Delft University of Technology

Delft Center for Systems and Control

Technical report 10-055

Hierarchical route control in DCV-based 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, "Hierarchical route control in DCV-based baggage handling systems," *International Journal of Services Operations and Informatics*, vol. 6, no. 1/2, pp. 5–29, Jan. 2011.

Delft Center for Systems and Control
Delft University of Technology
Mekelweg 2, 2628 CD Delft
The Netherlands
phone: +31-15-278.51.19 (secretary)
fax: +31-15-278.66.79
URL: `http://www.dcsc.tudelft.nl`

*This report can also be downloaded via `http://pub.deschutter.info/abs/10_055.html`

Hierarchical route control in DCV-based baggage handling systems

A.N. Tarău*

Delft Center for Systems and Control,
Delft University of Technology,
Delft, The Netherlands
Fax: 31-40-2461418/2474850
E-mail: a.n.tarau@tue.nl *Corresponding author

B. De Schutter

Delft Center for Systems and Control,
Delft University of Technology,
Delft, The Netherlands
Fax: 31-15-2786679/2785113
E-mail: b@deschutter.info

J. Hellendoorn

Delft Center for Systems and Control,
Delft University of Technology,
Delft, The Netherlands
Fax: 31-15-2786679/2789007
E-mail: j.hellendoorn@tudelft.nl

Abstract: State-of-the-art baggage handling systems transport the baggage at high speeds, on a network of tracks, using destination coded vehicles (DCV). In order to ensure the optimal routing of DCVs, in this paper we propose a hierarchical control framework. In this framework switch controllers provide position instructions for each switch in the network. The switch controllers are then supervised by a so-called network controller that mainly takes care of flows of DCVs. The routing control problem for the network controller is a nonlinear, mixed integer optimization problem, with high computational requirements, which makes it intractable in practice. Therefore, we present an alternative approach for reducing the complexity of the computations by approximating the nonlinear optimization problem and rewriting it as a mixed integer linear programming (MILP) problem. The advantage is that for MILP problems solvers are available that allow us to efficiently compute the global optimal solution. The solution of the MILP problem is then used for computing optimal switch control actions. For a benchmark case study we compare the hierarchical route control with switch control approaches that have been developed previously. Results indicate that the proposed hierarchical control offers a balanced trade-off between optimality and computational efficiency.

Keywords: Baggage handling systems, DCV route control, hierarchical control.

Reference to this paper should be made as follows: Tarău, A.N., De Schutter, B. and Hellendoorn, J. (2011) 'Hierarchical route control in DCV-based baggage handling systems', *Int. J. of Services Operations and Informatics*, Vol. 6, Nos. 1/2, pp. 5–29.

1 Introduction

The state-of-the-art technology used by baggage handling systems at airports to transport the bags in an automated way incorporates:

1. scanners that scan the (electronic) baggage tags on each piece of luggage,
2. baggage screening equipment for security scanning,
3. networks of conveyors equipped with junctions that route the bags through the system,
4. destination coded vehicles (DCVs). These vehicles are used in large airports only, where the distances between the check-in desks and the end points towards which the baggage has to be transported are too large (for these airports the conveyor systems are too slow, and therefore, a faster carrier is required for each bag).

As illustrated in Figure 1, a DCV is a metal cart with a plastic tub on top. These carts are propelled by linear induction motors mounted on the tracks. The DCVs transport the bags at high speed on a network of tracks.

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 low-level control problem. Low-level controllers determine the velocity of each DCV so that a minimum safe distance between DCVs is ensured and so that the DCVs are held at switching points, if required. So, a DCV runs at maximum speed, $v^{\max}$, unless overruled by the local on-board collision avoidance controller (for more details see Section 3). Other 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 (an end point of the baggage handling system is the final part of the system where the bags are lined up, waiting to be loaded into containers and from there to the plane). Note that we assume the low-level controllers already present in the system.

Figure 1 DCVs running on a network of tracks. Photo courtesy of Vanderlande Industries.

In the remainder of this paper we focus on higher-level control problems of a DCV-based baggage handling system. Currently, the track networks on which the DCVs transport the baggage have a simple structure, with the loaded DCVs being routed through the system using routing schemes based on preferred routes. These routing schemes adapt to respond to the occurrence of predefined events. However, the load patterns of the system are highly variable, depending on, e.g., the season, time of the day, type of aircraft at each gate, or the number of passengers for each flight (de Neufville, 1994). Also note that the first objective of a baggage handling system is to transport all the checked-in or transfer bags to the corresponding end points before the planes have to be loaded. However, due to the airport's logistics, an end point is allocated to a plane only within given span time before the plane's departure. Hence, the baggage handling system performs optimally if each of the bags to be handled arrives at its given end point within a specific time window. So, predefined routes are far from optimal. Therefore, in this paper we will not consider predefined preferred routes, but instead we will develop and compare efficient control methods to determine the optimal routing in case of dynamic demands.

In the literature, the route assignment problem has been addressed to a large extent for automated guided vehicles (AGVs), see e.g., (Taghaboni and Tanchoco, 1995; Langevin et al., 1996). Typically the AGV routing problem is written as an integer programming problem. Hence, the computational complexity increases exponentially with the number of vehicles to be routed. But in baggage handling systems the number of DCVs used for transportation is large (typically airports with DCV-based baggage handling systems have more than 700 DCVs). Also, 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 routing problem for a DCV-based baggage handling system has been presented by, e.g., Fay (2005) where an analogy to data transmission via internet is proposed, and by, e.g., Hallenborg and Demazeau (2006) where a multi-agent hierarchy has been developed. However, the analogy between routing DCVs through a track network and transmitting data over internet has limitations, see (Fay, 2005), while Hallenborg and Demazeau (2006) do not focus on control approaches for computing the optimal route of DCVs, but on designing a multi-agent hierarchy for baggage handling systems and analyzing the communication requirements. But the multi-agent system of Hallenborg and Demazeau (2006) is faced with major challenges due to the extensive communication required. The goal of our work is to develop and compare *efficient* control approaches for controlling the route of each DCV on the track network.

Theoretically, the maximum performance of such a DCV-based baggage handling system would be obtained if one computes the optimal routes using optimal control (Lewis, 1986). However, as shown by Tarău et al. (2008), 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 (Tarău et al., 2009a) and in (Tarău et al., 2009b), we have developed

and compared *centralized*, *decentralized*, and *distributed* predictive control approaches (MPC), and *decentralized* and *distributed* heuristic approaches to determine the routes of loaded DCVs — if the local control actions are computed without any communication or coordination between the local controllers, the control approach is said to be *decentralized*; if the local control actions are computed considering also communication and coordination between the local controllers, the control approach is said to be *distributed*. In (Tarău et al., 2008) the results confirmed that centralized MPC requires a high computation time to determine a solution. Furthermore, in (Tarău et al., 2009a) we have shown that the use of decentralized control approaches lowers the computation time, but at the cost of suboptimality. Moreover, the results of Tarău et al. (2009b) indicate that the distributed approaches typically improve the performance of the system when compared to decentralized methods, but at the cost of larger total computation time due to the required synchronization when computing the control actions.

The goal of this work is therefore to develop a control approach that will offer a good trade-off between the computational effort required to compute the optimal routing and the total performance of the system. To this aim we propose a hierarchical control framework where the higher-level controllers use MPC. Note that the large computation time obtained in previous work comes from solving the nonlinear, nonconvex, mixed integer optimization problems. Typically, such problems have multiple local minima; are NP hard, and therefore, difficult to solve. So, in this paper we investigate whether the computational effort required by the optimal routing approaches developed so far can be lowered by using mixed integer linear programming (MILP) since for MILP problems efficient solvers are available. Moreover, we expect that using a hierarchical route control framework can improve the efficiency of the routing approaches previously developed.

The paper is organized as follows. In Section 2 we briefly introduce the DCV-based baggage handling systems. Next, in Section 3 we propose the hierarchical control framework that we will use to determine the optimal routing of loaded DCVs. Furthermore, in Section 4, we focus on the routing tasks of the network controller and we present a simplified nonlinear flow model for the DCV-based baggage handling system that can be recast as an MILP model, and the corresponding predictive routing problems for both the nonlinear and the MILP models. In the hierarchical approach the resulting optimal flows become targets to be achieved by optimal switch control. In Section 5 we briefly present how the switch controller computes its control signals. Then, for a benchmark case study we compare the results obtained when using the proposed hierarchical control framework and the switch control approaches that have proved to give good performance in (Tarău et al., 2009b) and (Tarău et al., 2009a): centralized MPC, distributed MPC with a single round of downstream and upstream communication, and distributed heuristics. The analysis of the simulation results is reported in Section 6. Section 7 concludes the paper.

2 System description and model

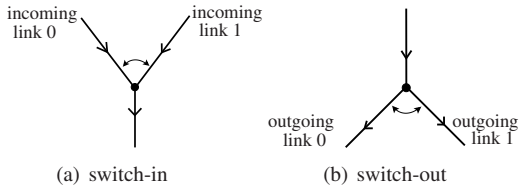


Figure 2 Incoming and outgoing links at a junction.

The track network of a DCV-based baggage handling systems consists of a set of loading stations as origin nodes, a set of unloading stations as destination nodes, and a set of junctions as internal nodes. Note that without loss of generality we can assume each junction to have maximum 2 incoming links and maximum 2 outgoing links (as illustrated in Figure 2). This assumption of a network corresponds to current practice in state-of-the-art baggage handling systems. Let us call the switch that makes the connection between a junction and its incoming links *switch-in*, and the switch that makes the connection between a junction and its outgoing links *switch-out*. Note that a switch-in is required only if the junction has 2 incoming links, otherwise the connection between the one incoming link and the junction is fixed. A similar remark is valid for a switch-out.

Consider the general DCV-based baggage handling system with L loading stations and U unloading stations sketched in Figure 4.

The DCV-based baggage handling system operates as follows: given a demand of bags and the network of tracks as a directed graph, the route of each DCV in the network has to be determined subject to the operational and safety constraints detailed in (Tarău et al., 2008) such that all the bags to be handled arrive at their end points within the corresponding time window. Note that in this paper we focus on optimally routing the loaded DCVs — from a loading station (origin) to an unloading station (destination). As a consequence, we assume that 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.

The model of the baggage handling system we have developed in (Tarău et al., 2008) 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,

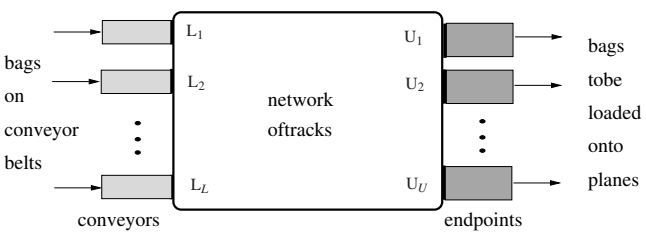


Figure 4 Baggage handling system using DCVs.

- unloading a bag that arrives at its end point,
- updating the position of the switches into and out of a junction,
- updating the speed of a DCV.

According to the discrete-event model of Tarău et al. (2008), as long as there are bags to be handled, the system evolves as follows: we shift the current time to the next event time, take the appropriate action, and update the state of the system. The **state** of the DCV-based baggage handling system consists of the link on which each of the DCVs travel, their speed and their position on that link, and the position of the switch-in and switch-out at each junction. Furthermore, the **input** of the system consists of the demand of bags together with their arrival times at the loading stations and of the control variables. Note that, depending on the control method, the **control variables** can be the switch positions or the time periods after which we toggle the position of the switch as presented later on. Finally, the **output** of the system consists of the time instants when we load and unload each of the bags to be handled (these time instants will be collected into a vector denoted by **t**; they will be derived via simulation and will be used later on when measuring the performance of the system). The event-driven model presented above will later on be used in computing optimal switch movements (for more details see Section 5).

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 has to wait at least τ^{switch} time units between two consecutive toggles in order to avoid the quick and repeated back and forth movements of the switch, which may lead to mechanical damage. These operational constraints are taken into account when choosing the sampling time for the proposed controllers.

3 Proposed control framework

In order to efficiently compute the route of each DCV we propose a hierarchical control framework that consists of a multi-level control structure as shown in Figure 3.

The layers of the framework can be characterized as follows:

- The *network controller* considers flows of DCVs instead of individual DCVs. Moreover, the network controller determines reference DCV flow trajectories over time for each link in the network. These flow trajectories are computed so that the performance of the DCV-based baggage handling system is optimized. Then the optimal reference flow trajectories are communicated to switch controllers.
- The *switch controller* present in each junction receives the information sent by the network controller and determines the sequence of optimal positions for its ingoing and outgoing switches at each time step so that the tracking error between the reference flow trajectory and the actual flow trajectory is minimal.
- The *DCV controller* present in each vehicle detects the speed and position of the vehicle in front of it, if any, and the position of the switch into the junction the DCV travels towards to. This information is then used to determine the speed to be used next such that no collision will occur and such that the DCV stops in front of a junction the switch of which is not positioned on the link that the DCV travels on.

The lower-levels in this hierarchy deal with faster time scales (typically in the milliseconds range for the DCV controllers up to the seconds range for the switch controllers), whereas for the higher-level layer (network controller) the frequency of updating is up to the minutes range.

In (Tarău et al., 2009b) and (Tarău et al., 2009a) we have developed heuristic and predictive control methods for a 2-level control framework that consists of switch controllers and DCV controllers only. These switch controllers determine optimal switch positions and consequently "optimal" routes by solving local optimization problems or by using heuristic rules. In the remainder of the paper we will focus on the network control level of the hierarchy illustrated in Figure 3 and in particular on how the optimal routes can be determined for the DCVs transporting bags through the network.

4 Route control

In this section we focus on the network controller.

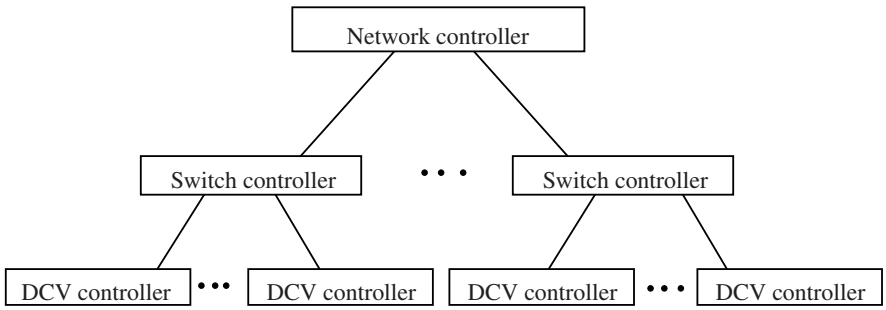


Figure 3 Hierarchical control for DCV-based baggage handling systems. Recall that we assume the low-level DCV controllers already present in the system.

4.1 Preliminaries

Since later on we will use the model predictive control (MPC) approach for determining the routes of the DCVs in the network, in this section we briefly introduce the basic MPC concept.

MPC is an on-line model-based predictive control design method for discrete time models, see, e.g., (Camacho and Bordons, 1995), (Maciejowski, 2002), (Rawlings and Mayne, 2009), that uses the receding horizon principle. In the basic MPC approach, given an horizon N, at step time 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 prediction period $[k\tau_{\rm s}, (k+N)\tau_{\rm s})$ with $\tau_{\rm s}$ the sampling time. The optimization problem is defined so that a cost criterion is optimized over the prediction period 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 step $k+1$ is solved using this new information. In this way, a feedback mechanism is introduced.

4.2 Approach

If we would consider each DCV individually, the predictive switch control problem in DCV-based baggage handling systems results in a huge nonlinear integer optimization problem with high computational complexity and requirements, making the problem in fact intractable in practice (Tarău et al., 2009a). So, since considering each individual switch is too computationally intensive we will consider streams of DCVs instead (characterized by real-valued demands and flows expressed in vehicles per second). The routing problem will then be recast as the problem of determining the flows on each link. Once these flows are determined, they can be implemented by switch controllers at the junctions. So, the network controller provides flow targets to the switch controllers, which then have to control the position of the switch into and out of each junction in such a way that these targets are met as well as possible. This corresponds to blocking flows before a junction whenever necessary and possible, and routing the DCVs towards the outgoing links.

4.3 Set-up

We consider the following set-up. We have a transportation network with a set of origin nodes $\mathscr{O}$ consisting of the loading stations, a set of destination nodes $\mathscr{D}$ consisting of the unloading stations, and a set of internal nodes $\mathscr{I}$ consisting of all the junctions in the network, see Figure 5. We define the set of all nodes as $\mathscr{V} = \mathscr{O} \cup \mathscr{I} \cup \mathscr{D}$. The nodes are connected by unidirectional links. Let $\mathscr{L}$ denote the set of all links.

Furthermore, let the time instant t_k be defined as

$$t_k = t_0 + k\tau^{\rm nc}$$

with t_0 that time when we start the simulation and $\tau^{\rm nc}$ the sampling time for the network controller. Then, for each

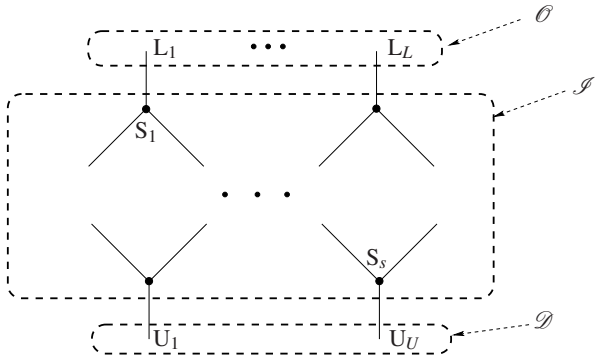


Figure 5 Set-up for the DCV-based baggage handling system. The transportation network has a set of origin nodes $\mathscr{O} = \{\mathrm{L}_1, \mathrm{L}_2, \ldots, \mathrm{L}_L\}$, a set of destination nodes $\mathscr{D} = \{\mathrm{U}_1, \mathrm{U}_2, \ldots, \mathrm{U}_U\}$, and a set of S internal nodes $\mathscr{I} = \{\mathrm{S}_1, \mathrm{S}_2, \ldots, \mathrm{S}_S\}$.

pair $(o,d) \in \mathscr{O} \times \mathscr{D}$, there is a dynamic, piecewise constant demand pattern $D_{o,d}(\cdot)$ as shown in Figure 6 with $D_{o,d}(k)$ the demand of bags at origin o with destination d in the time interval $[t_k, t_{k+1})$ for $k = 0, 1, \ldots, K^{\rm sim}-1$ with $K^{\rm sim}$ the simulation horizon (we assume that beyond $t_{K^{\rm sim}}$ the demand is 0).

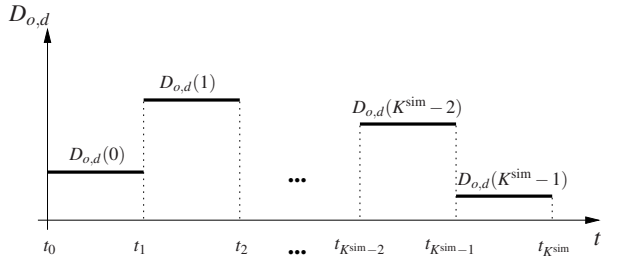


Figure 6 Piecewise constant demand profile $D_{o,d}$.

Next, let $\mathscr{L}_d$ be the set of links that belong to some route going to destination d, $\mathscr{L}_d \subseteq \mathscr{L}$. We denote the set of incoming links for node $v \in \mathscr{V}$ by $\mathscr{L}_v^{\rm in}$, and the set of outgoing links of v by $\mathscr{L}_v^{\rm out}$. Note that for origins $o \in \mathscr{O}$ we have $\mathscr{L}_o^{\rm in} = \emptyset$ and for destinations $d \in \mathscr{D}$ we have $\mathscr{L}_d^{\rm out} = \emptyset$. Also, assume each origin node to have only one outgoing link and each destination node to have only one incoming link — if a loading station would have more than one outgoing link, then one can virtually expand a loading station into a loading station connected via a link of length 0 to a junction with a switch-out and 2 outgoing links, etc.; similarly, one can virtually expand an unloading station with more than one incoming link. Then $|\mathscr{L}_o^{\rm out}| = 1$ and $|\mathscr{L}_d^{\rm in}| = 1$.

Next, for each destination $d \in \mathscr{D}$ and for each link $\ell \in \mathscr{L}_d$ in the network we will define a real-valued flow $u_{\ell,d}(k)$. The flow $u_{\ell,d}(k)$ denotes the number of DCVs per time unit traveling towards destination d that enter link ℓ during the time interval $[t_k, t_{k+1})$.

The aim is now to compute using MPC, for each time step k, flows $u_{\ell,d}(k)$ for every destination $d \in \mathscr{D}$ and for every link $\ell \in \mathscr{L}_d$ in such a way that the capacity of the links is not exceeded and such that the performance criterion is minimized over a given prediction period $[t_k, t_{k+N})$. Later on we will write a model of the baggage handling system

to be used by the network controller, and show that this model can be rewritten as an MILP model. Therefore, in order to obtain an MILP optimization problem one has to define a linear or piecewise affine performance criterion. Possible goals for the network controller that allow linear or piecewise affine performance criteria are reaching a desired outflow at destination d or minimizing the lengths of the queue in the network.

4.4 Model

We now determine the model for the DCV flows through the network. Let τ_ℓ denote the free-flow travel time on link ℓ. Note that the free-flow travel time of link ℓ represents the time period that a DCV requires to travel on link ℓ when using maximum speed. In this subsection we assume the travel time τ_ℓ to be an integer multiple of τ^{nc}, say

$$\tau_\ell = \kappa_\ell \tau^{\mathrm{nc}} \quad \text{with } \kappa_\ell \text{ an integer.} \tag{1}$$

In case the capacity of a loading station is less than the demand, queues might appear at the origin of the network. Let $q_{o,d}(k)$ denote the length at time instant t_k of the partial queue of DCVs at origin o going to destination d. In principle, the queue lengths should be integers as their unit is "number of vehicles", but we will approximate them using reals.

For every origin node $o \in \mathscr{O}$ and for every destination $d \in \mathscr{D}$ we now have:

$$u_{\ell,d}(k) \leqslant D_{o,d}(k) + \frac{q_{o,d}(k)}{\tau^{\mathrm{nc}}} \quad \text{for } \ell \in \mathscr{L}_o^{\mathrm{out}} \cap \mathscr{L}_d \tag{2}$$

with $D_{o,d}(k) = 0$ for $k \geqslant K$. Moreover,

$$q_{o,d}(k+1) = \max\left(0, q_{o,d}(k) + \Big(D_{o,d}(k) - \sum_{\ell \in \mathscr{L}_o^{\mathrm{out}} \cap \mathscr{L}_d} u_{\ell,d}(k)\Big)\tau^{\mathrm{nc}}\right) \tag{3}$$

But queues can form also inside the network. We assume that the DCVs run with maximum speed along the track segments and, if necessary, they wait before crossing the junction in vertical queues. Let $q_{v,d}(k)$ denote the length at time instant t_k of the vertical queue at junction $v \in \mathscr{I}$, for DCVs going to destination $d \in \mathscr{D}$.

Taking into account that a flow on link ℓ has a delay of κ_ℓ time steps before it reaches the end of the link, for every internal node $v \in \mathscr{I}$ and for every $d \in \mathscr{D}$ we have:

$$F_{v,d}^{\mathrm{out}}(k) \leqslant F_{v,d}^{\mathrm{in}}(k) + \frac{q_{v,d}(k)}{\tau^{\mathrm{nc}}} \tag{4}$$

where $F_{v,d}^{\mathrm{in}}(k)$ is the flow into the queue at junction v, being defined as:

$$F_{v,d}^{\mathrm{in}}(k) = \sum_{\ell \in \mathscr{L}_v^{\mathrm{in}} \cap \mathscr{L}_d} u_{\ell,d}(k - \kappa_\ell) \tag{5}$$

and where $F_{v,d}^{\mathrm{out}}(k)$ is the flow out of the queue at junction v, defined as:

$$F_{v,d}^{\mathrm{out}}(k) = \sum_{\ell \in \mathscr{L}_v^{\mathrm{out}} \cap \mathscr{L}_d} u_{\ell,d}(k) \ . \tag{6}$$

The evolution of the length of the queue for every internal node $v \in \mathscr{I}$ and for every $d \in \mathscr{D}$ is given by:

$$q_{v,d}(k+1) = \max\left(0, q_{v,d}(k) + \big(F_{v,d}^{\mathrm{in}}(k) - F_{v,d}^{\mathrm{out}}(k)\big)\tau^{\mathrm{nc}}\right) \tag{7}$$

Moreover, for each origin $o \in \mathscr{O}$ and for each junction $v \in \mathscr{I}$ we have the following constraints:

$$\sum_{d \in \mathscr{D}} q_{o,d}(k+1) \leq q_o^{\max} \tag{8}$$

$$\sum_{d \in \mathscr{D}} q_{v,d}(k+1) \leq q_v^{\max} \tag{9}$$

where $q_o^{\max}$ and $q_v^{\max}$ express (respectively) the maximum number of DCVs the conveyor belt transporting bags towards loading stations can accommodate and the maximum number of DCVs the track segments of the incoming links of that junction can accommodate.

We also have the following constraint for every link ℓ:

$$\sum_{d \in \mathscr{D}} u_{\ell,d}(k) \leqslant U_\ell^{\max} \tag{10}$$

where $U_\ell^{\max}$ is the maximum flow of DCVs that can enter link ℓ. In this paper we do not consider flow restrictions on links directly connected to a destination; so, $U_{\ell_d}^{\max} = \infty$ for each link ℓ_d directly connected to destination d.

Then, at time step k, the model of the DCV flows through the network of tracks describing (2)–(10) can be written as a system of equalities and a system of inequalities as follows:

$$\mathbf{q}_{k+1} = \mathscr{M}^{\mathrm{eq}}(\mathbf{q}_k, \mathbf{u}_k)$$
$$\mathscr{M}^{\mathrm{ineq}}(\mathbf{q}_{k+1}, \mathbf{u}_k) \leq 0$$

where

- $\mathbf{q}_k$ is the vector consisting of all the queue lengths $q_{o,d}(k)$, for all $o \in \mathscr{O}$ and for all $d \in \mathscr{D}$, and of all the queue lengths $q_{v,d}(k)$, for all $v \in \mathscr{I}$ and for all $d \in \mathscr{D}$,
- $\mathbf{u}_k$ is the vector consisting of all the flows $u_{\ell,d}(k)$, for all $d \in \mathscr{D}$ and for all $\ell \in \mathscr{L}_d$.

4.5 Performance index

Next we define the performance index to be used for computing the optimal routing at step k for a prediction period of N time steps.

The objective is to have each bag arriving at its end point within a given time interval $[t_d^{\mathrm{close}} - \tau_d^{\mathrm{open}}, t_d^{\mathrm{close}})$ where t_d^{close} is the time instant when the end point d closes and τ_d^{open} is the time period for which the end point d stays open for a specific flight. We assume t_d^{close} and τ_d^{open} to be integer multiples of τ_{s}.

Hence, one MPC objective that allows a piecewise affine performance criterion is to achieve a desired flow at destination d during the prediction period. Let u_d^{desired} denote the desired piecewise constant flow profile at destination d as sketched in Figure 7, where the area under u_d^{desired} equals the total number of bags out of the total demand that have to be sent to destination d. Note that $u_d^{\mathrm{desired}}(k) = 0$ for all $k < k_d^{\mathrm{open}}$ and all $k \geq k_d^{\mathrm{close}}$ with $k_d^{\mathrm{open}} = \dfrac{t_d^{\mathrm{close}} - \tau^{\mathrm{open}}}{\tau^{\mathrm{nc}}}$ and $k_d^{\mathrm{close}} = \dfrac{t_d^{\mathrm{close}}}{\tau^{\mathrm{nc}}}$.

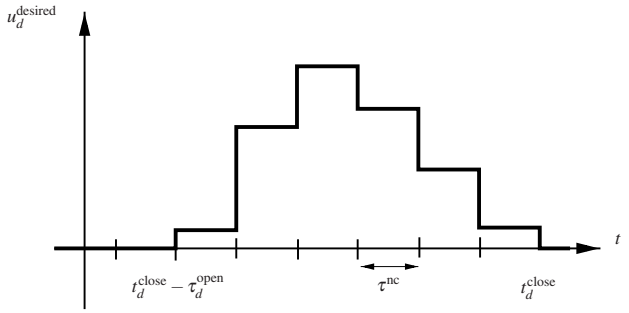


Figure 7 Desired arrival profile at destination d.

Let $\kappa_{\ell_d} = \dfrac{\tau_{\ell_d}}{\tau^{\text{nc}}}$. Hence, one can define the following penalty for flow profiles corresponding to destination $d \in \mathscr{D}$:

$$J_{d,k}^{\text{pen}} = \left| u_d^{\text{desired}}(k) - u_{\ell_d,d}(k+\kappa_{\ell_d}) \right|$$

where ℓ_d is the incoming link of destination d.

Later on we will include the penalty term $\sum_{i=k}^{k+N-1-\kappa_{\ell_d}} J_{d,i}^{\text{pen}}$ into the MPC performance criterion for each destination d and for each time step k. Note that we make the summation of these penalization indices only up to $k+N-1-\kappa_{\ell_d}$ since for $i > k+N-1-\kappa_{\ell_d}$ the variable $u_{\ell_d,d}(k+\kappa_{\ell_d})$ is not defined at MPC step k.

Moreover, note that using $\sum_{i=k}^{k+N-1-\kappa_{\ell_d}} J_{d,i}^{\text{pen}}$ for each destination d and for each time step k as MPC performance criterion, could have adverse effects for small prediction horizons. Therefore, to counteract these effects, we also consider as additional controller goal maximizing the flows of all links that are not directly connected to unloading stations. To this aim, let $\tau_{\ell,d,k}^{\text{link}}$ be the typical time required for a DCV that entered link ℓ in $[t_k, t_{k+1})$ to reach destination d, with $\tau_{\ell,d,k}^{\text{link}}$ an integer multiple of τ_{s}. Note that these durations are determined based on historical data. Also, let $\kappa_{l,d} = \dfrac{\tau_{\ell,d,k}^{\text{link}}}{\tau^{\text{nc}}}$. Then one can define the following penalty:

$$J_{\ell,d,k}^{\text{flow}} = \begin{cases} u_{\ell,d}(k) \text{ if } k_d^{\text{open}} - \kappa_{l,d} \le k < k_d^{\text{close}} - \kappa_{l,d} \\ 0 \qquad \text{otherwise} \end{cases}$$

This penalty will be later on used in the MPC performance criterion.

Next, in order to make sure that *all* the bags will be handled in finite time, we also include in the MPC performance criterion the weighted length of queues at each junction in the network as presented next. Let $\tau_{v,d}^{\text{junc}}$ be the typical time required for a DCV in the queue at junction v to reach destination d, with $\tau_{v,d}^{\text{junc}}(k)$ an integer multiple of τ^{nc}. Also, let $\kappa_{v,d} = \dfrac{\tau_{v,d}^{\text{junc}}(k)}{\tau^{\text{nc}}}$. Then we define the new penalty:

$$J_{v,d,k}^{\text{overdue}} = \begin{cases} d_{v,d}^{\min} q_{v,d}(k) \text{ if } k \ge k_d^{\text{close}} - \kappa_{v,d} \\ 0 \qquad\qquad \text{otherwise} \end{cases}$$

where $d_{v,d}^{\min}$ represents the length of the shortest route from junction v to destination d. Note that $J_{v,d,k}^{\text{overdue}}$ is nonzero only for steps that are larger than or equal to $k_d^{\text{close}} - \kappa_{v,d}$. Moreover, for these steps $J_{v,d,k}^{\text{overdue}}$ is proportional to $d_{v,d}^{\min}$. The reason for this is that we want to penalize more the queues at junctions that are further away from destination d because the DCVs in those queues will need a longer time to travel to destination d.

Finally, let $\mathscr{L}^{\text{dest}}$ denote the set of links directly connected to unloading stations. Then the MPC performance index is defined as follows:

$$J_{k,N} = \sum_{d\in\mathscr{D}} \Bigg(\sum_{i=k}^{k+N-1-\kappa_{\ell_d}} \lambda_d J_{d,i}^{\text{pen}} + \beta \sum_{i=k}^{k+N-1} \sum_{v\in\mathscr{J}} J_{v,d,i}^{\text{overdue}} - \alpha \sum_{i=k}^{k+N-1} \sum_{\ell\in(\mathscr{L}\setminus\mathscr{L}^{\text{dest}})\cap\mathscr{L}_d} J_{\ell,d,i}^{\text{flow}} \Bigg) \tag{11}$$

with $\lambda_d > 0$ a weight that expresses the importance of the flight assigned to destination d, $\alpha \ll 1$ and $\beta \ll 1$ nonnegative weighting parameters.

Then the nonlinear MPC optimization problem is defined as follows:

$$\begin{aligned} &\min_{\mathbf{u}_k,\ldots,\mathbf{u}_{k+N-1},\mathbf{q}_{k+1},\ldots,\mathbf{q}_{k+N}} J_{k,N}(\mathbf{q}_k,\mathbf{u}_k) \\ &\quad \text{subject to} \\ &\quad \mathbf{q}_{k+1} = \mathscr{M}^{\text{eq}}(\mathbf{q}_k,\mathbf{u}_k) \\ &\qquad \vdots \\ &\quad \mathbf{q}_{k+N} = \mathscr{M}^{\text{eq}}(\mathbf{q}_{k+N-1},\mathbf{u}_{k+N-1}) \\ &\quad \mathscr{M}^{\text{ineq}}(\mathbf{q}_{k+1},\mathbf{u}_k) \le 0 \\ &\qquad \vdots \\ &\quad \mathscr{M}^{\text{ineq}}(\mathbf{q}_{k+N},\mathbf{u}_{k+N-1}) \le 0 \end{aligned} \tag{12}$$

The nonlinear MPC optimization problem defined above is typically complex and it requires a large computational effort to solve. Therefore, in the next section we will recast this problem into an MILP one for which efficient and fast solvers are available.

4.6 MILP optimization problem for the network controller

Mixed integer linear programming (MILP) problems are optimization problems with a linear objective function, subject to linear equality and inequality constraints. The general formulation for a mixed-integer linear programming problem is the following:

$$\begin{aligned} &\min_{\mathbf{x}^{\text{MILP}}} \mathbf{c}^\top \mathbf{x}^{\text{MILP}} \\ &\quad \text{subject to} \\ &\qquad \mathbf{A}^{\text{eq}}\mathbf{x}^{\text{MILP}} = \mathbf{b}^{\text{eq}} \\ &\qquad \mathbf{A}\mathbf{x}^{\text{MILP}} \le \mathbf{b} \\ &\qquad \mathbf{x}^{\text{low}} \le \mathbf{x}^{\text{MILP}} \le \mathbf{x}^{\text{up}} \end{aligned}$$

where $\mathbf{c}$, $\mathbf{x}^{\text{MILP}}$, $\mathbf{x}^{\text{low}}$, $\mathbf{x}^{\text{up}}$, $\mathbf{b}^{\text{eq}}$, and $\mathbf{b}$ are vectors, with $\mathbf{x}^{\text{low}}$ the lower bound of $\mathbf{x}^{\text{MILP}}$ and $\mathbf{x}^{\text{up}}$ its upper bound, and where $\mathbf{A}^{\text{eq}}$ and $\mathbf{A}$ are matrices (all these vectors and matrices have appropriate size). Note that MILP solvers compute solutions

$\mathbf{x}^{\text{MILP}}$ for the problem above, where some of the elements of $\mathbf{x}^{\text{MILP}}$ are restricted to integer values.

Next we transform the dynamic optimal route choice problem presented above into an MILP problem, for which efficient solvers have been developed (Fletcher and Leyffer, 1998). To this aim we use the following equivalences, see (Bemporad and Morari, 1999), where f is a function defined on a bounded set X with upper and lower bounds M and m for the function values, δ is a binary variable, y is a real-valued scalar variable, and ε is a small tolerance (typically the machine precision):

P1: $[f(x) \leqslant 0] \iff [\delta = 1]$ is true if and only if

$$\begin{cases} f(x) \leqslant M(1-\delta) \\ f(x) \geqslant \varepsilon + (m-\varepsilon)\delta \end{cases},$$

P2: $y = \delta f(x)$ is equivalent to

$$\begin{cases} y \leqslant M\delta \\ y \geqslant m\delta \\ y \leqslant f(x) - m(1-\delta) \\ y \geqslant f(x) - M(1-\delta) \end{cases}.$$

As example we will show how equation (3) of the nonlinear route choice model presented in the previous section can be transformed into a system of linear equations and inequalities by introducing some auxiliary variables. For the other equations of the route choice model we apply a similar procedure.

We consider now (3). This is a nonlinear equation and thus it does not fit the MILP framework. Therefore, we will first introduce the binary variables $\delta_{o,d}(k)$ such that

$$\delta_{o,d}(k) = 1 \text{ if and only if}$$
$$q_{o,d}(k) + \Big(D_{o,d}(k) - \sum_{l \in \mathscr{L}_o^{\text{out}} \cap \mathscr{L}_d} u_{\ell,d}(k)\Big)\tau^{\text{nc}} \leqslant 0 \qquad (13)$$

and rewrite (3) as follows:

$$q_{o,d}(k+1) = \big(1-\delta_{o,d}(k)\big) \cdot \Big(q_{o,d}(k) + \big(D_{o,d}(k) - \sum_{l \in \mathscr{L}_o^{\text{out}} \cap \mathscr{L}_d} u_{\ell,d}(k)\big)\tau^{\text{nc}}\Big). \qquad (14)$$

Condition (13) is equivalent to (cf. Property P1):

$$\begin{cases} f(k) \leqslant (q_o^{\max} + D_{o,d}^{\max}\tau^{\text{nc}})(1-\delta_{o,d}(k)) \\ f(k) \geqslant \varepsilon + (-U^{\max}\tau^{\text{nc}} - \varepsilon)\delta_{o,d}(k) \end{cases},$$

where $f(k) = q_{o,d}(k) + \big(D_{o,d}(k) - \sum_{l \in \mathscr{L}_o^{\text{out}} \cap \mathscr{L}_d} u_{\ell,d}(k)\big)\tau^{\text{nc}}$, $q_o^{\max}$ is the maximal queue length at origin o, and where $D_{o,d}^{\max} = \max_k D_{o,d}(k)$ is the maximal demand for origin-destination pair (o,d).

However, (14) is still nonlinear since it contains a multiplication of a binary variable $\delta_{o,d}(k)$ with a real-valued (linear) function. However, by using Property **P2** this equation can be transformed into a system of linear inequalities.

The rest of the model equations can be transformed, in a similar way, into a system of MILP equations. Next we will transform the MPC performance index into its MILP form.

The problem

$$\min \sum_{i=k}^{k+N-1} \sum_{d \in \mathscr{D}} \lambda_d \left| u_d^{\text{desired}}(i) - u_{\ell_d,d}(i+\kappa_{\ell_d}) \right|$$

can be written as:

$$\begin{aligned} \min \ & \sum_{i=k}^{k+N-1} \sum_{d \in \mathscr{D}} \lambda_d u_d^{\text{diff}}(i) \\ \text{s.t.} \ & \\ & u_d^{\text{diff}}(i) \geqslant u_d^{\text{desired}}(i) - u_{\ell_d,d}(i+\kappa_{\ell_d}) \\ & u_d^{\text{diff}}(i) \geqslant -u_d^{\text{desired}}(i) + u_{\ell_d,d}(i+\kappa_{\ell_d}) \\ & \text{for } i = k, \ldots, k+N-1. \end{aligned}$$

which is a linear programming problem that has as optimal solution

$$\begin{aligned} u_d^{\text{diff},*}(i) &= \max\Big(u_d^{\text{desired}}(i) - u_{l,d}^*(i + \frac{\tau_d^{\text{dest}}}{\tau^{\text{nc}}}), \\ &\qquad - u_d^{\text{desired}}(i) + u_{l,d}^*(i + \frac{\tau_d^{\text{dest}}}{\tau^{\text{nc}}})\Big) \\ &= |u_d^{\text{desired}}(i) - u_{l,d}^*(i + \frac{\tau_d^{\text{dest}}}{\tau^{\text{nc}}})| \end{aligned}$$

with $u_{l,d}^*(i)$ the optimum flow on link l for destination d during the time window $[t_i, t_{i+1})$.

If we add the MILP equations of the model, the nonlinear optimization problem of Section 4.5 can be written as an MILP problem.

Several efficient branch-and-bound MILP solvers (Fletcher and Leyffer, 1998) are available for MILP problems. Moreover, there exist several commercial and free solvers for MILP problems such as, e.g, CPLEX, Xpress-MP, GLPK, or lp_solve, see (Atamtürk and Savelsbergh, 2005) for an overview. In principle, — i.e., when the algorithm is not terminated prematurely due to time or memory limitations, — these algorithms guarantee to find the global optimum. This global optimization feature is not present in the other optimization methods that can be used to solve the original nonlinear, nonconvex, nonsmooth optimization problem (12). Moreover, if the computation time is limited (as is often the case in on-line real-time control), then it might occur that the MILP solution can be found within the allotted time whereas the global and multi-start local optimization algorithm still did not converge to a good solution As a result, the MILP solution may even give a better system performance than the solution returned by the prematurely terminated global and multi-start local optimization method (as will be illustrated in Section 6.4).

5 Switch control

We now focus on the switch controller for the proposed hierarchy, and on how optimal switch positions can be determined.

Recall that at each control step k, the network controller provides optimal flows for each link in the network and for

each destination. Let these flows be denoted by $u^{\text{opt}}_{\ell,d}(k),\ldots,u^{\text{opt}}_{\ell,d}(k+N-1)$ with $d\in\mathscr{D}$, $\ell\in\mathscr{L}\cap\mathscr{L}_d$ and N the prediction horizon of the network controller. Then the switch controller of each junction has to compute optimal switch-in and switch-out positions such that the tracking error between the reference optimal flow trajectory and the flow trajectory obtained by the switch controller is minimal for each network controller time step $k=0,\ldots,K^{\text{sim}}$.

Recall that the optimal flows $u^{\text{opt}}_{\ell,d}(k),\ldots,u^{\text{opt}}_{\ell,d}(k+N-1)$ are determined for the time window $[t_k,t_{k+N})$ with $t_k=t_0+k\tau^{\text{nc}}$. In order to determine the switch control action during the time window $[t_k,t_{k+N})$ we will use again MPC. Next we will refer to one junction $v\in\mathscr{I}$ only. For all other junctions, the switch control actions are determined similarly.

Let τ^{sc} be the switch controller sampling time. We select the sampling time τ^{sc} to be an integer multiple of τ^{switch} (τ^{switch} is the minimum duration between consecutive toggles imposed as an operational constraint). Moreover, we select the sampling time τ^{nc} of the network controller and the sampling time τ^{sc} of the switch controller such that τ^{nc} is an integer multiple of τ^{sc}.

Let k^{sc} be an integer that expresses the number of switch control actions determined until now. At t_k, k^{sc} is defined as $k^{\text{sc}}=\dfrac{\tau^{\text{nc}}}{\tau^{\text{sc}}}k$. Then let $t^{\text{sw}}_{k^{\text{sc}}}$ denote the time instant corresponding to the time step k^{sc} of the switch controller, $t^{\text{sw}}_{k^{\text{sc}}}=t_0+k^{\text{sc}}\tau^{\text{sc}}$ with t_0 the time instant when we start the simulation.

Furthermore, let $s^{\text{in}}_v(k^{\text{sc}})$ denote the position of the switch-in at junction v during the time interval $\left[t^{\text{sw}}_{k^{\text{sc}}},t^{\text{sw}}_{k^{\text{sc}}+1}\right)$ and let $s^{\text{out}}_v(k^{\text{sc}})$ denote the position of the switch-out at junction v during $\left[t^{\text{sw}}_{k^{\text{sc}}},t^{\text{sw}}_{k^{\text{sc}}+1}\right)$.

We want to determine the switch control sequence during the time window $[t_k,t_{k+N})$ while using MPC with a prediction period of N^{sc} steps. Hence, at each MPC step k^{sc}, the switch controller solves the following optimization problem:

$$\min_{\mathbf{s}_{v,k^{\text{sc}},N^{\text{sc}}}} \quad J^{\text{sw}}_{v,k^{\text{sc}},N^{\text{sc}}} \tag{15}$$

with $\mathbf{s}_{v,k^{\text{sc}},N^{\text{sc}}}=[s^{\text{in}}_v(k^{\text{sc}})\ldots s^{\text{in}}_v(k^{\text{sc}}+N^{\text{sc}}-1)\ldots s^{\text{out}}_v(k^{\text{sc}})\ldots s^{\text{out}}_v(k^{\text{sc}}+N^{\text{sc}}-1)]^\top$ if junction v has 2 incoming and 2 outgoing links ($\mathbf{s}_{v,k^{\text{sc}},N^{\text{sc}}}$ contains only switch-in or only switch-out positions if junction v has only 1 outgoing or only 1 incoming link respectively) and with $J^{\text{sw}}_{v,k^{\text{sc}},N^{\text{sc}}}$ the local MPC performance index defined as:

$$J^{\text{sw}}_{v,k^{\text{sc}},N^{\text{sc}}}=\sum_{\ell\in\mathscr{L}^{\text{out}}_v}\left|X^{\text{opt}}_{\ell,k,k^{\text{sc}},N^{\text{sc}}}(\mathbf{u}^{\text{opt}}_\ell)-X_{\ell,k^{\text{sc}},N^{\text{sc}}}(\mathbf{s}_{v,k^{\text{sc}},N^{\text{sc}}})\right| + \gamma\left(n^{\text{sw\_in}}_{k^{\text{sc}},N^{\text{sc}}}(\mathbf{s}_{v,k^{\text{sc}},N^{\text{sc}}})+n^{\text{sw\_out}}_{k^{\text{sc}},N^{\text{sc}}}(\mathbf{s}_{v,k^{\text{sc}},N^{\text{sc}}})\right)$$

where

- $X^{\text{opt}}_{\ell,k,k^{\text{sc}},N^{\text{sc}}}(\mathbf{u}^{\text{opt}}_\ell)$ denotes the optimal number of DCVs to enter the outgoing link ℓ of junction v during the period $\left[t^{\text{sw}}_{k^{\text{sc}}},t^{\text{sw}}_{k^{\text{sc}}+N^{\text{sc}}-1}\right)$, where $\mathbf{u}^{\text{opt}}_\ell$ is the vector consisting of all the flows $u^{\text{opt}}_{\ell,d}(k),\ldots,u^{\text{opt}}_{\ell,d}(k+N)$ with $d\in\mathscr{D}$ and $\ell\in\mathscr{L}\cap\mathscr{L}_d$. The variable $X^{\text{opt}}_{\ell,k,k^{\text{sc}},N^{\text{sc}}}(\mathbf{u}^{\text{opt}}_\ell)$ is derived later on (see (16)).
- $X_{\ell,k^{\text{sc}},N^{\text{sc}}}(\mathbf{s}_{v,k^{\text{sc}},N^{\text{sc}}})$ is the actual number of DCVs entering link ℓ during the prediction period. The variable $X_{\ell,k^{\text{sc}},N^{\text{sc}}}$ is determined via simulation for a nonlinear (event-based) model similar to the one of Tarău et al. (2009a) (the difference is that now the switch positions $\mathbf{s}_{v,k^{\text{sc}},N^{\text{sc}}}$ are given for each period $[t^{\text{sw}}_{k^{\text{sc}}},t^{\text{sw}}_{k^{\text{sc}}+1}),\ldots,[t^{\text{sw}}_{k^{\text{sc}}+N^{\text{sc}}-1},t^{\text{sw}}_{k^{\text{sc}}+N^{\text{sc}}})$ instead of for each of the next N^{sc} DCVs to cross a junction);
- $n^{\text{sw\_in}}_{k^{\text{sc}},N^{\text{sc}}}(\mathbf{s}_{v,k^{\text{sc}},N^{\text{sc}}})$ and $n^{\text{sw\_out}}_{k^{\text{sc}},N^{\text{sc}}}(\mathbf{s}_{v,k^{\text{sc}},N^{\text{sc}}})$ represent the number of toggles of the switch-in and of the switch-out respectively during the prediction window $\left[t^{\text{sw}}_{k^{\text{sc}}},t^{\text{sw}}_{k^{\text{sc}}+N^{\text{sc}}}\right)$, which are obtained from simulation;
- γ is a nonnegative weighting parameter.

Next we derive the variable $X^{\text{opt}}_{\ell,k,k^{\text{sc}},N^{\text{sc}}}(\mathbf{u}^{\text{opt}}_\ell)$. To this aim, we first determine how many steps $p_{k^{\text{sc}}}$ of the network controller will be involved in solving (15) as follows: $p_{k^{\text{sc}}}=\left\lceil\dfrac{N^{\text{sc}}\tau^{\text{sc}}}{\tau^{\text{nc}}}\right\rceil$ where $\lceil x\rceil$ denotes the smallest integer larger than or equal to x (so, $p_{k^{\text{sc}}}\geq 1$). Furthermore, note that the index k of the time instant t_k for which $t_k\leq t^{\text{sw}}_{k^{\text{sc}}}<t_{k+1}$ can be computed as follows: $k=\left\lfloor\dfrac{k^{\text{sc}}\tau^{\text{sc}}}{\tau^{\text{nc}}}\right\rfloor$ where $\lfloor x\rfloor$ denotes the largest integer less than or equal to x. Figure 8 illustrates the prediction window $\left[t^{\text{sw}}_{k^{\text{sc}}},t^{\text{sw}}_{k^{\text{sc}}+N^{\text{sc}}-1}\right)$ with respect to the window $[t_k,t_{k+p_{k^{\text{sc}}}})$.

The variable $X^{\text{opt}}_{\ell,k,k^{\text{sc}}}(\mathbf{u}^{\text{opt}}_\ell)$ is given by:

$$X^{\text{opt}}_{\ell,k,k^{\text{sc}},N^{\text{sc}}}(\mathbf{u}^{\text{opt}}_\ell)=\tau^{\text{left}}_{1,k^{\text{sc}}}\sum_{d\in\mathscr{D}}u^{\text{opt}}_{\ell,d}(k)+\tau^{\text{nc}}\sum_{i=k+1}^{k+p_{k^{\text{sc}}}-2}\sum_{d\in\mathscr{D}}u^{\text{opt}}_{\ell,d}(i)+\tau^{\text{left}}_{2,k^{\text{sc}}}\sum_{d\in\mathscr{D}}u^{\text{opt}}_{\ell,d}(k+p_{k^{\text{sc}}}-1) \tag{16}$$

where $\sum_{i=k+1}^{k+j}x(i)=0$ by definition for $j<1$ and where

$$\tau^{\text{left}}_{1,k^{\text{sc}}}=\min(t_{k+1},t^{\text{sw}}_{k^{\text{sc}}+N^{\text{sc}}-1})-t^{\text{sw}}_{k^{\text{sc}}},$$

$$\tau^{\text{left}}_{2,k^{\text{sc}}}=\begin{cases}t^{\text{sw}}_{k^{\text{sc}}+N^{\text{sc}}-1}-t_{k+p_{k^{\text{sc}}}-1} & \text{if } p_{k^{\text{sc}}}>1\\ 0 & \text{otherwise.}\end{cases}$$

6 Case study

In this section we present a benchmark case study involving a basic set-up to illustrate the network-level control approach for DCV-based baggage handling systems proposed in this paper. First, we will describe the set-up and the details of the scenarios used for our simulations. Next, we will discuss and analyze the obtained results.

6.1 Set-up

We consider the network of tracks depicted in Figure 9 with 4 loading stations, 2 unloading stations, 9 junctions, and

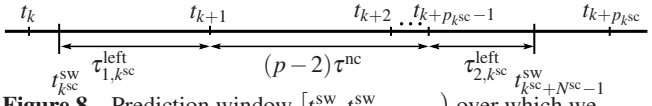


Figure 8 Prediction window $\left[t^{\text{sw}}_{k^{\text{sc}}},t^{\text{sw}}_{k^{\text{sc}}+N^{\text{sc}}-1}\right)$ over which we solve the MPC optimization problem (15) illustrated with respect to the window $[t_k,t_{k+p_{k^{\text{sc}}}})$ for $p_{k^{\text{sc}}}>2$.

20 unidirectional links, where the free-flow travel time is provided for each link. This network allows more than four possible routes to each destination from any origin point (e.g., d_1 can be reached from o_1 via junctions v_1, v_4, v_8; v_1, v_4, v_8, v_9, v_8; v_1, v_2, v_5, v_4, v_8; $v_1, v_2, v_6, v_7, v_9, v_8$, and so on). We consider 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.

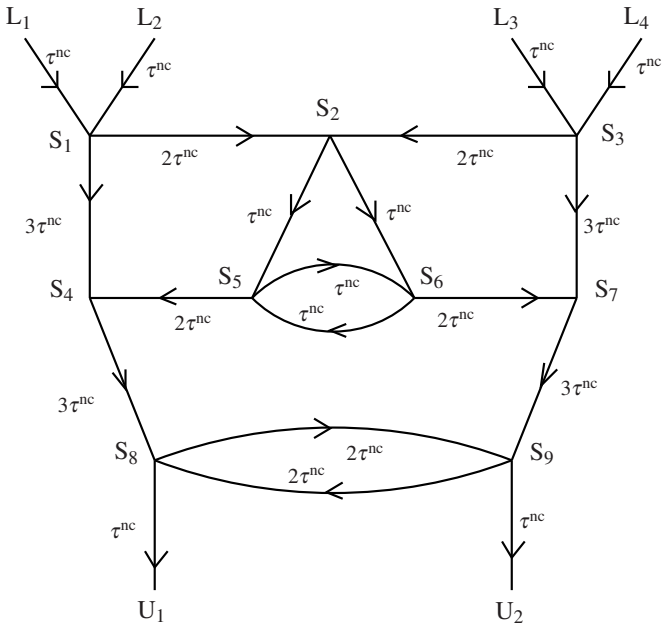


Figure 9 Case study for a DCV-based baggage handling system.

We assume that the velocity of each DCV varies between 0 m/s and 10 m/s. 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

In order to assess the performance of the proposed hierarchical control framework we define six scenarios where 2400 bags will be loaded into the baggage handling system (600 bags at each loading station). We consider three classes of demand profiles called "$\mathbf{dp}_1$", "$\mathbf{dp}_2$", and "$\mathbf{dp}_3$" hereafter. According to these classes, the bags arrive at each loading station in the time interval $[t_0, t_0 + 180\,\text{s})$, the arrival times at a loading station being allocated randomly, using a uniform distribution according to the following cases:

$\mathbf{dp}_1$: the 600 bags arrive at the loading station with a constant rate of 3.3 bags/s;

$\mathbf{dp}_2$: 100 bags arrive at the loading station during the time window $[t_0, t_0 + 60\,\text{s})$, 100 bags arrive during $[t_0 + 120\,\text{s}, t_0 + 180\,\text{s})$, and the rest of 400 bags arrives during $[t_0 + 60\,\text{s}, t_0 + 120\,\text{s})$;

$\mathbf{dp}_3$: 100 bags arrive during the time interval $[t_0, t_0 + 120\,\text{s})$ and the rest of the bags, i.e., 500 bags, arrives after $t = t_0 + 120\,\text{s}$, i.e., during $[t_0 + 120\,\text{s}, t_0 + 180\,\text{s})$.

We also consider two different initial states of the system where 60, and respectively 120 DCVs are already transporting bags in the network, running from origins $o_1, \ldots, o_4$ to junctions v_1 and v_3, from v_1 to v_2, and from v_3 to v_2. Their positions at t_0 are assigned such that between each 2 consecutive DCVs we have a minimum safe distance of 2 m, and between the DCV closest to the next to be passed junction and the junction we again have 2 m. Later on, when comparing the control methods in Section 6.4, we will also use as criterion the static priorities (the flight priorities) of all the bags to be handled. These priorities are assigned randomly in the set $\{1, 2\}$ using a uniform distribution.

We assume that we have only two flights assigned to the unloading stations d_1 and d_2 (one flight assigned to one unloading station). Furthermore, we assume that the time windows within which we need the bags at their end points are $[t_0 + 800\,\text{s}, t_0 + 1400\,\text{s})$ for d_1 and $[t_0 + 1000\,\text{s}, t_0 + 1600\,\text{s})$ for d_2.

We simulate a period of 40 minutes. The control time step for the network controller is set to 60 s, while the control time step for the switch controller is set to 2 s. Moreover, the minimum duration between two consecutive switches is set to 2 s (recall that this is an operational constraint).

6.3 Control approaches

In the next section we will compare the results obtained when using the proposed hierarchical route control framework and the switch control approaches that have shown to give good performance in (Tarău et al., 2009a) and (Tarău et al., 2009b): centralized MPC, distributed MPC with a single round of downstream and upstream communication, and distributed heuristics.

In order to solve the MILP optimization of the network controller we have used the CPLEX solver of the Matlab optimization toolbox *Tomlab*, while to solve the nonlinear optimization problem of the switch controller we have chosen the *genetic* algorithm implemented in Matlab via the function `ga` with multiple runs (for these simulations we run the genetic algorithm three times for each optimization). Note that in order to keep the total computation time low, for both approaches — hierarchical MPC and centralized MPC — we shift the horizon with N, respectively N^{sc} samples at each MPC step. Also, due to the same reason (computational requirements), we allow a limited amount of time for solving the optimization problem corresponding to the centralized route control and distributed MPC with a single round of downstream and upstream communication (the computation time allowed for each optimization is of 1 hour for centralized MPC and 80 seconds for distributed MPC).

As prediction horizon we consider $N = 6$ for the network controller and $N^{\text{sc}} = 30$ for the switch controller of the hierarchical control, $N = 40$ for the centralized MPC switch control, and $N = 5$ for the distributed MPC. We have chosen these values since simulations indicate that they give

a good trade-off between the total computation time and performance.

As model of the real system (i.e. the simulation model) we use the event based model of (Tarău et al., 2009a) while the prediction model is either the model of the hierarchical control proposed in this paper (when we refer to this control approach) or the event based model of (Tarău et al., 2009a) when we refer to centralized MPC and distributed heuristics, or a local event based model when we refer to distributed MPC with a single round of downstream and upstream communication.

6.4 Results

Based on simulations we now compare, for the given scenarios, the results obtained for the proposed control frameworks. The simulations were performed on a 3.0 GHz P4 with 1 GB RAM. The results of the simulations are reported in Figure 11. For this comparison we consider the total performance of the system used in both papers (Tarău et al., 2009b) and (Tarău et al., 2009a). As illustrated in Figure 10 this performance index penalizes both the overdue time for each bag to be handled and its additional storage time. This goes as follows. Assume that bag index i with $i \in \{1,2,\ldots,N^{\text{bags}}\}$ (where N^{bags} is the total number of bags to be handled) arrives at its endpoint at time instant $t_i^{\text{bag,unload}}$, then the penalization of bag index i is given by:

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

where $t_d^{\text{bag,close}}$ is the time instant when the end point d closes for bag index i, $\tau_i^{\text{bag,open}}$ 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, the weighting parameter σ_i represents the static priority of bag index i (the priority of the flight), and the weighting parameter $\lambda_1 > 0$ expresses the penalty for the additionally stored baggage.

The total performance index of the baggage handling system that we use when comparing the methods is defined as:

$$J^{\text{tot}}(\mathbf{t}) = \sum_{i=1}^{N^{\text{bags}}} J_i^{\text{pen}}(t_i^{\text{bag,unload}})$$

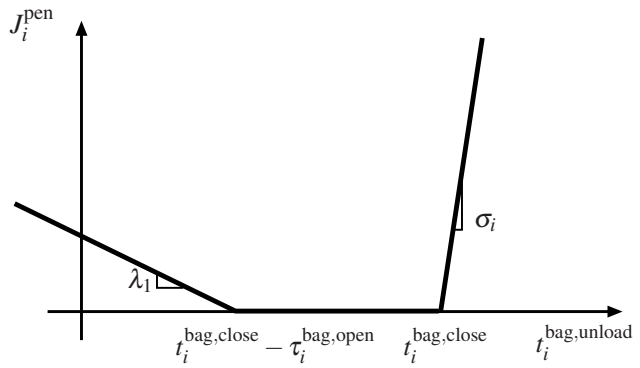
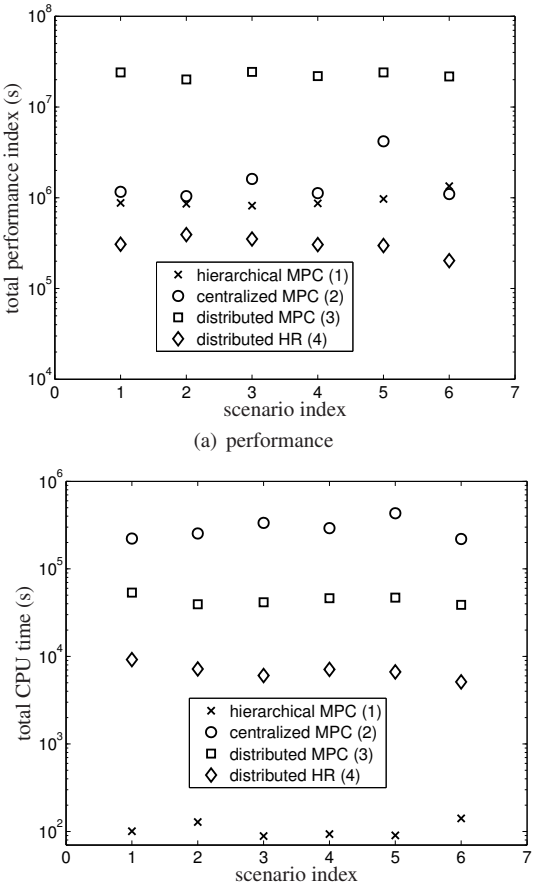


Figure 10 Objective function J_i^{pen}.

with $\mathbf{t}$ the vector that consists of time instants when each of the bags to be handled is actually unloaded $\mathbf{t} = [t_1^{\text{bag,unload}}\ t_2^{\text{bag,unload}}\ \ldots\ t_{N^{\text{bags}}}^{\text{bag,unload}}]^\top$. Note that in Figure 11(a) the lower the total performance index corresponding to one scenario is, the better the efficiency of the baggage handling system.



(a) performance

(b) computation time

Figure 11 Comparison of the results obtained for the total closed-loop simulation when using (1) the hierarchical control framework, (2) the centralized switch control approach of Tarău et al. (2009a), (3) the distributed MPC switch control with a single round of downstream and upstream communication presented in (Tarău et al., 2009a), and (4) the distributed heuristic switch control of Tarău et al. (2009b).

Moreover, let $J^{\text{approach,avg}}$ denote the average performance index:

$$J^{\text{approach,avg}} = \frac{1}{|\Delta|} \sum_{j \in \Delta} J_j^{\text{tot,approach}}$$

with Δ the set of considered scenarios and $J_j^{\text{tot,approach}}$ the total performance corresponding to the specific control approach and scenario index j. Then in Table 1 we list the average results.

The simulation results indicate that using the hierarchical control framework typically yields a better system performance than using centralized MPC or distributed MPC with a single round of downstream and upstream communication. But, note that the solutions of centralized MPC or distributed MPC were returned by the prematurely

Table 1 Comparison of average performance of the system and total computation time.

Control approach	$J^{\text{approach,avg}}(s)$	total CPU time (s)
Centralized MPC	$1.70 \cdot 10^6$	$2.92 \cdot 10^5$
Distributed MPC	$2.27 \cdot 10^7$	$4.44 \cdot 10^4$
Distributed HR	$3.09 \cdot 10^5$	$6.89 \cdot 10^3$
Hierarchical MPC	$9.55 \cdot 10^5$	$1.07 \cdot 10^2$

terminated global and multi-start local optimization method. However, even with the computational restrictions mentioned above (we allow a limited amount of time for solving an optimization problem), the total computation time of centralized MPC and of distributed MPC with a single round of downstream and upstream communication is much larger (over 40 hours) than the one of the hierarchical control (an average of 102 s per junction, plus 6 s for solving the MILP optimization problems).

The performance index J^{tot} obtained when using the distributed heuristics (for a prediction window of 5 s) is a bit lower than the one obtained when using the hierarchical control framework, but the total computation time required to determine the solution is also much larger. Also note that typically the heuristic approaches give worse performance than the predictive methods (see, e.g., Tarău et al. (2009b)). But for this we have to allow sufficient time for the computations for the predictive methods.

Hence, the hierarchical control with MILP solutions offers a balanced trade-off between the performance of the system and the total computation time required to determine the route choice solution.

7 Conclusions

In this paper we have proposed a hierarchical control framework for efficiently computing routes for destination coded vehicles (DCVs) that transport bags in an airport on a railway network. In the proposed control framework the network controller computes reference flow trajectories over time for each link in the network so that the performance of the DCV-based baggage handling system is optimized. Then the switch controllers determine the sequence of optimal positions for their ingoing and outgoing switches so that the tracking error between the reference trajectory and the future flow trajectory is minimized. In general, the problem of computing optimal routes for a collection of DCVs is a nonlinear, nonconvex, mixed integer optimization problem, and very expensive to solve in terms of computational efforts. Therefore, we have considered flows of DCVs and then used an alternative approach for reducing the complexity of the computations by rewriting the nonlinear optimization problem of the network controller as a mixed integer linear programming (MILP) problem. The advantage is that for MILP optimization problems solvers are available that allow us to efficiently compute the global optimal solution. The solution of the MILP problem is then used in computing optimal switch control actions. For a benchmark case study we have compared the hierarchical route control with switch control approaches that have proved to give good performance in previous work. The obtained results indicate that the proposed hierarchical route control offers a balanced trade-off between efficiency and total computation time when compared to distributed heuristics and to predictive switch control approaches where the multi-start local optimization method has been terminated prematurely.

In future work we will perform extensive simulations in order to assess the efficiency of the hierarchical route control approach. Furthermore, in order to better assess the performance of the proposed approach, we will study in more depth the quality of the approximation used at the higher level of the proposed framework. Moreover, we will also consider the line balancing problem (i.e. route assignment for each empty DCV such that all the loading stations have enough empty DCVs at any time instant). Finally, in future work we will also use the concept of *platooning* and develop efficient control methods for optimally creating the platoons.

Acknowledgments

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 of Large Scale Systems” (contract number INFSO-ICT-223854).

References

A. Atamtürk and M. Savelsbergh. Integer-programming software systems. *Annals of Operations Research*, 140(1): 67–124, November 2005.

A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3):407–427, March 1999.

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

R. de Neufville. The baggage system at Denver: Prospects and lessons. *Journal of Air Transport Management*, 1(4): 229–236, December 1994.

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

R. Fletcher and S. Leyffer. Numerical experience with lower bounds for MIQP branch-and-bound. *SIAM Journal on Optimization*, 8(2):604–616, May 1998.

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

A. Langevin, D. Lauzon, and D. Riopel. Dispatching, routing, and scheduling of two automated guided vehicles in a flexible manufacturing system. *International Journal of Flexible Manufacturing Systems*, 8(3):247–262, July 1996.

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

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

J.B. Rawlings and D.Q. Mayne. *Model Predictive Control: Theory and Design*. Nob Hill Publishing, Madison, Wisconsin, USA, 2009.

F. Taghaboni and J.M.A. Tanchoco. Comparison of dynamic routeing techniques for automated guided vehicle systems. *International Journal of Production Research*, 33(10): 2653–2669, October 1995.

A. Tarău, B. De Schutter, and J. Hellendoorn. Travel time control of destination coded vehicles in baggage handling systems. In *Proceedings of the 17th IEEE International Conference on Control Applications*, pages 293–298, San Antonio, Texas, USA, September 2008.

A.N. Tarău, B. De Schutter, and H. Hellendoorn. Receding horizon approaches for route choice control of automated baggage handling systems. In *Proceedings of the European Control Conference 2009*, pages 2978–2983, Budapest, Hungary, August 2009a.

A.N. Tarău, B. De Schutter, and H. Hellendoorn. Distributed route choice control in DCV-based baggage handling systems. In *Proceedings of the 18th IEEE International Conference on Control Applications*, pages 818–824, Saint Petersburg, Russia, July 2009b.