Delft University of Technology

Delft Center for Systems and Control

Technical report 09-048

Model-based control for route choice in automated baggage handling systems*

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

If you want to cite this report, please use the following reference instead:
A.N. Tarău, B. De Schutter, and H. Hellendoorn, "Model-based control for route choice in automated baggage handling systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 40, no. 3, pp. 341–351, May 2010. doi:10.1109/TSMCC.2009.2036735

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.bartdeschutter.org/abs/09_048.html

Model-based control for route choice in automated baggage handling systems

Alina N. Tarău, Bart De Schutter, Hans Hellendoorn

***Abstract*—State-of-the-art baggage handling systems transport luggage in an automated way using destination coded vehicles (DCVs). These vehicles transport the bags at high speeds on a network of tracks. Currently, the DCVs are routed through the system using routing schemes based on preferred routes. These routing schemes respond to the occurrence of predefined events. We do not consider such predefined preferred routes. Instead we develop advanced control methods to determine the optimal routing in case of dynamic demand. In order to optimize the performance of the system we first develop and compare efficient centralized, decentralized, and distributed predictive methods. Next, to reduce the computational requirements, we also propose some heuristic methods. Finally, to assess the performance of the proposed control approaches, the methods are compared for several scenarios on a benchmark case study.**

***Index Terms*—Baggage handling systems, route choice, model predictive control.**

I. Introduction

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

Higher-level control problems for a DCV-based baggage handling 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. 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 stopped at switching points, if required. Finally, low-level control problems are also coordination and synchronization when loading a bag onto a DCV, and when unloading it to its end point. Note that we assume the low-level controllers already present in the system.

We focus on the higher-level control problem of optimally routing DCVs on the network of tracks so that all the bags to be handled arrive at their end points within given time windows. The goal of our work is to develop and compare efficient control approaches (viz. predictive control methods and heuristic approaches) for route choice control of each DCV transporting a bag through the track network. These control approached are developed in a centralized, a decentralized, and a distributed manner. The control approach is said to be decentralized if the local control actions are computed without any communication or coordination between the local controllers, while the control approach is said to be distributed if additional communication and coordination between neighboring controllers is involved, see e.g. [1], [2].

All authors are with Delft University of Technology, Delft Center for Systems and Control, Mekelweg 2, 2628 CD Delft, The Netherlands (e-mail: a.n.tarau@tudelft.nl, b@deschutter.info, j.hellendoorn@tudelft.nl)

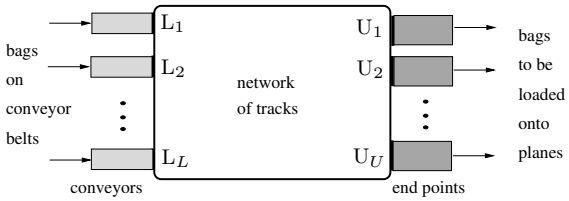


Fig. 1. Baggage handling system using DCVs.

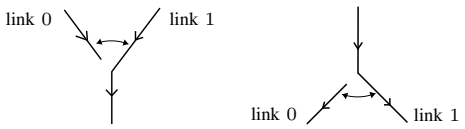
The paper is organized as follows. In Section II, the event-based model of the system is presented. Afterwards, in Section III we introduce the objective function to be used when computing the control law. Section IV presents some preliminaries. In Section V, VI, VII, VIII, and IX we propose advanced control methods for computing the route of loaded DCVs. The analysis of the simulation results and the comparison of the proposed control methods are elaborated in Section X. Finally, Section XI draws the concluding remarks.

II. Route choice model and operational constraints

A. Operation of the system

Consider the general DCV-based baggage handling system sketched in Figure 1. This baggage handling system operates as follows: given a demand of bags (identified by their unique code) together with their arrival times at the loading stations, and the network of tracks, the route of each DCV has to be computed subject to the operational constraints presented in Section II-C, such that all the bags to be handled arrive at their end points within given time windows. The bags unloaded outside their end points' time window are then penalized as presented in Section III.

We consider a system with L loading stations and U unloading stations as depicted in Figure 1. Let us index the bags loaded onto DCVs at station L_l with $\ell \in \{1, \ldots, L\}$ as $b_{\ell,1}, ..., b_{\ell,N_l}$ with N_ℓ the number of bags that will be loaded at station L_ℓ during the entire simulation period. Then let



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

Fig. 2. Incoming and outgoing links at a junction.

$t_{\ell,i}^{\text{arrival}}$ denote the time instant when bag $b_{\ell,i}$ actually arrives at loading station L_ℓ ($t_{\ell,i}^{\text{arrival}} < t_{\ell,i+1}^{\text{arrival}}$ for $i = 1, \ldots, N_\ell - 1$). Then we define the L-tuple $\mathcal{T} = (\mathbf{t}_1^{\text{arrival}}, \mathbf{t}_2^{\text{arrival}}, \ldots, \mathbf{t}_L^{\text{arrival}})$ that comprises the vectors of bag arrival times $\mathbf{t}_\ell^{\text{arrival}} = [t_{\ell,1}^{\text{arrival}} \ldots t_{\ell,N_\ell}^{\text{arrival}}]^{\text{T}}$. We also assume that the track network has S junctions S_1, S_2,..., S_S. Without loss of generality we can assume that each junction S_s with $s \in \{1, 2, \ldots, S\}$ has maximum 2 incoming links and maximum 2 outgoing links, both indexed by $l \in \{0, 1\}$ as sketched in Figure 2. If S_s has 2 incoming links then it also has a switch going into the junction (called switch-in hereafter). If S_s has 2 outgoing links then it has also a switch going out of the junction (called switch-out hereafter). This corresponds to current practice in state-of-the-art baggage handling systems.

B. Model

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

We assume that there is a sufficient number of DCVs present in the system so that when a bag is at the loading station there is always a DCV ready to transport it. If this is not the case, then one has to solve also the line balancing and empty cart management problem (i.e. optimally assigning loading stations to empty DCVs and optimally route them towards the loading stations). We also assume piecewise constant[1] velocity for each DCV.

Let X be the number of bags that the baggage handling system has to handle and let X^{crt} be the total number of bags that entered the track network up to the current time instant $t^{\text{crt}} \leq t_0 + \tau^{\text{max\_sim}}$ with t_0 the initial simulation time and $\tau^{\text{max\_sim}}$ the maximum simulation period. Also, let DCV_i denote the DCV that transports the ith bag that entered the track network up to the current time instant, $i \leq X^{\text{crt}}$.

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

[1]One can always approximate an arbitrary speed profile arbitrarily well by a piecewise constant speed profile.

Algorithm 1. Model of the baggage handling system

$t^{\text{crt}} \leftarrow t_0$
while $t^{\text{crt}} \leq t_0 + \tau^{\text{max\_sim}}$ **do**
  for $\ell = 1$ to L **do**
    $\tau_\ell^{\text{load}} \leftarrow$ time until next loading event at L_ℓ
  end for
  for $\upsilon = 1$ to U **do**
    $\tau_\upsilon^{\text{unload}} \leftarrow$ time until next event at U_υ
  end for
  for $s = 1$ to S **do**
    $\tau_s^{\text{cross}} \leftarrow$ time until next DCV-crosses-S_s event
    $\tau_s^{\text{sw\_in}} \leftarrow$ time until next switch-in event at S_s
    $\tau_s^{\text{sw\_out}} \leftarrow$ time until next switch-out event at S_s
  end for
  for $i = 1$ to X^{crt} **do**
    $\tau_i^{\text{v\_update}} \leftarrow$ time until next velocity-update event of DCV_i
  end for
  $\tau^{\min} \leftarrow \min\big(\min_{\ell=1,\ldots,L} \tau_\ell^{\text{load}}, \min_{\upsilon=1,\ldots,U} \tau_\upsilon^{\text{unload}}, \min_{s=1,\ldots,S} \tau_s^{\text{sw\_in}}, \min_{s=1,\ldots,S} \tau_s^{\text{sw\_out}}, \min_{i=1,\ldots,X^{\text{crt}}} \tau_i^{\text{v\_update}} \big)$
  $t^{\text{crt}} \leftarrow t^{\text{crt}} + \tau^{\min}$
  take action (i.e. load, unload, cross junction, switch-in update, switch-out update, velocity update)
  update the state of the system
end while

If multiple events occur at the same time, we take all these events into account when updating the state of the system.

Next we describe the variables involved in determining the model of **Algorithm 1**.

τ_ι^{load}: If there is no bag coming towards loading station L_ι, then $\tau_\iota^{\text{load}} = \infty$. Otherwise, a conveyor transports bags towards loading station L_ι. Recall that we assume that there are sufficient DCVs present in the system so that when a bag is at the loading station there is a DCV ready for transporting it. Then, for the current state of the system at time instant t^{crt}, the time period τ_ι^{load} is equal to $\max\left(t_{\iota,j}^{\text{arrival}} - t^{\text{crt}}, t_{\iota,j,i}^{\text{safe}}\right)$ where $t_{\iota,j}^{\text{arrival}}$ denote the time instant when bag $b_{\iota,j}^{\text{load}}$ actually arrives at loading station L_ι, $j-1$ is the number of bags that have been already loaded from L_ι (so, the next bag to be loaded at L_ι has local index j), and $\tau_{\iota,j,i}^{\text{safe}}$ expresses the time period that has to pass until it is safe for bag $b_{\iota,j}^{\text{load}}$ to be loaded onto a DCV. Then $\tau_{\iota,j,i}^{\text{safe}} = 0$ if $d_{\iota,j-1}^{\text{travel}} \geq d^{\min}$ and $\tau_{\iota,j,i}^{\text{safe}} = \frac{d^{\min} - d_{\iota,j-1}^{\text{travel}}}{\max\left(v^{\text{jam}}, v_{\iota,j-1}^{\text{load}}\right)}$ otherwise, where $d^{\min}$ is the minimum safe distance between DCVs, $d_{\iota,j-1}^{\text{travel}}$ is the position of the DCV transporting bag $b_{\iota,j-1}^{\text{load}}$ on the outgoing link of loading station L_ι, $v_{\iota,j-1}^{\text{load}}$ is the velocity of that DCV, and $v^{\text{jam}} \ll 1\,\text{m/s}$ is the speed to be used in case of jam. The speed v^{jam} is determined based on empirical data.

$\tau_\upsilon^{\text{unload}}$: The time period that will pass until the next unloading event occurs at unloading station U_υ is $\tau_\upsilon^{\text{unload}} = \frac{d_\upsilon^{\text{link}} - d_\upsilon^{\text{travel,closest}}}{v_\upsilon^{\text{closest}}}$ where d_υ^{link} is the length of the incoming link of unloading station U_υ, $d_\upsilon^{\text{travel,closest}}$ is the position of the DCV closest to U_υ on the incoming link of U_υ, and $v_\upsilon^{\text{closest}}$ is the current speed of this DCV. If there is no DCV on the

incoming link of U_v, then $\tau_v^{\text{unload}} = \infty$ by definition.

τ_s^{cross}: Consider the switch into junction S_s to be positioned at the current time on the incoming link $l \in \{0,1\}$ of S_s. Then the time that will pass until the next DCV crosses S_s is $\tau_s^{\text{cross}} = \frac{d_{s,l}^{\text{link}} - d_{s,l}^{\text{travel,closest}}}{\max\left(v^{\text{jam}}, v_{s,l}^{\text{closest}}\right)}$ if there is a DCV on link l into S_s and $\tau_s^{\text{cross}} = \infty$ otherwise, where $d_{s,l}^{\text{link}}$ is the length of the incoming link l of junction S_s, $d_{s,l}^{\text{travel,closest}}$ is the position of the DCV closest to S_s on the incoming link l of S_s, and $v_{s,l}^{\text{closest}}$ is the velocity of that DCV.

$\tau_s^{\text{sw\_in}}$, $\tau_s^{\text{sw\_out}}$: Once the toggle command of switch-in and switch-out is given, the position of the switch-in and switch-out is toggled after $\tau_s^{\text{sw\_in}}$ and $\tau_s^{\text{sw\_out}}$ time units respectively. Assume that the toggle commands are given at $t^{\text{sw\_in}} \geq t^{\text{crt}}$ and $t^{\text{sw\_out}} \geq t^{\text{crt}}$. Then $\tau_s^{\text{sw\_in}} = \max\left(t^{\text{sw\_in}}, t_s^{\text{sw\_in\_prev}} + \tau^{\text{sw}}\right) - t^{\text{crt}}$ and $\tau_s^{\text{sw\_out}} = \max\left(t^{\text{sw\_out}}, t_s^{\text{sw\_out\_prev}} + \tau^{\text{sw}}\right) - t^{\text{crt}}$ where τ^{sw} is the minimum time period after which the switch at a junction can be toggled, and where $t_s^{\text{sw\_in\_prev}}$ and $t_s^{\text{sw\_out\_prev}}$ are respectively the time instants when the switch-in and the switch-out at junction S_s have been toggled last.

$\tau_i^{\text{v\_update}}$: We calculate $\tau_i^{\text{v\_update}}$ according to the cases enumerated below — $d^{\min}$ is the minimum safe distance between DCVs and $d_{\mathrm{DCV}_i}^{\text{travel}}$ is the position of DCV_i on the incoming link of S_s. First assume DCV_i to be traveling towards junction S_s on link $l \in \{0,1\}$, with no other DCV traveling in front of DCV_i on the same link l. Then, if $v_{\mathrm{DCV}_i} < v^{\max}$ and $d_{s,l}^{\text{link}} - d_{\mathrm{DCV}_i}^{\text{travel}} > d^{\min}$, the velocity of DCV_i has to be updated immediately to $v^{\max}$ ($\tau_i^{\text{v\_update}} = 0$). If $v_{\mathrm{DCV}_i} > 0$, $d_{s,l}^{\text{link}} - d_{\mathrm{DCV}_i}^{\text{travel}} \leq d^{\min}$ and the switch-in at S_s is *not* positioned on the incoming link l that DCV_i travels, then $\tau_i^{\text{v\_update}} = 0$ and $v_{\mathrm{DCV}_i} \leftarrow 0$. Next let $\mathrm{DCV}_i^{\text{prev}}$ denote the DCV traveling on the same incoming link as DCV_i, in front of DCV_i, with no other DCV between them. Also, let $d_{\mathrm{DCV}_i^{\text{prev}}}^{\text{travel}}$ denote the position of $\mathrm{DCV}_i^{\text{prev}}$ on link l. Then, if $v_{\mathrm{DCV}_i} < v^{\max}$ and $d_{\mathrm{DCV}_i^{\text{prev}}}^{\text{travel}} - d_{\mathrm{DCV}_i}^{\text{travel}} > d^{\min}$, $\tau_i^{\text{v\_update}} = 0$ and $v_{\mathrm{DCV}_i} \leftarrow v^{\max}$. If $v_{\mathrm{DCV}_i} > v_{\mathrm{DCV}_i^{\text{prev}}}$ and $d_{\mathrm{DCV}_i^{\text{prev}}}^{\text{travel}} - d_{\mathrm{DCV}_i}^{\text{travel}} \leq d^{\min}$, then $\tau_i^{\text{v\_update}} = \frac{d_{\mathrm{DCV}_i^{\text{prev}}}^{\text{travel}} - d_{\mathrm{DCV}_i}^{\text{travel}} - d^{\min}}{\max\left(v^{\text{jam}}, v^{\max} - v_{\mathrm{DCV}_i^{\text{prev}}}\right)}$ and $v_{\mathrm{DCV}_i} \leftarrow v_{\mathrm{DCV}_i^{\text{prev}}}$. For any other case, we set $\tau_i^{\text{v\_update}} = \infty$.

According to the model, for each bag that has to be handled, we compute the time instants when the bag enters and exits the track network. Let t_i^{load} denote the time instant when the ith bag that entered the track network is loaded onto a DCV and let t_i^{unload} denote the time instant when the same bag is unloaded at its end point. We denote the model of the baggage handling system as $\mathbf{t} = \mathcal{M}(\mathcal{T}, x(t_0), \mathbf{u}, \mathbf{v})$, where: $\mathbf{t} = [t_1^{\text{load}} \ \ldots \ t_X^{\text{load}} \ t_1^{\text{unload}} \ \ldots \ t_X^{\text{unload}}]^{\mathrm{T}}$, $\mathbf{u}$ is the route control sequence, and $\mathbf{v}$ is the velocity sequence for each DCV.

C. Operational constraints

The operational constraints derived from the mechanical and design limitations of the system are the following: (1) a switch at a junction has to wait at least τ^{sw} time units after a toggle has occurred, in order to avoid the quickly and repeatedly movement back and forth of the switch which may lead to mechanical damage, and (2) the speed of each DCV is bounded between 0 and $v^{\max}$. These constraints are denoted as $\mathcal{C}(\tau^{\text{sw}}, v^{\max}) \leq 0$.

III. Performance criterion

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

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

where t_i^{end} is the time instant when the end point closes and the bags are loaded onto the plane, σ_i is the static priority of bag index i (the flight priority), $1 \leq \sigma_i \leq 10$, and τ_i^{open} is the maximum possible length of the time window for which the end point corresponding to bag index i is open for that specific flight. The weighting parameter $\lambda_1 > 0$ expresses the penalty for the delayed baggage.

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

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

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

The final objective function is given by:

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

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

Next, in order to determine the route of each DCV transporting a bag, we propose several predictive control methods — centralized, decentralized, and distributed model predictive control—, and two heuristic approaches — decentralized and distributed heuristics.

IV. Model-based predictive control

Model predictive control (MPC) is an on-line model-based predictive control design method [3], [4], [5]. Since its development in 1980 [6], [7], MPC has become the preferred control strategy for a large number of industrial processes, see, e.g. [8] for chemical engineering. Currently, MPC is viewed as one of the most promising control methods that can deal with nonlinear systems that are subject to operational constraints.

In the basic MPC approach, given an horizon N, at step $k \geq 0$, the future control sequence $u(k+1), u(k+2), \ldots, u(k+N)$

is computed by solving a discrete-time optimization problem over a period $[t_k, t_k + \tau_s N]$, where $t_k = t_0 + k\tau_s$ with τ_s the sampling time, so that the cost criterion is optimized subject to the operational constraints and the evolution of the system. MPC uses a receding horizon approach. So, after computing the optimal control sequence, only the first control sample is implemented, and subsequently the horizon is shifted. Next, the new state of the system is measured or estimated, and a new optimization problem at time t_{k+1} is solved using this new information. In this way, also a feedback mechanism is introduced.

V. Centralized model predictive control

We define now a variant of MPC, where k is not a time index, but a bag index. In this context bag step k corresponds to the time instant t_k^{load} when the kth bag has just entered the track network — if $k = 0$ bag step k corresponds to the time instant t_0. For this variant of MPC, the horizon N corresponds to the number of bags for which we look ahead, while computing the control $u(k+j)$ with $j \in \{1, 2, \ldots, N\}$ consists in determining the route of DCV_{k+j}. Next, we implement all the computed control samples, and accordingly we shift the horizon with N steps.

Assume that there is a fixed number R of possible routes from a loading station to an unloading station and that the R routes are numbered $1, 2, \ldots, R$. Let $r(i) \in \{1, 2, \ldots, R\}$ denote the route of DCV_i. We assume that at bag step k the route is selected once for each DCV without being adjusted after the decision has been made. Now let $\mathbf{r}(k)$ denote the future route sequence for the next N bags entering the network at bag step k, $\mathbf{r}(k) = [r(k+1)\, r(k+2)\, \ldots\, r(k+N)]^{\text{T}}$.

The total objective function of the centralized MPC is:

$$J_{k,N}^{\text{CMPC}}(\mathbf{r}(k)) = \sum_{i=1}^{k+N} J_i(\hat{t}_i^{\text{unload}})$$

where $\hat{t}_i^{\text{unload}}$ is the predicted unloading time of DCV_i depending on the routes of the first $k+N$ bags that entered the network. Accordingly, the MPC optimization problem at bag step k is defined as follows:

$$\begin{aligned} \min_{\mathbf{r}(k)} \; & J_{k,N}^{\text{CMPC}}(\mathbf{r}(k)) \text{ s.t.} \\ & \mathbf{t} = \mathcal{M}(\mathcal{T}, x(t_k), \mathbf{r}(k), \mathbf{v}(k)) \\ & \mathcal{C}(\tau^{\text{sw}}, v^{\max}) \leq 0 \end{aligned}$$

When using centralized MPC, at each bag step k, the future route sequence $\mathbf{r}(k)$ is computed over an horizon of N bags so that the objective function is minimized subject to the dynamics of the system and the operational constraints. To solve this nonlinear, non-convex, integer-valued optimization problem one could use e.g. *genetic* algorithms, *simulated annealing*, or *tabu search* [9].

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

VI. Decentralized model predictive control

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

A. Local system

Each local system consists of a junction, its incoming and its outgoing links. Let us now consider the most complex case, where junction S_s with $s \in \{1, 2, \ldots, S\}$ has both a switch-in and a switch-out. Moreover, S_s is not directly connected to an unloading station. Then we first index the bags that successively cross junction S_s during the entire simulation period as $b_{s,1}, b_{s,2}, \ldots, b_{s,N_s^{\text{bags}}}$, where N_s^{bags} is the number of bags that cross S_s during the simulation period.

B. Local control measures

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

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

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

C. Local objective function

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

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

where N_s^{cross} is the number of DCVs that actually crossed junction S_s during the prediction period, $\hat{t}_{s,k+j}^{\text{unload},*}$ is the

predicted unloading time instant of bag $b_{s,k+j}$, and λ^{pen} is a nonnegative weighting parameter. N_s^{cross} and $\hat{t}_{s,k+j}^{\text{unload},*}$ are determined by simulating the prediction model presented next for a given control sequence $\mathbf{u}_s(k)$.

D. Local prediction model

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

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

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

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

where

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

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

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

Then the optimal predicted unloading time instant is:

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

E. Local optimization problem

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

$$\begin{aligned} \min_{\mathbf{u}_s(k)} \; & J_{s,k,N_s}^{\text{DMPC}}(\mathbf{u}_s(k)) \text{ s.t.} \\ & \mathbf{t} = \mathcal{M}^{\text{local}}(\mathcal{T}, x_s(t_k), \mathbf{u}_s(k), \mathbf{v}_s(k)) \\ & \mathcal{C}(\tau^{\text{sw}}, v^{\max}) \leq 0 \end{aligned}$$

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

After computing the optimal control, only $u_s^{\text{sw\_in}}(k+1)$ and $u_s^{\text{sw\_out}}(k+1)$ 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_s bags.

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

VII. Distributed model predictive control

One can increase the performance of the *decentralized* control approach proposed above by implementing a *distributed* approach that uses additional communication between neighboring junctions.

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. But, in contrast to decentralized MPC, data will be now communicated between neighboring junctions which are characterized by the concept of level of influence. The levels of influence are defined as follows.

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 link to a loading station. Next, we consider all junctions connected to some junction with influence level 1 via an outgoing link of the junction with level 1, and we assign influence level 2 to them. In this way we recursively assign an influence level to each junction with the constraint that at most $\kappa_{\text{d}}^{\max}$ downstream influence levels are assigned to a given junction[2].

[2]The constraint that at most $\kappa^{\text{d,max}}$ downstream influence levels are assigned to a junction limits the computational complexity and keeps all levels of influence finite.

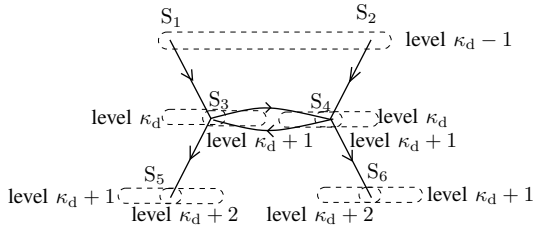


Fig. 3. Levels of downstream influence for parallel computation.

For example see Figure 3 where we define maximum 2 levels of downstream influence for each junction in the network. For this example we have considered the junctions S_1 and S_2 to have assigned downstream influence level $\kappa_\mathrm{d}-1$. Then S_3 and S_4 have assigned level κ_d (since these junctions are connected to S_1 and S_2 via outgoing links). Next, we assign influence level $\kappa_\mathrm{d}+1$ to S_4, S_5, S_3, and S_6 (since they are connected to S_3 and S_4). Note that now S_3 and S_4 have assigned 2 levels of downstream influence κ_d and $\kappa_\mathrm{d}+1$. Therefore, S_5 and S_6 have also assigned influence level $\kappa_\mathrm{d}+2$ (since they are connected to S_3 and S_4 with influence level $\kappa_\mathrm{d}+1$).

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 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 round of downstream communication

Let us now consider distributed MPC with a single round of downstream communication. This means that the local controller of each junction with influence level $\kappa_\mathrm{d}=1$ solves the local optimal control problem of Section VI.

After computing the optimal switch control sequence, each junction with influence level κ_d communicates to its neighboring junctions at level $\kappa_\mathrm{d}+1$ which bags (out of all the bags over which we make the prediction for the corresponding junction with influence level κ_d) will enter the incoming link of the junction at level $\kappa_\mathrm{d}+1$ and at which time instant. Next, we iteratively consider the junctions at levels $\kappa_\mathrm{d}=2,3$, etc. until level of influence $K^{\mathrm{downstream}}$, were $K^{\mathrm{downstream}}$ is the largest level of downstream influence assigned in the network. Then, for each junction on influence level $\kappa_\mathrm{d}>1$, we compute a local solution to the local MPC problem as presented next.

Assume S_s with $s\in\{1,\dots,S\}$ to have assigned influence level $\kappa_\mathrm{d}>1$. Let $\mathrm{S}_{s,l}^{\mathrm{prev}}$ denote the neighboring junction of S_s connected via the incoming link l of S_s (accordingly, $\mathrm{S}_{s,l}^{\mathrm{prev}}$ has assigned influence level $\kappa_\mathrm{d}-1$). Then, we compute a local solution for S_s to the local MPC problem over an horizon of

$$N_s=\min\Big(N^{\max},\sum_{l=0}^{1}\big(n_{s,l}^{\mathrm{horizon}}+n_{s,l,0}^{\mathrm{pred\_cross}}+n_{s,l,1}^{\mathrm{pred\_cross}}\big)\Big)$$

bags where $N^{\max}$ is the maximum prediction horizon for the local MPC problem, $n_{s,l}^{\mathrm{horizon}}$ is the number of DCVs traveling at time instant t^{crt} on link $l\in\{0,1\}$ going into S_s, and $n_{s,l,m}^{\mathrm{pred\_cross}}$ with $m\in\{0,1\}$ is the number of DCVs traveling towards $\mathrm{S}_{s,l}^{\mathrm{prev}}$ on its incoming link m that we predict (while solving the local optimization problem at $\mathrm{S}_{s,l}^{\mathrm{prev}}$) to cross $\mathrm{S}_{s,l}^{\mathrm{prev}}$ and continue their journey towards S_s.

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

Every time some bag has crossed some junction we update the local control of junctions in the network as follows. Assume that some bag has just crossed junction S_s at level κ_d. Then, we update the control of $\mathrm{S}_{s,l}^{\mathrm{next}}$ at level $\kappa_\mathrm{d}+1$, $\mathrm{S}_{s,l,m}^{\mathrm{next}}$ at level $\kappa_\mathrm{d}+2$, and so on until level $K^{\mathrm{downstream}}$, where $\mathrm{S}_{s,l,m}^{\mathrm{next}}$ is the junction connected to $\mathrm{S}_{s,l}^{\mathrm{next}}$ via the outgoing link m of $\mathrm{S}_{s,l}^{\mathrm{next}}$.

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 in computing the control actions.

C. Distributed MPC with a single round of downstream and upstream communication

Now we add an extra round of communication and consider distributed MPC with a round of downstream and upstream communication. 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 as explained above. Next, for the junctions on level 1 of upstream influence we update the release rate of their incoming links as follows. We take as example junction S_s with $\kappa_\mathrm{u}=1$. For all other junctions we apply the same procedure. We virtually apply at S_s the optimal control sequence $\mathbf{u}_s^*$ that we have computed when optimizing downstream. Let $t_s^{\mathrm{last},*}$ be the time instant at which the last bag crossed S_s (out of all the bags over which we make the prediction for S_s). If $t_s^{\mathrm{last},*}<\tau^{\mathrm{rate}}$ we set $\zeta_{s,l}=\zeta^{\max}$ for $l=0,1$. Otherwise, if $n_{s,l}^{\mathrm{rate}}>0$, we set $\zeta_{s,l}=\dfrac{n_{s,l}^{\mathrm{rate}}}{\tau^{\mathrm{rate}}}$ with $n_{s,l}^{\mathrm{rate}}$ the number of DCVs that left the outgoing link l of S_s within the time window $[t_s^{\mathrm{last},*}-\tau^{\mathrm{rate}},t_s^{\mathrm{last},*}]$. Finally, if $n_{s,l}^{\mathrm{rate}}=0$ we set $\zeta_{s,l}=\varepsilon$.

Next we solve the local MPC problem presented in Section VII-B using the updated release rates and we compute the local control of all junctions at upstream level $\kappa_\mathrm{u}+1$. Recursively, we compute the local control until level K^{upstream} where K^{upstream} is the largest level of upstream influence assigned in the network.

By also performing the upstream round of communication, 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, but also the computational effort increases.

VIII. Decentralized heuristic approach

Decentralized MPC solves the switch control problem with the smallest computational effort among all the MPC approaches considered in this paper. However, in order to lower the computation time even more, we propose heuristic approaches to control the route of each DCV in this section and the next one. In contrast to decentralized MPC, the heuristic approaches use a prediction horizon of one step only. Each switch is now locally controlled based on heuristic rules as presented next. The local switch control of the decentralized heuristic approach is determined based only on local information regarding the flow of DCVs on the incoming and outgoing links of a junction. Consider junction S_s with $s \in \{1, 2, ..., S\}$.

A. Control of the switch-in

If S_s has a switch-in, every time when a bag enters one of the incoming links of S_s we update the local control of the switch-in at S_s. Let t^{crt} be such a time instant. Then we compute (as presented next) the control variable $\tau_s^{\mathrm{sw\_in}}$ which represents the time period until the position of the switch-in has to be changed.

For a junction S_s, we define the following variables:

- $\Gamma_{s,l}$ 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},
- $\rho_{s,l}^{\mathrm{static}}$ is the total static priority of link l, $\rho_{s,l}^{\mathrm{static}} = \sum_{i \in \Gamma_{s,l}} \sigma_i$,
- $\rho_{s,l}^{\mathrm{dyn}}$ is the total dynamic priority of link l, $\rho_{s,l}^{\mathrm{dyn}} = \sum_{i \in \Gamma_{s,l}} \frac{\hat{\delta}_i}{\delta_i^{\max}}$ 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 $\delta_i^{\max}$ the maximum time left to 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 $t_i^{\mathrm{new\_end}}$ for bag i is considered. Then the variable $\delta_i^{\max}$ is defined as $\delta_i^{\max} = t_i^{\mathrm{end}} - t^{\mathrm{crt}}$ if $t_i^{\mathrm{end}} - t^{\mathrm{crt}} > 0$ and $\delta_i^{\max} = t_i^{\mathrm{new\_end}} - t^{\mathrm{crt}}$ if $t_i^{\mathrm{end}} - t^{\mathrm{crt}} \leq 0$.

In order to determine the next position of the switch-in at S_s we compute the performance measure $p_{s,l}^{\mathrm{sw\_in}}$ 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 position of the switch-in at junction S_s (due to the operational constraint according to which the position of a switch at a junction can only change after minimum τ^{sw} time units):

$$p_{s,0}^{\mathrm{sw\_in}} = w^{\mathrm{st\_pr}} \rho_{s,0}^{\mathrm{static}} + w^{\mathrm{dyn\_pr}} \rho_{s,0}^{\mathrm{dyn}} - w^{\mathrm{sw\_in}} \tau^{\mathrm{sw}} I_s^{\mathrm{crt}}$$
$$p_{s,1}^{\mathrm{sw\_in}} = w^{\mathrm{st\_pr}} \rho_{s,1}^{\mathrm{static}} + w^{\mathrm{dyn\_pr}} \rho_{s,1}^{\mathrm{dyn}} - w^{\mathrm{sw\_in}} \tau^{\mathrm{sw}} (1 - I_s^{\mathrm{crt}})$$

where I_s^{crt} is the current position of the switch-in at junction S_s (i.e. $I_s^{\mathrm{crt}} = 0$ if the switch-in is positioned on the incoming link 0, and $I_s^{\mathrm{crt}} = 1$ if the switch-in is positioned on the incoming link 1). The weighting parameters $w^{\mathrm{st\_pr}}$, $w^{\mathrm{dyn\_pr}}$, and $w^{\mathrm{sw\_in}}$ are calibrated by solving an off-line optimization problem (for more details see [10]).

Let $z_{s,l} \in \Gamma_{s,l}$ denote the bag traveling on the incoming link l of S_s and which is closest to S_s and let $\tau_l^{\mathrm{arrival\_at\_S_s}}$ be the time period that the DCV transporting bag $z_{s,l}$ needs to travel (at maximum speed) the distance between the current position of bag $z_{s,l}$ and S_s.

The position of the switch-in at S_s is toggled only if $p_{s,0}^{\mathrm{sw\_in}} > p_{s,1}^{\mathrm{sw\_in}}$ and $I_s^{\mathrm{crt}} = 1$, or if $p_{s,1}^{\mathrm{sw\_in}} > p_{s,0}^{\mathrm{sw\_in}}$ and $I_s^{\mathrm{crt}} = 0$. If this is the case, then the current position of the switch-in is changed after $\tau_s^{\mathrm{sw\_in}} = \max(\tau^{\mathrm{sw}} - \tau_s^{\mathrm{sw\_in\_prev}}, \tau_{1-I_s^{\mathrm{crt}}}^{\mathrm{arrival\_at\_S_s}})$ time units where $\tau_s^{\mathrm{sw\_in\_prev}}$ is the time for which the switch-in at junction S_s with $s \in \{1, 2, \ldots, S\}$ has been in its current position.

B. Control of the switch-out

If S_s has a switch-out, every time when a bag is at junction S_s we compute the control variable $\tau_s^{\mathrm{sw\_out}}$ which represents the time period until the position of the switch-out has to be changed. This goes as follows.

Assume that bag i is at junction S_s. Then, using (4), we can predict the unloading time $\hat{t}_{s,l,r,i}^{\mathrm{unload}}$ of bag i at $\mathrm{S}_{s,i}^{\mathrm{dest}}$, the end point of bag i, when traveling on link $l \in \{0, 1\}$ out of S_s and route $r \in \mathcal{R}_{s,l,i}^{\mathrm{next}}$ where $\mathcal{R}_{s,l,i}^{\mathrm{next}}$ is the set of routes from $\mathrm{S}_{s,l}^{\mathrm{next}}$ to $\mathrm{S}_{s,i}^{\mathrm{dest}}$.

Next we compute the cost criterion $c_{s,l,i}^{\mathrm{sw\_out}}$ for $l = 0, 1$ that takes into account $J_i(\hat{t}_{s,l,i}^{\mathrm{unload},*})$, where

$$\hat{t}_{s,l,i}^{\mathrm{unload},*} = \underset{\{\hat{t}_{s,l,r,i}^{\mathrm{unload}} | r \in \mathcal{R}_{s,l,i}^{\mathrm{next}}\}}{\arg\min} J_i(\hat{t}_{s,l,r,i}^{\mathrm{unload}}),$$

and the current position O_s^{crt} of the outgoing switch:

$$c_{s,0,i}^{\mathrm{sw\_out}} = w^{\mathrm{pen}} J_i(\hat{t}_{s,0,i}^{\mathrm{unload},*}) + w^{\mathrm{sw\_out}} \tau^{\mathrm{sw}} O_s^{\mathrm{crt}}$$
$$c_{s,1,i}^{\mathrm{sw\_out}} = w^{\mathrm{pen}} J_i(\hat{t}_{s,1,i}^{\mathrm{unload},*}) + w^{\mathrm{sw\_out}} \tau^{\mathrm{sw}} (1 - O_s^{\mathrm{crt}})$$

The weighting parameters w^{pen} and $w^{\mathrm{sw\_out}}$ are previously calibrated.

The position of the switch-in at junction S_s is toggled only if $c_{s,0,i}^{\mathrm{sw\_out}} < c_{s,1,i}^{\mathrm{sw\_out}}$ and $O_s^{\mathrm{crt}} = 1$, or if $c_{s,1,i}^{\mathrm{sw\_out}} < c_{s,0,i}^{\mathrm{sw\_out}}$ and $O_s^{\mathrm{crt}} = 0$. If this is the case, then the switch-out is toggled after $\tau_s^{\mathrm{sw\_out}} = \max(0, \tau^{\mathrm{sw}} - \tau_s^{\mathrm{sw\_out\_prev}})$ where $\tau_s^{\mathrm{sw\_out\_prev}}$ is the time for which the switch-out at junction S_s with $s \in \{1, 2, \ldots, S\}$ has been in its current position.

IX. Distributed heuristic approach

In this section we develop an approach where the switch control is performed based on both local information and additional data regarding the flow of DCV on the incoming and outgoing links of the neighboring junctions. This is an extension of the previous decentralized heuristic approach.

A. Control of the switch-in

As in Section VIII-A, we consider the local system corresponding to a junction S_s with 2 incoming and 2 outgoing links, the control of the switch-in being updated every time

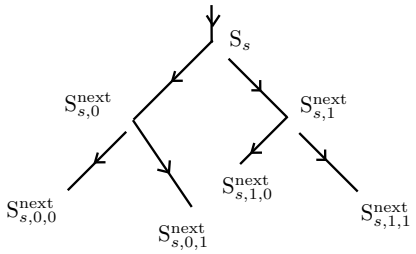


Fig. 4. Neighboring junctions.

(t^{crt}) some bag enters the incoming links of S_s. When applying the distributed heuristic approach we compute again the objective functions $p^{\text{sw\_in}}_{s,0}$ and $p^{\text{sw\_in}}_{s,1}$ defined in the previous section. However, in this section we also take into account the bags that will come towards S_s from its neighboring junctions in the next τ^{pred} time units. The period τ^{pred} is determined based on empirical data. Let $S^{\text{prev}}_{s,l}$ be the junction connected to S_s via the incoming link $l \in \{0,1\}$ of S_s.

We predict which bags will cross $S^{\text{prev}}_{s,l}$ and continue traveling towards S_s as follows. At time instant t^{crt} we compute the control sequence of the switch-in and switch-out using the decentralized heuristic rules for switch-in presented above, and the heuristic rules for switch-out presented in Section IX-B respectively. As prediction model we use the simulation model of **Algorithm 1** for the time period $[t^{\text{crt}}, t^{\text{crt}}+\tau^{\text{pred}})$. As result of this simulation we determine which bags will cross $S^{\text{prev}}_{s,l}$ and continue traveling towards S_s and at which time they will cross $S^{\text{prev}}_{s,l}$. Let Ω_l be the set of bags that will cross $S^{\text{prev}}_{s,l}$ when traveling towards S_s in the next τ^{pred} time units.

The time when junction S_s toggles its position is computed as in *Control of the switch-in* of Section VIII. The difference is that here we use the following performance measures:

$$\begin{aligned} p^{\text{sw\_in}}_{s,0} =& w^{\text{st\_pr}}(\rho^{\text{static}}_{s,0} + \varphi^{\text{static}}_{s,0}) + w^{\text{dyn\_pr}}(\rho^{\text{dyn}}_{s,0} + \varphi^{\text{dyn}}_{s,0}) \\ & - w^{\text{sw\_in}}\tau^{\text{sw}} I^{\text{crt}}_s \\ p^{\text{sw\_in}}_{s,1} =& w^{\text{st\_pr}}(\rho^{\text{static}}_{s,1} + \varphi^{\text{static}}_{s,1}) + w^{\text{dyn\_pr}}(\rho^{\text{dyn}}_{s,1} + \varphi^{\text{dyn}}_{s,1}) \\ & - w^{\text{sw\_in}}\tau^{\text{sw}}(1 - I^{\text{crt}}_s) \end{aligned}$$

where $\varphi^{\text{static}}_{s,l}$ is total static priority of the bags in $\Omega_{s,l}$, $\varphi^{\text{static}}_{s,l} = \sum_{i\in\Omega_{s,l}} \sigma_i$ and $\varphi^{\text{dyn}}_{s,l}$ is the total dynamic priority of the bags in $\Omega_{s,l}$, $\varphi^{\text{dyn}}_{s,l} = \sum_{i\in\Omega_{s,l}} \frac{\hat{\delta}_i}{\delta^{\max}_i}$.

B. Control of the switch-out

The position of the switch-out of junction S_s is computed similarly to Section VIII-B. However, in this case, when computing the predicted objective function for link $l = 0, 1$ and bag i, we do not look only at the congestion on the outgoing links of junction S_s, but also at the congestion farther in the network. So, we will predict the time that bag i needs to travel on the next[3] $\nu^{\max}$ links when trying to reach its destination where $\nu^{\max}$ denotes the maximum number of links we look ahead.

[3]We look only at the next $\nu^{\max}$ links in order to get some extra information on the network congestion state, while keeping the communication requirements low.

Let us consider next the case where $\nu^{\max} = 2$. As sketched in Figure 4, $S^{\text{next}}_{s,l,m}$ for $m = 0, 1$ denotes the neighboring junction of $S^{\text{next}}_{s,l}$ connected via link m out of $S^{\text{next}}_{s,l}$. Then the time period that bag i needs to travel link m out of $S^{\text{next}}_{s,l}$ considering the release rate $\zeta_{s,l,m}$ of link m out of $S^{\text{next}}_{s,l}$ is defined as:

$$\hat{\tau}^{\text{link}}_{s,l,m,i} = \begin{cases} \max\left(\dfrac{d^{\text{link}}_{s,l,m}}{v^{\max}}, \dfrac{N^{\text{DCV}}_{s,l,m,i}}{\zeta_{l,m}}\right) & \text{if link } l \text{ is not jammed} \\ \max\left(\dfrac{d^{\text{link}}_{s,l,m}}{v^{\text{jam}}}, \dfrac{N^{\text{DCV}}_{s,l,m,i}}{\zeta_{l,m}}\right) & \text{if link } l \text{ is jammed} \end{cases}$$

where

- $d^{\text{link}}_{s,l,m}$ is the length of the link m out of $S^{\text{next}}_{s,l}$,
- $N^{\text{DCV}}_{s,l,m,i} = n_{s,l,m,i} + \tilde{n}_{s,l,m,i}$ with $n_{s,l,m,i}$ the number of DCVs on link m out of $S^{\text{next}}_{s,l}$ at the time instant when bag i crosses junction S_s and $\tilde{n}_{s,l,m,i}$ the predicted number of DCVs on link l out of S_s that choose link m out of $S^{\text{next}}_{s,l}$. We assume that for a junction $S^{\text{next}}_{s,l}$, a fraction $\eta_{s,l}$ of the DCVs crossing $S^{\text{next}}_{s,l}$ take link $m = 0$ out of $S^{\text{next}}_{s,l}$. The fraction $\eta_{s,l}$ is determined based on empirical data.

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

$$\hat{t}^{\text{unload}}_{s,l,m,r,i} = t^{\text{cross}}_{s,i} + \hat{\tau}^{\text{link}}_{s,l,i} + \tau^{\text{link}}_{s,l,m,i} + \tau^{\text{route}}_r$$

where $t^{\text{cross}}_{s,i}$ is the time instant at which bag i crosses S_s defined as $t^{\text{cross}}_{s,i} = t^{\text{crt}} + \max(0, t_{\text{x}} - \tau^{\text{sw\_out\_prev}}_s)$, $\hat{\tau}^{\text{link}}_{s,l,m,i}$ is the time we predict that bag i will spend on link m out of $S^{\text{next}}_{s,l}$ and τ^{route}_r is the average travel time on route $r \in \mathcal{R}^{\text{next}}_{s,l,m,i}$.

Finally, in computing the cost criterion $c^{\text{sw\_out}}_{s,l,i}$ for $l = 0, 1$ defined in Section VIII we use $J_i(\hat{t}^{\text{unload},*}_{s,l,i})$ where $\hat{t}^{\text{unload},*}_{s,l,i}$ is the predicted unloading time that optimizes the objective function of bag i when choosing link $m \in \{0,1\}$ out of $S^{\text{next}}_{s,l}$, and route $r \in \mathcal{R}^{\text{next}}_{s,l,m,i}$:

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

X. 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 4 loading stations, 2 unloading station, and 9 junctions. 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.

We assume that the velocity of each DCV varies between 0 m/s and $v^{\max} = 20$ m/s. The lengths of the track segments are indicated in Figure 5.

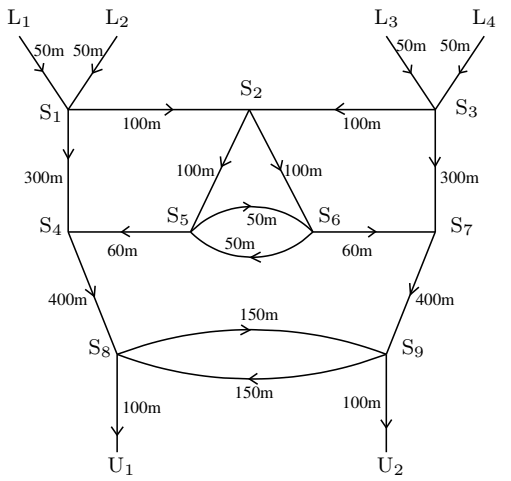


Fig. 5. Case study for a DCV-based baggage handling system.

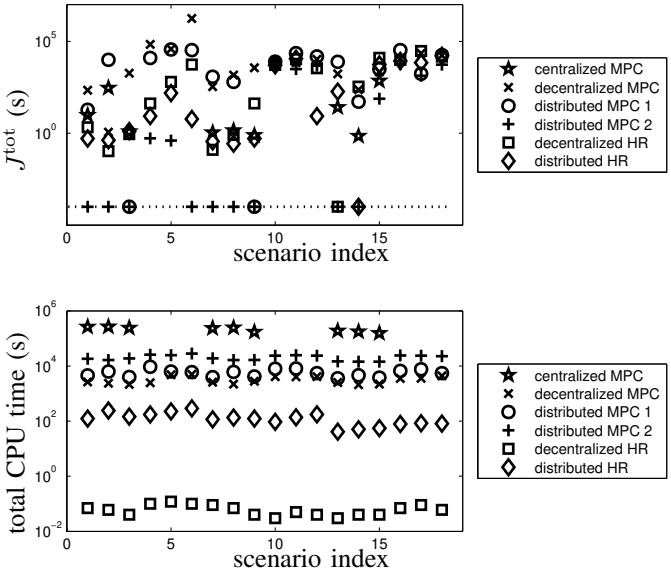


Fig. 6. Comparison of the results obtained using the proposed centralized, decentralized, and distributed control approaches (MPC and heuristics) for the total closed-loop simulation. In order to visualize on the logarithmic scale results such as $J^{\mathrm{tot}} = 0$ for some scenario, we set $J^{\mathrm{tot}} = 10^{-4}$ for that scenario.

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

B. Scenarios

For the calibration we have defined 18 scenarios where 120 bags will be loaded into the baggage handling system (30 bags at each loading station). We have considered 3 classes of demand profiles. According to these classes, the bags arrive at each loading station in the time interval $[t_0, t_0 + 100\,\mathrm{s}]$, the arrival times at a loading station being allocated randomly, using a uniform distribution according to the following cases (the tight arrival time instants that we use in these scenarios have the purpose to faster assess the efficiency of the proposed control methods): (1) the bags arrive at the loading station with a constant rate; (2) 5 bags arrive at a loading station during each of the time intervals $[t_0, t_0 + 40\,\mathrm{s})$ and $[t_0 + 60\,\mathrm{s}, t_0 + 100\,\mathrm{s}]$, and the rest of 110 the bags arrive during $[t_0+40\,\mathrm{s}, t_0+60\,\mathrm{s})$; (3) 10 bags arrive during the time interval $[t_0, t_0+80\,\mathrm{s})$ and the rest of the bags arrive after $t = t_0 + 80\,\mathrm{s}$.

We have also considered 2 different initial states of the system where 60, and respectively 120 DCVs are already transporting bags in the network, running from loading stations $\mathrm{L}_1 \ldots, \mathrm{L}_4$ to junctions S_1 and S_2, from S_1 to S_2, and from S_3 to S_2. Their position at t_0 and their static priorities $\in \{1, 2\}$ are assigned randomly.

The bags to be handled can be organized in 2 groups of bags. The first group consists of the bags that populate the DCV network before t_0 and the second one consists of the bags that enter the network after t_0. For a maximum storage period of 100 s at unloading stations, we examine both situations where the transportation of the bags is very tight (the last bag that enters the system can only arrive in time at the its endpoint if the DCV travels the shortest route with maximum speed), and respectively more relaxed.

C. Results

To solve the MPC optimization problems we have chosen a *genetic* algorithm with multiple runs of the Matlab optimization toolbox *Genetic Algorithm and Direct Search* 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. For all the proposed predictive control methods we have set the horizon to 3 bags. In Figure 6 we plot the total performance index J^{tot} and the total computation time[4] obtained when using the proposed control approaches. We plot the performance index corresponding to centralized MPC only for scenarios where the initial population of the network of tracks is small (60 DCVs) since the computation time for the case where the network is populated with 120 DCVs is larger than 10^6 s.

One expects that the best performance of the system is obtained when using *centralized* route choice control. This would have happened if we had allowed more runs[5] and a larger computation time[6] to calculate the solution of an MPC optimization problem. However, centralized control becomes intractable in practice when the number of junctions is large due to the high computation time required.

Simulation results indicate that using *decentralized MPC* lowers the computation time. Furthermore, the results confirm that *distributed MPC* gives better performance than decentralized MPC, but at the cost of higher computational effort. Note that the computation time of both decentralized and distributed MPC is obtained while, for solving one optimization problem, we run the genetic algorithm 4 times on a single computer, with a limited number of function evaluations[6].

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

[5]In these simulations we ran the *genetic* algorithm 4 times for each optimization problem.

[6]In order to reduce the computational effort we have optimized the number of function evaluations for each run with respect to the length of the prediction horizon. In future work we will also optimize other parameters of the search algorithm such as mutation and crossover fractions since this may improve the efficiency of the control approach.

Finally, the *decentralized* and *distributed heuristic approaches* give typically worse results than distributed MPC, but with very low computation time.

One can easily gain several orders of magnitude in the total computation time of the proposed control approaches by using parallel computation when solving an optimization problem, better implementation, object coded programming languages instead of Matlab, or dedicated optimization algorithms.

XI. Conclusions and discussion

We have considered the baggage handling process in large airports using destination coded vehicles (DCVs) running at high speeds on a network of tracks. A fast event-driven model of the continuous-time baggage handling process has been determined. Next we have considered the route choice control problem for each DCV transporting bags on the track network. In order to optimize the performance of the system, we have proposed predictive control methods that can be used to route the DCVs through the network (centralized, decentralized, and distributed predictive control) and two heuristic approaches.

The results show that the best performance of the system is obtained using centralized control. However, the centralized approach is not tractable in practice due to the large computational effort that this method requires. Decentralized approaches lower the computational effort, but using them would result in decrease of efficiency. The distributed approaches offer a balanced trade-off between the optimality of the system and the time required to compute the route choice.

In future work we will analyze an alternative approach for reducing the complexity of the computations by approximating the nonlinear route choice problem with linear ones by using mixed integer linear programming theory. First steps towards this direction are presented in [11] and [12]. Furthermore, in future work, we will also consider the line balancing problem (i.e. optimally assigning loading stations to the DCVs that unloaded their bag) and empty cart management (optimally route the empty DCVs towards the assigned loading stations). Finally, in future work we will also test these approaches on a full scale system.

Acknowledgments

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

References

[1] D. Šiljak, *Decentralized Control of Complex Systems*. San Diego, California, USA: Academic Press, 1991.

[2] G. Weiss, *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. Cambridge, Massachusetts, USA: The MIT Press, 2000.

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

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

[5] R. Findeisen, F. Allgöwer, and L. Biegler, Eds., *Assessment and Future Directions of Nonlinear Model Predictive Control*, ser. Lecture Notes in Control and Information Sciences. Berlin, Germany: Springer-Verlag, 2007, vol. 358.

[6] J. Richalet, A. Rault, J. Testud, and J. Papon, “Model predictive heuristic control: Applications to industrial processes,” *Automatica*, vol. 14, no. 1, pp. 413–428, Dec. 1978.

[7] C. Cutler and B. Ramaker, “Dynamic matrix control - a computer control algorithm,” in *Proceedings of Joint American Control Conference*, San Francisco, California, USA, Aug. 1980, paper WP5-B.

[8] Z. Nagya, B. Mahnb, R. Franke, and F. Allgöwer, “Evaluation study of an efficient output feedback nonlinear model predictive control for temperature tracking in an industrial batch reactorstar,” *IEEE Transactions on Control Systems Technology*, vol. 15, no. 7, pp. 839–850, Jul. 2007.

[9] P. Pardalos and M. Resende, Eds., *Handbook of Applied Optimization*. Oxford, UK: Oxford University Press, 2002.

[10] A. Tarău, “Model-based control for postal automation and baggage handling,” Ph.D. dissertation, Delft University of Technology, 2010.

[11] A. Tarău, B. De Schutter, and J. Hellendoorn, “Predictive route choice control of destination coded vehicles with mixed integer linear programming optimization,” in *Proceedings of the 12th IFAC Symposium on Control in Transportation Systems*, Redondo Beach, California, USA, Sep. 2009, pp. 64–69.

[12] A. Tarău, B. De Schutter, and H. Hellendoorn, “Hierarchical route choice control for baggage handling systems,” in *Proceedings of the 12th International IEEE Conference on Intelligent Transportation Systems*, St. Louis, Missouri, USA, Oct. 2009, pp. 679–684.