Delft University of Technology

Delft Center for Systems and Control

Technical report 09-035

Distributed route choice control in DCV-based 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, "Distributed route choice control in DCV-based baggage handling systems," *Proceedings of the 18th IEEE International Conference on Control Applications*, Saint Petersburg, Russia, pp. 818–824, July 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_035.html`

Distributed route choice control in DCV-based baggage handling systems

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

Abstract— **In this paper we develop advanced control methods for routing individual vehicles which ensure automatic transportation of bags in a baggage handling system of an airport. In particular we consider distributed model predictive control and a distributed heuristic control approach. The baggage handling system performs efficiently if all the bags are transported to the corresponding end points within a specific time window, and this makes the process of handling baggage time-critical. The proposed control approaches are effective for the given application. To assess their performance we consider a benchmark case study, in which the methods are compared for several scenarios. Results indicate that the distributed approaches improve the performance of the vehicle-based baggage handling system with up to** 20%**.**

I. Introduction

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, low-level controllers ensure the 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 the corresponding end point. Low-level controllers also compute the velocity of the DCVs such that collisions are avoided. The networks on which the DCVs run are simple and, therefore, the performance of these systems is limited. In the research we conduct more complex networks are considered. The aim of this work is to optimally route the DCVs in case of dynamic demand.

For applications such as automated guided vehicles route planning or traffic route guidance, the route assignment problem has been addressed in e.g. [1], [2]. But, in our case we do not deal with a shortest-path or shortest-time problem, since we need the bags at their corresponding end points within a given time window. An attempt to solve the routing problem of DCVs transporting bags using an analogy of how data are transmitted via internet is presented in [3], but without presenting any experimental results. Also, in [4], a multi-agent approach for the control software of a DCV-based baggage handling system is presented. However, this multi-agent system is faced with major challenges due to the extensive communication required. Therefore, the goal of our work is to develop and compare control approaches for route choice control for each DCV.

Theoretically, the maximum performance of a DCV-based baggage handling system is obtained if one computes the optimal routes using optimal control [5]. However, as we have shown in [6], this control method becomes intractable in practice due to the heavy computation burden. Therefore, in order to make a trade-off between computational effort and optimality, in [7], we have also implemented centralized and decentralized[1] model predictive control, and also a decentralized heuristic approach. As the results confirmed, centralized model predictive control requires high computation time to compute a solution. The use of decentralized predictive control lowers the computation time, but at the cost of suboptimality. Finally, we have seen that the decentralized heuristic approach needs very low computation time to calculate a solution, but usually the results are worse than those obtained when using decentralized predictive control. In this paper we investigate whether the performance of the system can be increased by using additional communication and coordination between neighboring junctions when computing the control. Hence, we now develop and implement distributed[2] control approaches viz. distributed model predictive control and a distributed heuristic approach. The heuristic approach uses rules to determine the position of the switches leading into and out of a junction. These rules depend on the static and dynamic priorities of the bags transported by DCVs on the incoming links, the optimal path to destination, and the current position of each switch.

The paper is organized as follows. In Section II, we present a continuous-time event-driven model that we have developed together with the performance index that describes the efficiency of a DCV-based baggage handling system and a general description of model predictive control. Afterwards, in Section III, we develop several distributed predictive control methods for computing the route of each DCV transporting a bag with various degrees of complexity. Due to the large computation effort that these approaches require, in Section IV we also propose distributed heuristics. The analysis of the simulation results and the comparison of the proposed control methods are elaborated in Section V. Finally, Section VI draws the conclusions of this paper.

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

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

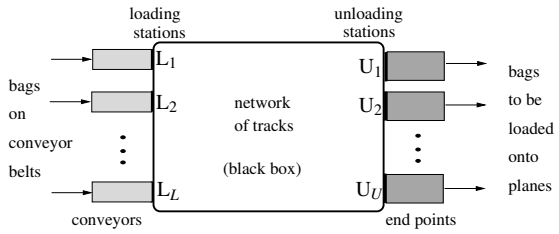


Fig. 1. Baggage handling system using DCVs.

II. Background

A. System description and model

In this paper we use the general DCV-based baggage handling system sketched in Figure 1. This system operates as follows: given a demand of bags and the network of tracks, the route of each DCV (from a given loading station to the given unloading station) has to be computed subject to operational and safety constraints such that the performance of the system is optimized.

The model of the baggage handling system we have developed in [6] consists of a continuous part describing the movement of the individual vehicles transporting the bags through the network, and of the following discrete events: loading a new bag onto a DCV, unloading a bag that arrives at its end point, updating the position of the switch into a junction (called switch-in hereafter), and the position of the switch going out of a junction (called switch-out hereafter), and updating the speed of a DCV. The state of the system consists of the positions of the DCVs in the network and the positions of the switch-ins and switch-outs of the junctions. According to the discrete-event model we have developed in [6], as long as there are bags to be handled, given the current state of the system, we shift the current time to the next event time, take the appropriate action, and update the state of the system.

The operational constraints derived from the mechanical and design limitations of the system are the following: the speed of each DCV is bounded between 0 and $v_{\max}$, while a switch at a junction can only change its position after minimum τ_x time units in order to avoid the quick and repeated back and forth movements of the switch, which may lead to mechanical damage.

B. Global performance index

In this section we define the global performance index that will be used in this paper.

On the one hand the baggage handling system performs successfully if all the bags are transported to their end point before a given time instant, so, the overdue time has to be minimized. On the other hand, due to the airport's logistics, an end point is allocated to a plane only a given amount of time before the departure of the plane. Hence, one way to construct the objective function corresponding to bag i is to penalize the overdue time and the additional storage time. Accordingly, we define the following penalty for bag i:

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

where $t_{\text{end},i}$ 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 i (the flight priority), and $\theta_{\text{max\_storage},i}$ is the maximum possible time window for which the end point of bag 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 for bag i:

$$J_i(t) = J_{\text{pen},i}(t) + \lambda_2 (t_{\text{dwell},i}) \quad (2)$$

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

The final performance index is given by $J_{\text{tot}} = \sum_{i=1}^{N_{\text{bags}}} J_i(t_{\text{unload},i})$, where N_{bags} is the number of bags to be handled and $t_{\text{unload},i}$ is the time instant when bag i is unloaded at its corresponding end point.

C. Model predictive control

Since later on we will be using the concept of model predictive control (MPC), in this section we briefly present this control method.

MPC is an on-line model-based predictive control design method [8] that uses the receding horizon principle. 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 $[t_k, t_k + T_s N]$, where $t_k = t_0 + kT_s$ with T_s the sampling time, so that a cost criterion defined over the period $[t_k, t_k + T_s N]$ is optimized subject to the operational constraints. 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, a feedback mechanism is introduced.

In the next section, we define a variant of MPC, where k is not a time index, but a bag index. Also, the horizon N corresponds to the number of bags that we consider for prediction.

III. Distributed model predictive control

In order to determine the route of each DCV transporting a bag, in this section, we first propose several distributed predictive control methods. Note 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.

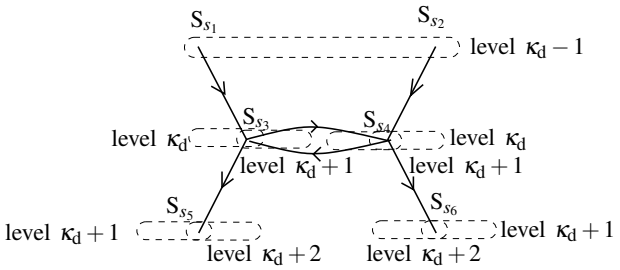


Fig. 2. Levels of parallel computation.

A. Levels of influence

In distributed 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. Distributed MPC uses communication and coordination between neighboring junctions. Data will be communicated between consecutive levels of influence. A level of influence consists of junctions for which we compute the local control in parallel.

Let us first assign levels of *downstream* influence to each junction in the network. We assign downstream 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 $\kappa_{\mathrm{d,max}}$ downstream influence levels are assigned to a given junction[3]. For example see Figure 2 where we define maximum 2 levels of downstream influence for each junction $\{s_1,s_2,s_3,s_4,s_5,s_6\} \subseteq \{1,2,\ldots,S\}$ with S the number of junctions in the network.

Similarly we can also assign levels of *upstream* influence to each junction in the network. We assign upstream influence level 1 to each junction in the network connected via a direct link to an unloading station. Next, we assign upstream influence level 2 to all the junctions connected by a link to some junction on upstream influence level 1. Recursively, we then assign levels of upstream influence to each junction with the constraint that at most $\kappa_{\mathrm{u,max}}$ levels of upstream influence are assigned to a given junction.

B. Distributed MPC with a single iteration of downstream communication

Let us now consider distributed MPC with a single iteration of downstream communication. This means that the local controller of each junction on influence level $\kappa_d = 1$ solves the local optimal control problem depending on the current traffic in the local subsystem and the demand of bags at loading stations. Furthermore, for each junction on the same influence level $\kappa_d > 1$, the intended switch control sequence of the junctions on influence level $\kappa_d - 1$ is communicated. So, the junctions on influence level κ_d use as additional information the expected arrival time of the bags sent from influence level $\kappa_d - 1$. Then, for each junction on influence level κ_d, we compute a local solution to the local MPC problem (see Section III-D for the exact definition of the local MPC problem) over a horizon of N bags —the bags are already traveling on the incoming links of the junction or coming from the neighboring junctions on influence level $\kappa_d - 1$.

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

Algorithm 2. Distributed computation of local control — a single iteration of downstream communication

1: **for** $\kappa_d = 1$ to $K_{\mathrm{downstream}}$ **do**
2: compute in parallel local switching sequences for influence level κ_d taking into account the control on influence level $\kappa_d - 1$
3: **end for**

For simplicity we update the local control of all the junctions in the network every time a bag has crossed a junction. Note that the controllers of the junctions on level κ_d 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.

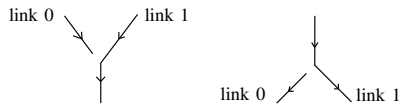
C. Distributed MPC with a single iteration of downstream and upstream coordination

Now we add an extra round of coordination and consider distributed MPC with a single iteration of downstream and upstream coordination. This method involves the following steps. Every time a bag has crossed a junction we first compute the local control sequences according to the downstream levels of influence. Then we identify the junctions on the last level of downstream influence $K_{\mathrm{downstream}}$ that are connected to unloading stations (these are junctions on level 1 of upstream influence). For these junctions we update the release rate corresponding to their incoming links. Then we compute the local control of all junctions on the upstream level $\kappa_u > 1$ taking into account the updated release rates and the intended control known from the downstream iteration. The computation of the local control is performed according to the following algorithm where K_{upstream} is the largest level of influence assigned in the network. The local MPC problem is defined in Section III-D

Algorithm 3. Distributed computation of local control — a single iteration of downstream and upstream coordination

1: **for** $\kappa_d = 1$ to $K_{\mathrm{downstream}}$ **do**
2: compute in parallel local switching sequences for influence level κ_d taking into account the local control on downstream influence level $\kappa_d - 1$
3: **end for**

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



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

Fig. 3. Incoming and outgoing links at a junction. The switch-in and switch-out are positioned on link 1.

4: **for** $\kappa_u = 1$ to $K_{upstream}$ **do**
5: compute in parallel local switching sequences for influence level κ_u taking into account the local control on upstream influence level $\kappa_u - 1$ and the updated release rate
6: **end for**

By performing also the upstream optimization more information about the future congestion is provided via the updated release rate. This information might change the initial intended control actions of each junction. Therefore, this new variant of distributed MPC may increase the performance of the system by ensuring more coordination between the neighboring junctions, but also the computational effort increases.

D. Local MPC problem

In this section we define the local MPC problem.

1) Local system boundaries: We consider that each local systems consists of a junction S_s with $s \in \{1,2,\ldots,S\}$, its maximum 2 incoming and 2 outgoing links as sketched in Figure 3. 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 since we refer to one junction only. For all the other junctions, the same procedure is applied.

2) Local control: As noted in Section II we do not use a time index, but a bag index. So, we control the positions of the switch-in and switch-out of junction S_s for each bag that crosses S_s. Recall that we update the local control every time some bag has just crossed a junction. Let t_{crt} denote a time instant at which the local controls are updated. For junction S_s we now determine bag step k such that $t_{cross,k} \leq t_{crt} < t_{cross,k+1}$, where $t_{cross,k}$ is defined as the time instant when bag b_k has just crossed the junction.

We index the bags that successively cross a junction S_s during the entire simulation period as $b_1, b_2, \ldots, b_{N_{bags}}$, where N_{bags} is the number of bags that cross S_s during the simulation period. The local optimization is performed over the next $N \leq N_{bags}$ bags that will pass junction S_s at bag step k. By solving this local optimization problem we compute the control sequence[4] $\mathbf{u}(k) = [u_{sw\_in}(k+1) \ldots u_{sw\_in}(k+N)\, u_{sw\_out}(k+1) \ldots u_{sw\_out}(k+N)]^T$ corresponding to the next N bags $b_{k+1}, b_{k+2}, \ldots, b_{k+N}$ that will cross the junction. The control decisions $u_{sw\_in}(k+1), \ldots, u_{sw\_in}(k+N)$ of the switch into S_s determine the order in which the bags cross the junction and the corresponding time instants at which the bags $b_{k+1}, \ldots, b_{k+N}$ enter S_s. The control decisions $u_{sw\_out}(k+1), \ldots, u_{sw\_out}(k+N)$ determine the next junction towards which the bag $b_{k+1}, \ldots, b_{k+N}$ will travel.

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

3) Local objective function: When solving the local MPC optimization problem, we will use a local objective function $J_{DMPC,N}$. 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:

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

where $\hat{t}^*_{unload,k+j}$ is the estimated unloading time instant of bag b_{k+j}. Next we present how the estimated unloading time instant is computed.

4) Prediction model: Our prediction model is a simulation of the local system.

Given the state of the local system at t_{crt} we compute the release rate of each outgoing link l for $l = 0, 1$. The computation of the release rate is required due to the fact that we use a local simulation as prediction. Let n_l denote the number of DCVs that left the outgoing link l within the time window $[t_{crt} - \tau_q, t_{crt}]$, of length τ_q time units. Then the fixed release rate of link l which will be used during the entire prediction period at bag step k is given by $\zeta_l = \frac{n_l}{\tau_q}$. However, for links that connect S_s with unloading stations, the release rate is by definition unbounded.

Let $S_{next,l}$ where $l = u_{sw\_out}(k+j)$ denote the junction that bag b_{k+j} will cross next, and let $S_{dest,k+j}$ denote the corresponding end point of bag b_{k+j}.

For each possible route $r \in \mathscr{R}_{next,k+j,l}$, where $\mathscr{R}_{next,k+j,l}$ is the set of routes from $S_{next,l}$ to $S_{dest,k+j}$, we estimate the time when bag b_{k+j} will arrive at $S_{dest,k+j}$ via route r as follows:

$$\hat{t}_{unload,k+j,l,r} = \hat{t}_{cross,k+j} + \hat{\tau}_{link\,l,k+j} + \hat{\tau}_{route\,r}$$

where

- $\hat{t}_{cross,k+j}$ is the estimated time instant (computed by the local prediction model) at which bag $k+j$ crosses S_s.
- $\hat{\tau}_{link\,l,k+j}$ is the time we estimate that bag b_{k+j} spends on link l out of S_s. For this estimation we take:

$$\hat{\tau}_{link\,l,k+j} = \begin{cases} \max\left(\frac{d_l}{v_{max}}, \frac{N_{k+j,l}}{\zeta_l}\right) & \text{if } Q_l < \alpha Q_{max,l} \\ \max\left(\frac{d_l}{v_{jam}}, \frac{N_{k+j,l}}{\zeta_l}\right) & \text{otherwise} \end{cases}$$

  where d_l is the length of the traveled link, v_{jam} is the speed to be used in case of jam (determined using empirical data), $N_{k+j,l}$ is the number of DCVs on the link at the time instant $\hat{t}_{cross,k+j}$, Q_l and $Q_{max,l}$ are the flow and respectively the maximum capacity of the link, and α is a weighting parameter.
- $\hat{\tau}_{route\,r}$ is the average travel time on route $r \in \mathscr{R}_{k+j,next,l}$ for an average speed $v_{avg,route\,r}$ determined based on historical data.

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

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

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

$$\min_{\mathbf{u}(k)} J_{\text{DMPC},N}(\mathbf{u}(k))$$
subject to
the local dynamics of the S_s with
its incoming and outgoing links and
additional data from neighboring junctions
operational constraints

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.

Since the optimization problem above involves integer variables, to solve it one has to use *integer* optimization algorithms such as *genetic* algorithms or *tabu search* see e.g. [9], [10].

IV. Distributed heuristic approach

In order to lower the computation time of the previous control methods, in this Section we present a distributed heuristic approach. Now the local control sequence is only one sample in contrast to distributed MPC where we have computed local control sequences of N samples. The control variables of this approach represent the time span after which the switch-in and respectively switch-out will change position.

Note that since the control of all local systems is similar we only refer to the control of junction S_s with $s \in \{1,2,\ldots,S\}$. Consequently, for the sake of simplicity of notation, we will not explicitly include the subscript s in the remainder of this section.

A. Local heuristic control of the switch-in

The local control of the switch-in is updated every time a bag crosses the neighboring junction $S_{\text{prev},l}$ with $l \in \{0,1\}$ connected to S_s via the incoming l. Let t_{crt} denote this time instant. Then each switch is locally controlled based on heuristic rules as presented next. The local switch control is determined based on both local information —the incoming and outgoing links of junction S_s— and additional data regarding the flow of DCV on the incoming links of $S_{\text{prev},l}$ for $l = 0,1$. This is an extension of the heuristic approach that we have developed in [6]. In [6] the local control is determined based on local information only.

For a junction S_s, we define the following variables:

- Γ_l is the set of bags transported by DCVs that travel on the incoming link $l \in \{0,1\}$ of junction S_s at the time instant t_{crt}.
- Ω_l is the set of bags that will cross $S_{\text{prev},l}$ traveling towards S_s in the next $\tau_{\text{prediction}}$ time units. For determining Ω_l we use a prediction of the system where the control of the switch-in and switch-out is determined using the heuristic approach we have developed in [6].
- a_l is total static priority of link l, $a_l = \sum_{i \in \Gamma_l} \sigma_i$.
- b_l is the total dynamic priority of link l, $b_l = \sum_{i \in \Gamma_l} \frac{\hat{\delta}_i}{\delta_{\max,i}}$ with $\hat{\delta}_i$ the estimate of the actual time bag i requires to get from its current position to its final destination in case of no congestion and maximum speed, and $\delta_{\max,i}$ the maximum time left for bag i to spend in the system while still arriving at the plane on time. If bag i misses the flight, then the bag has to wait for a new plane with the same destination. Hence, a new departure time is assigned to bag i, and consequently a new loading time $t_{\text{new\_end},i}$ for bag i is considered. Then the variable $\delta_{\max,i}$ is defined as $\delta_{\max,i} = t_{\text{end},i} - t_{\text{crt}}$ if $t_{\text{end},i} - t_{\text{crt}} > 0$ and $\delta_{\max,i} = t_{\text{new\_end},i} - t_{\text{crt}}$ if $t_{\text{end},i} - t_{\text{crt}} \leq 0$.
- c_l is total static priority of the bags in Ω_l.
- d_l is the total dynamic priority of the bags in Ω_l.

In order to determine the next position of the switch-in at junction S_s we compute the performance measure $p_{\text{sw\_in},l}$ for $l = 0,1$ every time a new bag enters the incoming link l. This performance measure takes into account the static and dynamic priorities of the bags transported by DCVs on the link l, and the current[5] position of the switch-in at junction S_s:

$$\begin{aligned} p_{\text{sw\_in},0} &= w_{\text{st\_pr}}(a_0 + c_0) + w_{\text{dyn\_pr}}(b_0 + d_0) - w_{\text{sw\_in}}\tau_{\text{x}} I_{\text{crt}} \\ p_{\text{sw\_in},1} &= w_{\text{st\_pr}}(a_1 + c_1) + w_{\text{dyn\_pr}}(b_1 + d_1) - \\ &\quad w_{\text{sw\_in}}\tau_{\text{x}}(1 - I_{\text{crt}}) \end{aligned}$$

where I_{crt} denotes the current position of the switch-in at junction S_s (i.e. $I_{\text{crt}} = 0$ if the switch-in is positioned on the incoming link 0 and $I_{\text{crt}} = 1$ if the switch-in is positioned on the incoming link 1). The weighting parameters $w_{\text{st\_pr}}$, $w_{\text{dyn\_pr}}$, and $w_{\text{sw\_in}}$ are calibrated as explained in Section 4.3 of [6].

Let $z_l \in \Gamma_l$ indicate the bag closest to S_s on incoming link l. The variable d_{z_l} denotes the distance between the current position of bag z_l and S_s while v_{z_l} denotes the current speed of DCV transporting bag z_l. Then we define the time period $\tau_{\text{arrival},l}$ that the DCV transporting bag z_l needs to travel the distance d_{z_l} in case of no speed-update events as $\tau_{\text{arrival},l} = \frac{d_{z_l}}{v_{z_l}}$ if $d_{z_l} > 0$, and $\tau_{\text{arrival},l} = 0$ if $d_{z_l} = 0$.

The position of the switch-in at S_s is toggled only if $p_{\text{sw\_in},0} > p_{\text{sw\_in},1}$ and $I_{\text{crt}} = 1$ or if $p_{\text{sw\_in},1} > p_{\text{sw\_in},0}$ and $I_{\text{crt}} = 0$. If this is the case, then the current position of the switch-in is changed after $\tau_{\text{sw\_in}} = \max(\tau_{\text{x}} - \tau_{\text{sw\_in\_prev}}, \tau_{\text{arrival},1-I_{\text{crt}}})$ time units where $\tau_{\text{sw\_in\_prev}}$ is the time for which the switch-in at junction S_s has been in its current position.

B. Local heuristic control of the switch-out

Every time when a bag is at junction S_s we compute the control variable $\tau_{\text{sw\_out}}$ which represents the time period until the position of the switch-out has to be changed.

[5]The current position of the switch-in is considered due to the operational constraint according to which the position of a switch at a junction can only change after minimum τ_{x} time units.

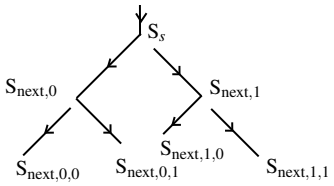


Fig. 4. $v_{max} = 2$ links connected out of S_s at $t_{cross,i}$.

Assume that bag i is at junction S_s. Let $t_{cross,i}$ denote the time instant when this happens. First we estimate the time that bag i needs to travel on the next maximum[6] v_{max} links when trying to reach the corresponding destination.

As sketched in Figure 4, for $v_{max} = 2$, let $S_{next,l,m}$ for $m = 0,1$ denote the neighboring junction of $S_{next,l}$ connected via link m out of $S_{next,l}$ and let $N_{new\,DCV,i,l,m}$ denote the number of DCVs on link l out of S_s that will choose link m out of $S_{next,l}$.

We assume that for a junction $S_{next,l}$, $l \in \{0,1\}$ with 2 outgoing links, half of the DCVs traveling from S_s to $S_{next,l}$ take link $m = 0$ out of $S_{next,l}$, and the other half take link $m = 1$. Then the time period that bag i needs to travel link m out of $S_{next,l}$ considering the release rate $\zeta_{l,m}$ of link m out of $S_{next,l}$ is defined as:

$$\hat{\tau}_{link\,l,m,i} = \begin{cases} \max\left(\dfrac{d_{l,m}}{v_{max}}, \dfrac{N_{DCV}}{\zeta_{l,m}}\right) & \text{if } Q_{l,m} < \alpha Q_{max,l,m} \\ \max\left(\dfrac{d_{l,m}}{v_{jam}}, \dfrac{N_{DCV}}{\zeta_{l,m}}\right) & \text{otherwise} \end{cases}$$

where $d_{l,m}$ is the length of the link m out of $S_{next,l}$, $N_{DCV} = N_{DCV,i,l,m} + N_{new\,DCV,i,l,m}$ with $N_{DCV,i,l,m}$ the number of DCVs on this link at the time instant when bag i crosses junction S_s.

Let $\mathscr{R}_{next,i,l,m}$ with $l \in \{0,1\}$ and $m \in \{0,1\}$ denote the set of routes from junction $S_{next,l,m}$ to $S_{dest,i}$. In this case, for each route $r \in \mathscr{R}_{next,i,l,m}$ we estimate the time $\hat{t}_{unload,i,l,m,r}$ when bag i will reach $S_{dest,i}$ if the bag takes link l out of S_s, link m out of $S_{next,l}$, and route r. This time is given by:

$$\hat{t}_{unload,i,l,m,r} = t_{cross,i} + \hat{\tau}_{link\,l,i} + \hat{\tau}_{link\,l,m,i} + \frac{d_r}{v_{avg}}$$

where d_r is the length of route r.

The estimated arrival time $J_i(\hat{t}^*_{unload,i,l,m,r})$ that optimizes the objective function of bag i when choosing link $m \in \{0,1\}$ out of $S_{next,l}$ and route $r \in \mathscr{R}_{next,i,l,m}$ is defined as follows:

$$\hat{t}^*_{unload,i,l} = \underset{\{\hat{t}_{unload,i,l,m,r} | m \in \{0,1\} \wedge r \in \mathscr{R}_{next,i,l,m}\}}{\arg\min} J_i(\hat{t}_{unload,i,l,m,r})$$

Then we compute the cost criterion $c_{sw\_out,i,l}$ for $l = 0,1$ that takes into account $\hat{t}^*_{unload,i,l,m,r}$ and the current position O_{crt} of the outgoing switch:

$$c_{sw\_out,i,0} = w_{pen} J_i(\hat{t}^*_{unload,i,0,r}) + w_{sw\_out}\tau_x O_{crt}$$
$$c_{sw\_out,i,1} = w_{pen} J_i(\hat{t}^*_{unload,i,1,r}) + w_{sw\_out}\tau_x (1 - O_{crt})$$

[6]We look only at the next maximum v_{max} links in order to get some extra information on the network congestion state, while keeping the communication requirements low.

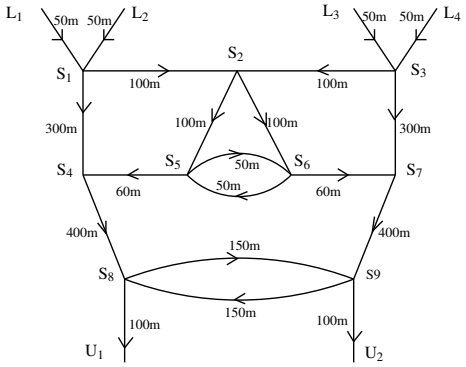


Fig. 5. Case study for a DCV-based baggage handling system.

The weighting parameters w_{pen} and w_{sw_out} can be calibrated as explained in the [6].

The position of the switch-in at junction S_s is toggled only if $c_{sw\_out,i,0} < c_{sw\_out,i,1}$ and $O_{crt} = 1$ or if $c_{sw\_out,i,1} < c_{sw\_out,i,0}$ and $O_{crt} = 0$. If this is the case, then the switch-out is toggled after $\tau_{sw\_out} = \max(0, \tau_x - \tau_{sw\_out\_prev})$ where $\tau_{sw\_out\_prev}$ is the time for which the switch-out at junction S_s has been in its current position.

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 six loading stations, two unloading stations, and nine junctions. This network is considered 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 applied control approaches, 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 20 m/s, the lengths of the track segments being indicated in Figure 5.

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

We have considered typical scenarios[7] with different classes of demand profiles for each loading station, different initial states of the system, congestion on different links, and different time criticality measures. We first consider six scenarios where transporting the bags via the shortest routes involves the need of additional storage. Afterwards we consider six more scenarios where the transportation of the bags is very tight, i.e. the last bag that enters the system can only arrive in time at the corresponding end point if the

[7]For comparing the control methods we have used the same scenarios, but different samples of the demand profiles than those considered for calibrating the weighting parameters.

TABLE I

COMPARISON OF RELATIVE IMPROVEMENT OF THE PERFORMANCE AND COMPUTATION TIME FOR THE PROPOSED CONTROL METHODS.

Control approach	relative improvement (%)	CPU time (s)
Decentralized MPC ($N = 3$)	14.3	$2.67 \cdot 10^3$
Distributed MPC downstream ($N = 3$)	18.6	$4.85 \cdot 10^3$
Distributed MPC back & forth ($N = 3$)	22.9	$1.16 \cdot 10^4$
Decentralized HR	0	0.08
Distributed HR ($\tau_{\text{prediction}} = 5\,\text{s}$)	19.4	128

shortest path is used and its DCV is continuously running with maximum speed.

C. Results

To solve the MPC optimization problems we have chosen a *genetic* algorithm 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 relative to their decentralized variants. This goes as follows. For all control methods we compute the average cost criterion over all scenarios, $J_{\text{avg,control approach}} = \frac{1}{N_{\text{scenario}}} \sum_{j=1}^{N_{\text{scenario}}} J_{j,\text{control approach}}$. Afterwards the control approach that results in maximum average cost criterion is considered to have 0 % improvement. The improvement of the rest of the control methods that we consider in this paper are computed relative to this cost criterion.

The preliminary results of the simulations are reported in Table I. These preliminary results confirm that distributed approaches improve the performance of the system relative to the decentralized approaches, but at the cost of higher computation time[8]. The improvement appears due to the additional information regarding future congestion. However, estimating future states requires additional computation time. Note that the performance of the heuristics approaches could be improved by performing more extensive tuning. The high computational effort required by the predictive approaches can also be lowered by e.g. parallelizing the computation of the local control, using faster computers, implementing the control methods in object coded programming language, or reducing the time allowed for solving the local MPC optimization at the cost of suboptimality.

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. In particular we consider the route choice control problem for each DCV transporting bags on the track network. Using a fast event-driven model of the continuous-time baggage handling process determined in previous work, in this paper we develop several distributed model predictive control (MPC) methods and a distributed heuristic approach to compute effective routing. In contrast to the decentralized control methods developed in previous work for the same purpose, the distributed approaches use additional communication and coordination between local controllers to compute a routing solution. The preliminary results confirm that distributed approaches improve the performance of the system relative to the decentralized approaches, but at the cost of higher computational effort. However, the computational time can be lowered by e.g. parallelizing even more the computation of the local control. In future work we will also consider reducing the computational effort by approximating the model with a linear one using mixed integer linear programming (MILP) theory. The solution of the MILP optimization could then be used as an initial starting point for the original nonlinear optimization problem.

[8]The simulations were performed in *Matlab*, 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", 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 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] A. Fay, "Decentralized control strategies for transportation systems," in *Proceedings of the International Conference on Control and Automation*, Budapest, Hungary, June 2005, pp. 898–903.

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

[5] F. Lewis, *Optimal Control*. New York, USA: John Wiley & Sons, Inc., 1986.

[6] A. Tarău, B. De Schutter, and J. Hellendoorn, "Travel time control of destination coded vehicles in baggage handling systems," in *Proceedings of the IEEE International Conference on Control Applications*, San Antonio, Texas, USA, Sept. 2008, pp. 293–298.

[7] ——, "Route choice control of automated baggage handling systems," in *Proceedings of the 88th Annual Meeting of the Transportation Research Board*, Washington DC, USA, Jan. 2009.

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

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

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