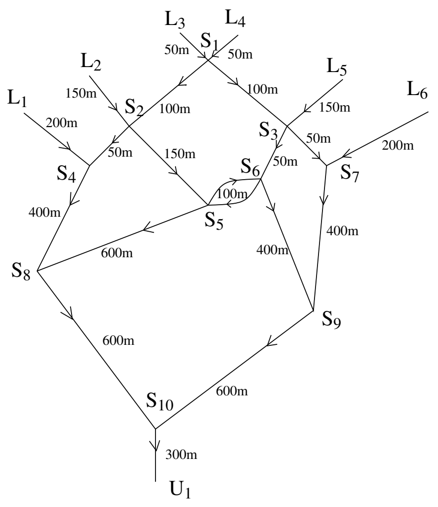


Fig. 5. Case study for a DCV-based BHS.

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.

B. Scenarios

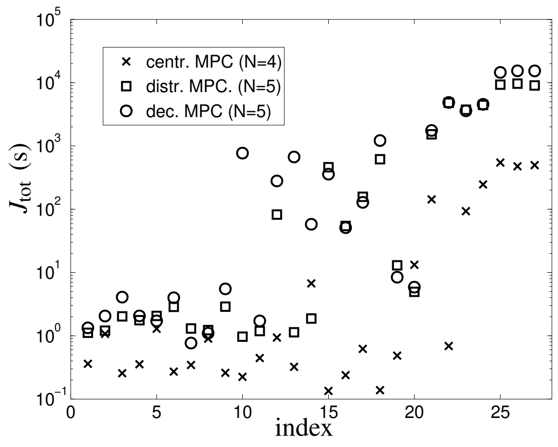
For the calibration of the weighting parameters we have defined 27 scenarios, each consisting of a stream of 120 bags.

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 BHS 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).

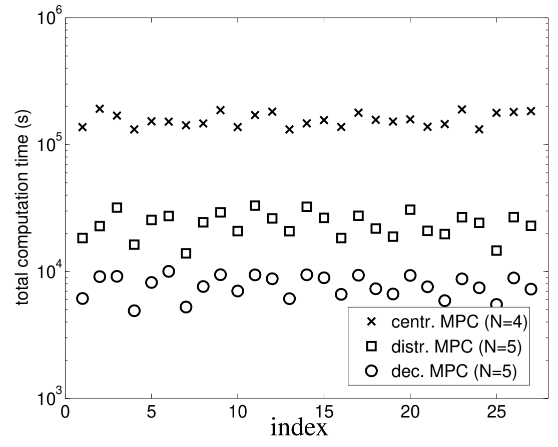
C. Results

To solve the optimization problems P_1 and P_2 we have chosen 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.

Based on simulations we now compare, for the given scenarios, the proposed control methods. In Figure 6 we plot



(a) closed loop performance



(b) computation time

Fig. 6. Comparison of the proposed control approaches.

the results obtained when using centralized, decentralized, and respectively distributed MPC.

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[8] 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.

VI. 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 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.

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

Acknowledgment

This research is supported by the STW-VIDI project "Multi-Agent Control of Large-Scale Hybrid Systems", the BSIK project "Next Generation Infrastructures" (NGI), 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).

References

[1] H. Gang, J. Shang, and L. Vargas, "A neural network model for the free-ranging AGV route-planning problem," *Journal of Intelligent Manufacturing*, vol. 7, no. 3, pp. 217–227, 1996.

[2] D. Kaufman, J. Nonis, and R. Smith, "A mixed integer linear programming model for dynamic route guidance," *Transportation Research Part B: Methodological*, vol. 32, no. 6, pp. 431–440, 1998.

[3] L. Qiu, W.-J. Hsu, S.-Y. Huang, and H. Wang, "Scheduling and routing algorithms for AGVs: a survey," *International Journal of Production Research*, vol. 40, no. 3, pp. 745–760, 2002.

[4] J. Maciejowski, *Predictive Control with Constraints*. Harlow, UK: Prentice Hall, 2002.

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

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

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

[8] K. Dowsland, "Simulated annealing," in *Modern heuristic techniques for combinatorial problems*, C. Reeves, Ed. New York, USA: John Wiley & Sons, Inc., 1993, ch. 2, pp. 20–69.

[9] F. Glover and F. Laguna, *Tabu Search*. Norwell, Massachusetts, USA: Kluwer Academic Publishers, 1997.