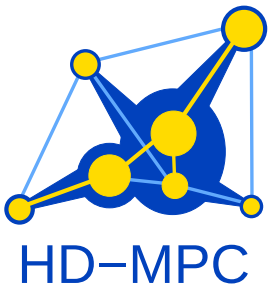
SEVENTH FRAMEWORK PROGRAMME
THEME – ICT
[Information and Communication Technologies]

Contract Number:	223854
Project Title:	Hierarchical and Distributed Model Predictive Control of Large-Scale Systems
Project Acronym:	HD-MPC



Deliverable Number:	D2.3
Deliverable Type:	Report
Contractual Date of Delivery:	March 1, 2010
Actual Date of Delivery:	**February 28, 2010**
Title of Deliverable:	**Final report on the results regarding multi-level models and architectures for hierarchical and distributed MPC**
Dissemination level:	Public
Workpackage contributing to the Deliverable:	WP2
WP Leader:	POLIMI
Partners:	TUD, EDF, POLIMI, UNC, SUPELEC, INOCSA
Author:	F.V. Arroyave, L.D. Baskar, D. De Vito, B. De Schutter, J. Espinosa Oviedo, J.F. Garcia, A. Marquez, B. Picasso, R. Scattolini, A.N. Tarău

Table of contents

Project co-ordinator

Name: Bart De Schutter
Address: Delft Center for Systems and Control
Delft University of Technology
Mekelweg 2, 2628 Delft, The Netherlands
Phone Number: +31-15-2785113
Fax Number: +31-15-2786679
E-mail: b.deschutter@tudelft.nl

Project web site: `http://www.ict-hd-mpc.eu/`

Executive Summary

This report describes the research activity in the Seventh Framework Programme, Theme 3 "Information and Communication Technologies", STREP research project **Hierarchical and Distributed Model Predictive Control of Large Scale Systems- HD-MPC**, focusing on work package WP2 – "Definition of the hierarchical architecture for control design". Specifically, Task 2.2 (Definition of the control architecture), Task 2.3 (Extension of the control architecture), and Task 2.4 (Multi-level models) are considered. First, a number of methods for the development and use of multi-level models with different kinds of spatial and temporal aggregation are described and some model reduction methods are reported. Then, it is shown how multilevel models can profitably be used in a couple of application fields. Finally, the topic of control system reconfiguration is addressed by considering a hierarchical structure where the system at the high level, with slower dynamics, is driven by a number of systems at the lower level, i.e., the actuators. By resorting to the MPC paradigm, it is proven that actuators' replacement or addition does not influence the convergence properties of the overall control structure.
The report is organized in four chapters:

- Chapter 1 presents a literature review about multi-model structures in model predictive control and model reduction. The methods surveyed complete and extend the results already presented in Deliverable D2.1. Three decomposition approaches are considered, namely functional, temporal and spatial decomposition, for the design of multi-level, multi-resolution MPC regulators. For each one of them, the main contributions proposed in the technical literature are reported and critically examined.

- By considering two different application fields, viz., intelligent vehicle highway systems and baggage handling systems — Chapter 2 shows how an efficient hierarchical control structure can be designed with the MPC approach applied to models with different levels of aggregation at the various levels of the control hierarchy. In Section 2.2 intelligent vehicle highway systems are considered, with emphasis on the control hierarchy and on the models used. Section 2.3 deals with the complete design of a hierarchical controller for a baggage handling system. In this case, a simulation example is also reported and discussed.

- Chapter 3 considers the design of a two-layer control architecture. The high layer of the control hierarchy corresponds to the system under control, while the lower layer represents the available actuators. Focus is centered on the possibility to reconfigure the control system by adding or substituting an actuator. This is of major importance within the project, and in particular with reference to Tasks 2.3 and 2.4 to meet the requirement of improving the availability of control schemes in response to changes in the subsystems. Section 3.1 presents the problem formulation, i.e., the model of the plant at the high and the low levels. Section 3.2 describes the MPC control synthesis technique adopted at the high level and a convergence result for the overall (high and low layers) system. Section 3.3 deals with the extension of these results to the case of system reconfiguration due to an actuator addition or substitution.

- Finally, in Chapter 4 the activity performed in work package WP2 is briefly summarized. This is done by recalling the WP tasks as well as the results achieved in the research activity and reported in Deliverables D2.1, D2.2, and D2.3.

Chapter 1

Multi-models and model reduction techniques for MPC

In this chapter, some approaches reported in the literature for the definition multi time-scales models and for the model reduction are surveyed. These results complete and extend those already presented in Deliverable D2.1. In this chapter it is also discussed how to use these models for the design of hierarchical MPC regulators. The main idea in these control schemes is to decompose the original control task into a sequence of simpler and hierarchically structured subtasks, handled by dedicated control layers, as proposed in [52]. The contribution of this chapter is due to the UNC research unit.

1.1 Multi-models for MPC

According to [52], there are three basic methods for hierarchical system decomposition:

1. Functional decomposition.
2. Temporal decomposition.
3. Spatial decomposition.

The first decomposition is based on the information flow along the system and the decisions taken based on this information. Basically three functions are identified: management, plant-wide control and direct control. Each one of these functions has a different model, depending on its objective. The second decomposition is based on the time response of the dynamics of the plant. In the temporal decomposition the model of the whole system is reduced based on the time scales of the dynamics of the system, and for each time scale a model predictive controller is designed. The third decomposition is based on the distribution of the system. In this case the whole system is divided into several places or regions and the models are developed based on the similarities among the subsystems belonging to the same place.
In the next sections, several approaches on each one of the ways to decompose the system to design a multi-model model predictive controllers are presented.

1.1.1 Functional decomposition approaches

In [52] the authors propose to control the whole system based on assigning a set of functionally different partial control objectives in a structure of vertical, hierarchical dependence. Figure 1.1 shows

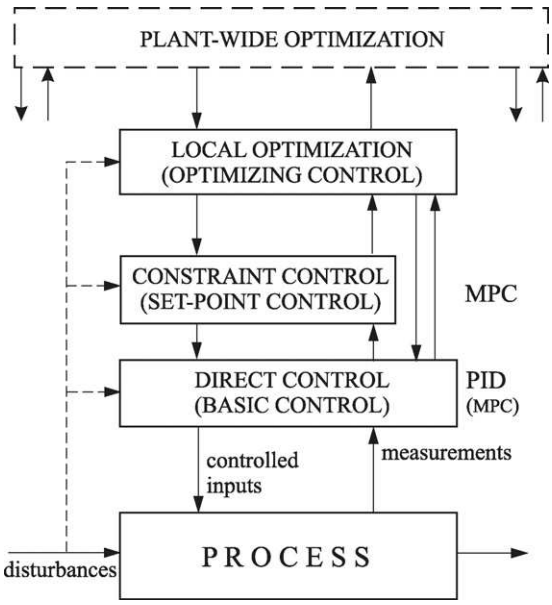


Figure 1.1: Hierarchical control scheme[52]

.

an schematic diagram of this proposed scheme.

In [27], the author proposes a dual-based optimization for production, supply and inventor plan. The proposed optimization scheme has two steps:

1. Long-term plan: it is based on the predicted demands represented by a random sequence. The role of the long-term plan is to impose a final condition set to the short-term plan.

2. Short-term plan: it is based on the firm orders received from the customers.

In [56], the authors propose a multi-level model predictive control scheme based on the use of multi-form models. In this approach, the plant is represented by a variety of models for different end-uses, including:

1. A distributed parameters model.

2. A lumped parameters model.

3. A matrix representation with off-line computed matrix elements.

4. Local linear models.

5. Reduced order models obtained with balanced truncation methods.

It can be shown through dynamic simulations that significant reductions in computing time can be achieved with properly selecting the model forms. Since both open-loop optimal control and closed loop MPC rely on iterative dynamic optimization, overall computing time reduction makes on-line applications possible [56].

In [54], the authors propose a two-layer production control method. A predictive controller is proposed as a coordinator in the highest layer and a distributed control policy is used as a follow-up

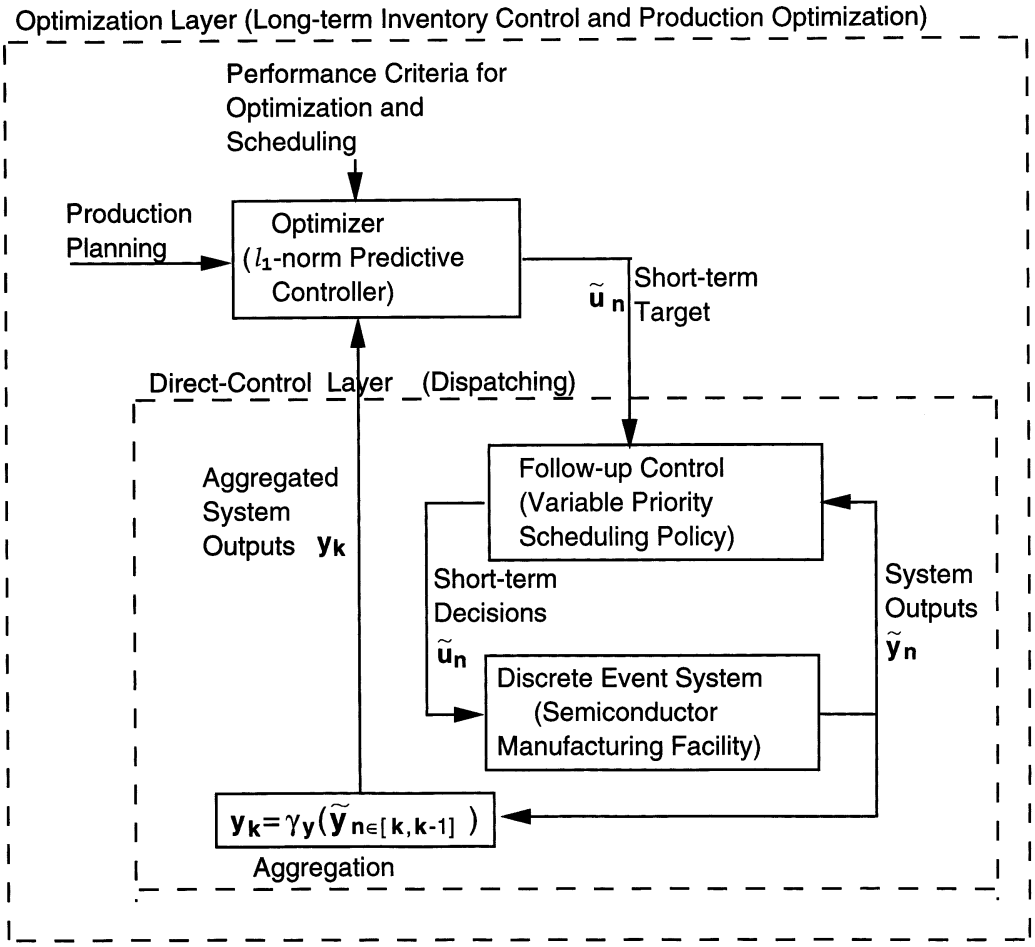


Figure 1.2: Two-layer model predictive control scheme [54]

.

controller in the lowest layer. The use of a model predictive control formulation allows the scheduling algorithm to simultaneously solve the production optimization and in-process inventory control problems at each sampling time. In Figure 1.2, a scheme of the proposed control structure is displayed: a linear state-space model of the system is used to compute the optimal reference values in the optimization layer.

1.1.2 Temporal decomposition approaches

In [59] the authors propose a multi-rate linear model predictive controller in order to avoid the troubles associated with the delay in the measurements of some important variables in the control of a Weyerhaeuser digester. Dynamic linear models between the inputs, the disturbances and the selected manipulated variables are used. An identification procedure based on normalized moments of an impulse response is proposed to identify general linear models of the form:

$$G(s) = \frac{K\zeta s + K}{a_3 s^3 + a_2 s^2 + a_1 s + 1} e^{t_a s} \tag{1.1}$$

In [46], multi-scale models are used to design model-predictive controllers, resulting in design techniques with several important advantages, such as:

1. Natural depiction of performance characteristics and treatment of output constraints.
2. Fast algorithms for establishing the constrained control policies over long prediction/control horizons.
3. Rich depiction of feedback errors at several scales.
4. Optimal fusion of multi-rate measurements and control actions.

The main idea is to provide an alternative framework to design a model predictive controller, in which the representation of the system captures the scale characteristics of the system. To do that, the authors propose to decompose the whole system model using binary trees. The procedure can be described as follows. Consider the discrete linear state-space model:

$$x(k+1) = Ax(k) + Bu(k) \tag{1.2}$$

Taking the expression (1.2) as the level zero in the binary tree, the state space model (1.2) becomes:

$$x(0,k+1) = Ax(0,k) + Bu(0,k) \tag{1.3}$$

Then, for any left-node τ at the level one in the binary tree, the model (1.2) can be transformed to the following two-scale model

$$x(\tau) = \sqrt{2}(I+A)^{-1}(I-A)x(\alpha\tau) - (I+A)^{-1}Bu(\alpha\tau) \tag{1.4}$$

or equivalently

$$\delta x(\tau) = (I+A)^{-1}(I-A)x(\alpha\tau) - \sqrt{2}(I+A)^{-1}Bu(\alpha\tau) \tag{1.5}$$

where $\delta x(\tau)$ is the Haar wavelet coefficient of the state at the level τ, $x(\tau)$ is the Haar scaling coefficient of the state at the node τ, and $x(\alpha\tau)$, $u(\alpha\tau)$ are the values of the states and control inputs at the node $\alpha\tau$ which is the left-offspring of the node τ.

With this formulation, it can be shown that the dynamics of the system at level one are governed by the following discrete-time model:

$$x(-1,\frac{k}{2}+1) = A^2x(-1,\frac{k}{2}) + (I+A)Bu(-1,\frac{k}{2}) \tag{1.6}$$

where

$$Bu(-1,\frac{k}{2}) = \frac{1}{\sqrt{2}}(I+A)^{-1}[ABu(0,k) + (I+A)Bu(0,k+1) + Bu(0,k+2)] \tag{1.7}$$

Based on (1.6), the multi-scale model for any left-node at levels $-2\ -3\ \ldots -m$ are generated recursively by

$$x(\tau\alpha) = \sqrt{2}(I+A^{2^m})^{-1}x(\tau) - B^{(m)}u(\tau\alpha) \tag{1.8}$$

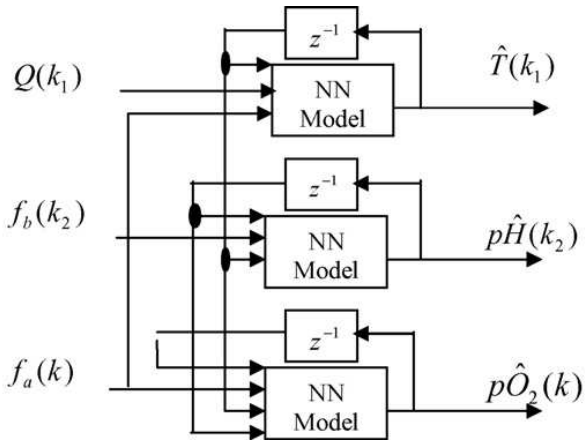


Figure 1.3: Multi-model prediction scheme [62]

where $B^{(m)} = (I + A^{2^m})^{-1}(I + A^{2^{m-1}}) \dots (I + A)B$. Thus, the process model given by (1.2) describing the process over 2^N points, is transformed to a multi-scale model given by (1.8), defined over 2^N left-nodes, τ, of a binary tree, where each node describes process behavior over localized sections of time and scale.

In [62], a multi-rate control strategy is proposed based on multi-rate models, which allows to reduce the dimensionality of the optimizations. To do that, three multi-input single-output neural network models are developed with different sampling times. Each neural network can be represented as a *NARX* model of the form

$$y(k) = f(y(k-1), \dots, y(k-n_y), u(k-1-d), \dots, u(k-n_u-d)) + \varepsilon(t) \tag{1.9}$$

Since the system outputs are coupled, the estimation of each neural network is used to compute the output of the complementary output variables. The interaction among neural networks and the complete scheme used to compute the optimal input is shown in Figure 1.3.

The algorithm to compute the optimal input is illustrated in Figure 1.4.

In [18], another multi-rate model predictive control algorithm is presented. In this work, the problem of dimensionality is avoided using the principal component analysis. The reduced model is linearized and the resulting linear state-space representation of the system is used to compute the model predictive control output. Since the states have different time responses and the measurements have different arrival times, two extended Kalman filters are used: one to estimate the slow dynamics based on the fast states and fast measurements, the other to improve the first estimation based on the information of the slow measurements.

In [57], an iterative learning model predictive controller is proposed for batch processes.

Finally, in [60], a multi-objective optimization problem is considered. The proposed scheme uses a reduced order *MIMO* model of the system, with two inputs and two outputs, to estimate the optimal input. The model reduction is carried out assuming that the fast dynamics are in steady-state.

1.1.3 Spatial decomposition approaches

In [61], a cascade model predictive control scheme is proposed. In this scheme, the internal and the external control loops are designed with MPC. Figure 1.5 shows schematically the proposed control strategy.

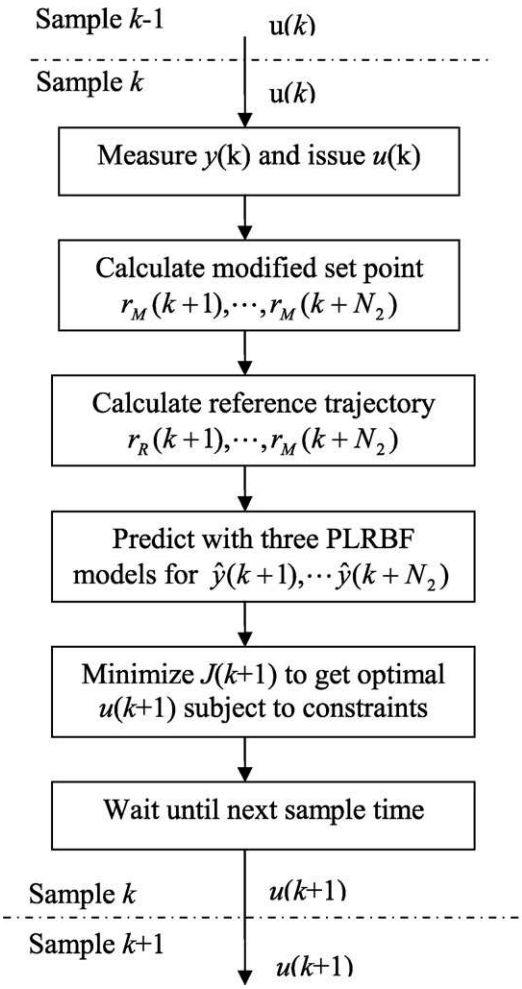


Figure 1.4: Algorithm to compute the optimal input using the multi-model prediction scheme [62]

.

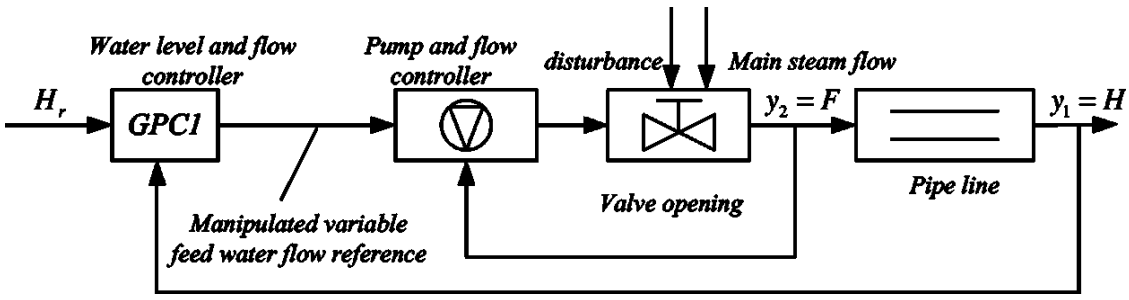


Figure 1.5: Cascade Model Predictive Control Scheme [61]

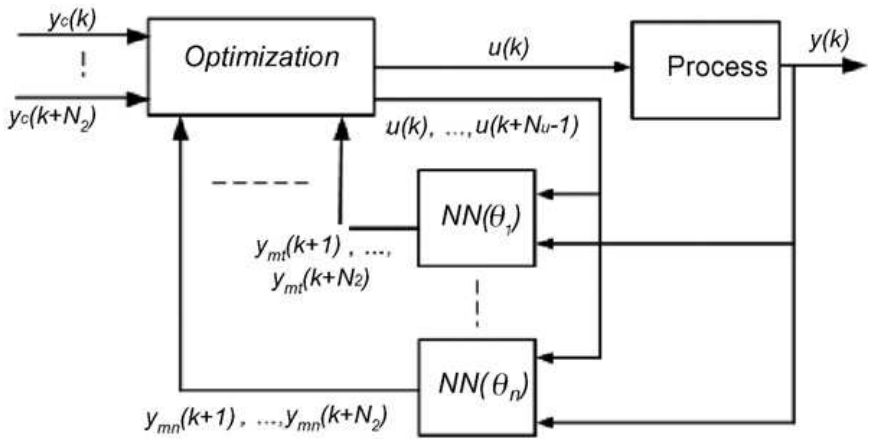


Figure 1.6: Multi-criteria model predictive control scheme based on neural networks [34]

.

In this scheme, the *CARIMA* black-box model (1.10) is used for the design of each model predictive controller to compute the optimal input.

$$A(q^{-1})y(k) = B(q^{-1})u(k-d) + C(q^{-1})\frac{\xi(k)}{\Delta} \tag{1.10}$$

where y and u are the output and input, respectively, d is the input delay, Δ is a difference operator $1-q^{-1}$, and $\xi(t)$ is an uncorrelated random noise sequence with zero mean. $A(.),B(.),C(.)$ are polynomials in the backward shift operator q^{-1}.

In [34] multi-criteria optimization is used to design a predictive controller for nonlinear dynamical systems. Artificial neural networks and genetic algorithms are considered. Neural networks are used to determine process models at each operating level and an aggregation method based on a genetic algorithm is used to solve the multi-criteria optimization problem. To carry out the proposed control scheme, a generic nonlinear system of the form

$$y(k) = g(y(k-1),\ldots,y(k-m_1),u(k),\ldots,u(k-m_2)) \tag{1.11}$$

is considered, being $g(.)$ an unknown nonlinear function. To identify $g(.)$ in each operating region, a multilayer perceptron neural network (NN) is used. Thus the output of the system becomes

$$y(k) = NN(u(k),\theta_i) \tag{1.12}$$

where θ_i denotes the parameters of the neural network in the $i-th$ operating region. The scheme of the proposed control strategy is shown in Figure 1.6, where the optimization block represents the genetic algorithm used for multi-criteria optimization.

1.2 Model reduction

Spatially distributed systems such as tubular reactors or reactor networks hosting multiple autocatalytic species demonstrate a rich spectrum of complex behavior. From a control systems perspective, these systems offer a difficult control challenge because of their distributed nature, nonlinearity, and high order. Furthermore, manipulation of the network states may require simultaneous control actions

Table 1.1: Approximation methods

Approximation methods for dynamical systems		
SVD		**Krylov**
Nonlinear systems	Linear systems	
POD methods Empirical grammians	Balanced truncation Hankel approximation	Realization Interpolation Lanczos Arnoldi

in different parts of the system and require to pass through several operating regimes to achieve the desired operation, [51]. These facts justify the need to use reduced models in order to improve the numerical stability and shorten the computational efforts in the design of MPC regulators.

1.2.1 Approximation of large-scale dynamical systems

Approximation methods can be cast into three broad categories: (a) SVD based methods, (b) Krylov based methods, (c) Iterative methods combining aspects of both the SVD and Krylov methods.
The SVD-based approximation methods have their roots in the Singular Value Decomposition and the resulting solution of the approximation of matrices by means of matrices of lower rank, which are optimal in the 2-norm (or more generally in unitarily invariant norms). The quantities which are important in deciding to what extent a given finite-dimensional operator can be approximated by one of lower rank are the so-called singular values.
Krylov-based approximation methods do not rely on the computation of singular values. Instead they are based on moment matching of the impulse response of the dynamical system. Two widely used methods fall under this category, namely the Lanczos and the Arnoldi procedures, which were put forward by C. Lanczos in 1950 and by W.E. Arnoldi in 1951, respectively. These methods have been very influential in iterative eigenvalue computations and more recently in model reduction. Their drawbacks are that the resulting reduced order systems have no guaranteed error bound, stability is not necessarily preserved and some of them are not automatic. Table 1.1 summarizes the approximation methods considered in the following.

Balancing and Hankel Norm Approximation

In this section we describe some of the most relevant and popular methods to reduce the complexity of models. It is assumed that a (stable) linear time-invariant system is given and we address the problem to approximate this system by a less complex (simpler) one. The approximate system is required to have a dynamic behavior which is similar, or as close as possible, to the behavior of the system which we wish to approximate.

State truncations [58]

Consider a (continuous-time or discrete-time) dynamical system in input-state-output form:

$$\begin{aligned} x(t+1) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned} \tag{1.13}$$

Suppose that the state x of this system is partitioned in two components as:

$$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \tag{1.14}$$

Any such partitioning causes a compatible partitioning of the system matrices as follows

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}, B = \begin{pmatrix} B_1 \\ B_2 \end{pmatrix}, C = \begin{pmatrix} C_1 & C_2 \end{pmatrix} \tag{1.15}$$

If we assume that the vector x_1 has dimension k, then $A_{11} \in R^{k\times k}$, $B_1 \in R^{k\times m}$ $C_1 \in R^{p\times k}$ and $D \in R^{p\times m}$. We will view the quadruple (A_{11}, B_1, C_1, D) as a k-th order truncation of (A, B, C, D). This k-th order truncation defines a system in input-state-output form:

$$\begin{aligned} \xi(t+1) &= A_{11}\xi(t) + B_1 u(t) \\ y(t) &= C_1\xi(t) + Du(t) \end{aligned} \tag{1.16}$$

which will be viewed as the k-th order truncation of the system defined by (1.13). Note that the state variable ξ in (1.16) has dimension k but is not the same as x_1 . Also, note that any system theoretic property (like stability, controllability, minimality, etc.) which the system (1.13) may have, may not be inherited by the truncated system (1.16). In particular, the system (1.16) may not be stable, may not be minimal or dissipative while the system described by (1.13) may have these properties.

Modal truncations [58]

Consider a state space transformation:

$$x = Tx' \tag{1.17}$$

for the system (1.13) with T a non-singular matrix of dimension $n \times n$. Since such a transformation only amounts to rewriting the state variable in a new basis, it is well known that this transformation does not affect the input-output behavior associated with (1.13). Thus,

$$\begin{aligned} x'(t+1) &= T^{-1}ATx'(t) + T^{-1}Bu(t) \\ y(t) &= CTx'(t) + Du(t) \end{aligned} \tag{1.18}$$

The non-singular transformation T can be computed so that the resulting system is in the Jordan canonical form. Now, suppose that the system (1.13) is stable. This implies that the absolute values $|\lambda_i| < 1$,for all $i = 1, ..., n$. Without loss of generality we may therefore order the natural frequencies according to

$$0 \leq |\lambda_1| \leq |\lambda_2| \leq ..... \leq |\lambda_n| < 1 \tag{1.19}$$

With this ordering, the states x' of a modal canonical form are ordered so that the first components of the states x' correspond to low frequency (or slow) modes, and the last components of the state vector x' correspond to high frequency (fast modes). If we partition

$$x' = \begin{pmatrix} x'_1 \\ x'_2 \end{pmatrix} \tag{1.20}$$

where x_1' has dimension $k < n$, then the truncated system is defined by leaving out the fast modes of the system.

Balanced truncations [58]

This method, extensively described in [58], is based on a balanced state space representation that eliminate the poorly reachable and poorly observable states from a state space model.

Hankel norm reductions

The Hankel norm of a continuous-time system described in state-space form by the matrices (A,B,C,D) is defined by:

$$\|G\|_H^2 := \sup_{u\in\mathscr{L}_2(-\infty,0]} \frac{\int_0^\infty y(t)^2 dt}{\int_0^\infty u(t)^2 dt} \tag{1.21}$$

where

$$y(t) = \int_{-\infty}^{0} Ce^{A(t-s)}Bu(s)ds \tag{1.22}$$

The Hankel norm is a measure of the energy can be transferred from past inputs into future outputs through the system. The Hankel-norm model reduction problem is defined as:

Given an n-th order stable system G, find a k-th order stable system G_k so as to minimize the Hankel norm of the error $\|G - G_k\|_H$.

The algorithm to find the reduced order model by means of Hankel norm is described in [58].

Proper Orthogonal Decomposition-POD

Proper orthogonal decomposition and Galerkin projection are two well known techniques that have been used together for deriving reduced order models of high-dimensional systems. These high-dimensional systems are typically obtained after discretizing in space the partial differential equations that model many processes. In the POD method, an orthonormal basis for modal decomposition is extracted from an ensemble of data (called snapshots) obtained in the course of experiments or numerical simulations. The basis functions calculated with the POD technique are commonly called empirical eigenfunctions, empirical basis functions, empirical orthogonal functions, Proper Orthogonal Modes (POMs) or basis vectors. The POD method provides an orthonormal basis and also a measure of the importance of each basis vector. This measure of importance is sometimes referred to as Proper Orthogonal Value (POV) . Now, if we select the most relevant basis vectors and project (Galerkin projection) the original high-dimensional model on the space spanned by this subset, then we can obtain a reduced order model of the process. The most striking feature of the POD method is its optimality: "it provides the most efficient way of capturing the dominant components of an infinite-dimensional process with only a finite number of modes, and often surprisingly few modes [1]".

Now let $x(t) \in \Re^N = [x_1(t), x_2(t), ..., x_N]^T$ be the state vector of a given dynamical system, and let $X \in \Re^{N\times Nd}$ with $N_d \geq N$ be the so-called snapshot matrix that contain a finite number of samples or snapshots of the evolution of $x(t)$ at $t = t_1, t_2, ..., t_{N_d}$. In POD, we start by observing that each

snapshot can be written as a linear combination of a set of ordered orthonormal basis vectors (POD basis vectors) $\varphi_j \in \Re^N, \forall j = 1,2,...,N$:

$$x(t_i) = \sum_{j=1}^{2N} a_j(t_i)\varphi_j, \forall i = 1,2,...,N_d \tag{1.23}$$

where $a_j(t_i)$ is the coordinate of $x(t_i)$ with respect to the basis vector φ_j (it is also called time-varying coefficient or POD coefficient). Since the first n most relevant basis vectors capture most of the energy in the data collected, we can construct an nth order approximation of the snapshots by means of the following truncated sequence

$$x(t_i) = \sum_{j=1}^{n} a_j(t_i)\varphi_j, \forall i = 1,2,...,N_d, n \ll 2N \tag{1.24}$$

This is the essence of model reduction by POD. In POD, the orthonormal basis vectors are calculated in such a way that the reconstruction of the snapshots using the first n most relevant basis vectors is optimal in the sense that the Sum-Squared-Error (SSE) between $x(t_i)$ and $x_n(t_i)$, $\forall i = 1,...,N_d$. The POD basis Functions are determined from simulation or experimental data (Snapshot matrix) of the process. The dynamic model for the first n time varying coefficients can be found by means of the Galerkin projection [2] or using subspace identification techniques [29] .
The derivation of a reduced order model of (1.13) is done in the following steps.

A. ***Generation of the Snapshot Matrix***. A snapshot matrix $X_{snap} \in \Re^{N \times N_d}$ is created from the system response when independent step changes are made in the input $u(t)$ and perturbation $d(t)$ signals.

$$X_{snap} = [x(t = \Delta t), x(t = 2\Delta t), ..., x(t = N_d \Delta t)]$$

B. ***Derivation of the POD basis vectors***. The POD basis vectors are obtained by computing the SVD of the snapshot matrix X_{snap}

$$\mathbf{X_{snap}} = \Phi \Sigma \Psi^T$$

where $\Phi \in \Re^{N \times N}$ and $\Psi \in \Re^{N_d \times N_d}$ are unitary matrices, and $\Sigma \in \Re^{N \times N_d}$ is a matrix that contains the singular values of X_{snap} in a decreasing order on its main diagonal. The left singular vectors, i.e., the columns of Φ

$$\Phi = [\varphi_1, \varphi_2, ..., \varphi_N]$$

are the POD basis vectors.

C. ***Selection of the most relevant POD basis vectors***. The singular values of X_{snap} are checked. The larger the singular value the more relevant the basis function is. The n-th order approximation of $x(t)$ is given by

$$x(t_i) = \sum_{j=1}^{n} a_j(t_i)\varphi_j = \Phi_n \mathrm{a(t)} \tag{1.25}$$

where $\Phi_n = [\varphi_1, \varphi_2, ..., \varphi_n]$ and $\mathrm{a(t)} = [a_1(t), a_2(t), ..., a_n(t)]^T$

D. ***Construction of the model for the first n POD coefficients***. The Galerkin projection is used to derive the dynamical model for the POD coefficients as follows. Define a residual function $R(x)$ for equation (1.13) as:

$$R(x) = \dot{x}(t) - Ax(t) - Bu(t), \tag{1.26}$$

and replace $x(t)$ by its nth order approximation $x_n(t) = \Phi_n$a(t), the Galerkin projection sets that the projection of $R(x_n)$ on the space spanned by the basis functions Φ_n vanishes.

Empirical Grammians

Consider a continuous-time, nonlinear controlled dynamical system:

$$\begin{aligned} \dot{x} &= f(x(t), u(t)) \\ z(t) &= h(x(t)) \end{aligned} \tag{1.27}$$

In [35], authors define a new resolution technique for such systems which rely on classical model reduction, but introduces a balancing algorithm in order to deal with nonlinearities. Balancing means to apply a kind of linear transformations to two different matrices (here, the grammians) to obtain in booth cases the same diagonal matrix. This technique includes several steps:

1. To evaluate the (discrete) empirical grammians.
2. To balance both empirical grammians and to evaluate the squared eigenvalues of the common diagonal matrix (Hankel singular values).
3. According to the magnitudes of the eigenvalues, to choose the rank of the projection subspace.
4. To solve the reduced model obtained by Galerkin projection onto a suitable subspace.

The construction of empirical grammians depends on some parameters:

1. n, the number of states; and p, the number of inputs;
2. $\mathbb{T}^r = \{T_1, ..., T_r\}$, a set of orthogonal $n \times n$ matrices that will span the perturbation directions;
3. $\mathbb{M} = \{c_1, ..., c_s\}$, a set of s positive constants (the different sizes of the perturbations);
4. $\mathbb{E}^p$, the set of standard unit vectors in $\Re^p$.

Let $x^{ilm}(t)$ be the state corresponding to the impulsive input $u(t) = c_m T_l e_i \delta(t)$. Recall the definition of the temporal mean of any function $g(t)$:

$$\overline{g}(t) = \lim_{T \to \infty} \frac{1}{T} \int_0^T g(t)dt \tag{1.28}$$

From a theoretical point of view, empirical grammians have the following definition:

1. *Controllability empirical grammian*

$$G_c =: \sum_{i=1}^{p} \sum_{l=1}^{r} \sum_{m=1}^{s} \frac{1}{rsc_m^2} \int_0^\infty (x^{ilm}(t) - \bar{x}^{ilm})(x^{ilm}(t) - \bar{x}^{ilm})^T dt \tag{1.29}$$

2. *Observability empirical grammian*

$$G_o =: \sum_{l=1}^{r}\sum_{m=1}^{s}\frac{1}{rsc_m^2}\int_0^\infty T_l\Psi^{lm}T_l^T\,dt \tag{1.30}$$

where Ψ^{lm} is the $n\times n$ matrix given by

$$(\Psi^{lm})_{i,j} = (z^{ilm}(t)-\bar{z}^{ilm})^T(z^{ilm}(t)-\bar{z}^{ilm}) \tag{1.31}$$

For practical application, one should use a finite sum of a sampled trajectory instead an infinite integral. Also, temporal means are replaced by steady states. The later implies that one should know an input reference u_{ss} , (probably suggested by physical meaning of the underlying problem). Then, the steady state x_{ss} is obtained from $f(x_{ss}(t),u_{ss}(t))=0$ and the corresponding output is denoted by z_{ss} .

1.3 Conclusions

In this chapter, a literature review about multi-model structures in model predictive control have been presented. From this review it is possible to conclude that multi-model structures are an interesting tool to face-up complex, large-scale, control problems. Moreover, due to system decomposition, these control structures allow one to improve the performance of model predictive controllers by increasing the details of whole system model. Finally, system decomposition allows one to reduce the computational cost associated with the optimization problem in model predictive controllers, because it is possible to divide the whole system optimization problem into several ones.

Chapter 2

Design of hierarchical model predictive control systems with multi-resolution models

2.1 Introduction

The effectiveness of an efficient and tractable hierarchical MPC approach relying on the use of models with different levels of aggregation is discussed in two significant application fields, viz., intelligent vehicle highway systems and baggage handling systems. The analysis of these case studies is also useful to draw quite general guidelines for the design of hierarchical and multilevel control systems in a wide number of applications.

This chapter is organized as follows. In Section 2.2 intelligent vehicle highway systems are considered, while in Section 2.3 baggage handling systems are studied. The emphasis in Section 2.2 is mainly on the control hierarchy and the models used, whereas in Section 2.3 a worked example is considered to illustrate the adopted hierarchical control approach.

The work reported in this chapter is based on the PhD theses of Lakshmi Baskar [6] and Alina Tarău [48].

2.2 Multi-resolution models in hierarchical control for intelligent vehicle highway systems

In this section we propose a hierarchical control framework for intelligent vehicle highway systems. This framework consists of several levels, where at each level different types of models are used by the controller, depending on the temporal scale and spatial scale at which the given controller operates.

This section is organized as follows. In Section 2.2.1 we introduce Intelligent Vehicle Highway Systems (IVHS). We recapitulate the hierarchical traffic management and control framework of [5] in Section 2.2.2. In Section 2.2.3 we report on vehicle and traffic models. In Section 2.2.4 we propose an MPC method for the roadside controllers to determine optimal speeds, lane allocations, and on-ramp release times for the platoons. Next, we focus on the route guidance tasks of the area controllers and we present a simplified flow model and the corresponding optimal route guidance problem in Section 2.2.5. We consider both the static (constant demands) and the dynamic case (time-varying demands). In general, the dynamic case leads to a nonlinear non-convex optimization problem, but in

Section 2.2.5 we show that this problem can be approximated using mixed integer linear programming (MILP). Section 2.2.6 concludes the worked example.

2.2.1 Intelligent vehicle highway systems (IVHS)

Introduction

The recurring traffic congestion problems and their related costs have resulted in various solution approaches. One of these involves the combination of the existing transportation infrastructure and equipment with advanced technologies from the field of control theory, communication, and information technology. This results in integrated traffic management and control systems, called Intelligent Vehicle Highway Systems (IVHS), that incorporate intelligence in both the roadside infrastructure and in the vehicles. Although this step is considered to be a long-term solution, this approach is capable of offering significant increases in the performance of the traffic system [47, 30, 21].

In IVHS all vehicles are assumed to be fully automated with throttle, braking, and steering commands being determined by automated on-board controllers. Such complete automation of the driving tasks allows to organize the traffic in platoons, i.e., a closely spaced group of vehicles traveling together with short intervehicle distances [53, 45]. Platoons can travel at high speeds and to avoid collisions between platoons at these high speeds, a safe interplatoon distance of about 20–60 m should be maintained. Also, the vehicles in each platoon travel with small intraplatoon distances of about 2–5 m, which are maintained by the automated on-board speed and distance controllers. By traveling at high speeds and by maintaining short intraplatoon distances, the platoon approach allows more vehicles to travel on the network, which improves the traffic throughput [12, 38].

Intelligent vehicles and IV-based traffic control measures

Intelligent Vehicles (IVs) are equipped with control systems that can sense the environment around the vehicle and that result in a more efficient vehicle operation by assisting the driver or by taking partial or complete control of the vehicle [10]. The platoon-based approach used in this paper assumes that all IVs are fully autonomous, i.e., complete control is taken of the vehicle operation.

There are several IV technologies that support and improve the platooning concept by allowing vehicle-vehicle and vehicle-roadside coordination [10, 13]:

- Intelligent Speed Adaptation (ISA),
- Adaptive Cruise Control (ACC),
- dynamic route planning and guidance.

In this section we will focus on ISA and ACC.

ISA is based on a speed limiter incorporated within each vehicle that can take into account speed limit restrictions, that can adjust the maximum driving speed to the speed limit specified by the roadside infrastructure, and that can provide feedback to the driver or take autonomous action when that speed limit is exceeded. ISA systems could use fixed or dynamic speed limits. In the fixed case, the driver is informed about the speed limit, which could be obtained from a static database. Dynamic speed limits take into account the current road conditions such as bad weather, slippery roads, or major incidents before prescribing the speed limit.

An ACC system is a radar-based system that extends conventional cruise control and that is designed to monitor the immediate predecessor vehicle in the same lane, and to automatically adjust

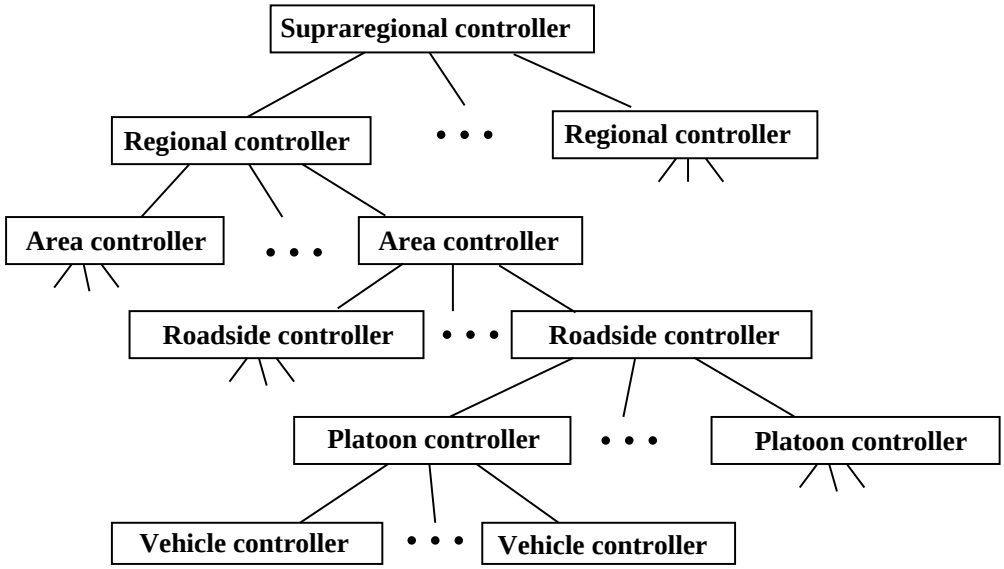


Figure 2.1: The hierarchical control framework for IVHS.

the speed of the equipped vehicle to match the speed of the preceding vehicle and to maintain a safe intervehicle distance [16]. Cooperative ACC is a further enhancement of ACC systems that uses wireless communication technologies to obtain real-time information about the speed, acceleration, etc. of the preceding vehicle. Vehicles equipped with cooperative ACC can exchange the information much quicker and allow to set the safe minimum time headway as small as 0.5 s. Hence, with reduced headways between vehicles, the maximal traffic flow can be augmented even further.

2.2.2 Hierarchical control of IVHS

In [5] a hierarchical traffic management and control framework for IVHS is proposed that builds upon earlier research in this field such as the PATH framework [45]. The control architecture of [5] consists of a multi-level control structure with local controllers at the lowest level and one or more higher supervisory control levels (see Figure 2.1).

We now briefly present the hierarchical control framework for IVHS developed in [5]. This framework is based on the platoon concept and it distributes the intelligence between the roadside infrastructure and the vehicles using control measures such as intelligent speed adaption, adaptive cruise control, lane allocation, on-ramp access control, route guidance, etc. to prevent congestion and to improve the performance of the traffic network. The control architecture of [5] consists of a multi-level control structure with local controllers at the lowest level and one or more higher supervisory control levels as shown in Figure 2.1. The layers of the framework can be characterized as follows:

- The *vehicle controllers* present in each vehicle receive commands from the platoon controllers (e.g., set-points or reference trajectories for speeds (for intelligent speed adaption), headways (for adaptive cruise control), and paths) and they translate these commands into control signals for the vehicle actuators such as throttle, braking, and steering actions.

- The *platoon controllers* receive commands from the roadside controllers and are responsible for control and coordination of each vehicle inside the platoon. The platoon controllers are mainly concerned with actually executing the interplatoon maneuvers (such as merges with

other platoons, splits, and lane changes) and intraplatoon activities (such as maintaining safe intervehicle distances).

- The *roadside controllers* may control a part of a highway or an entire highway. The main tasks of the roadside controllers are to assign speeds for each platoon, safe distances to avoid collisions between platoons, appropriate platoon sizes, and ramp metering values at the on-ramps. The roadside controllers give instructions for merging, splitting, and lane changes to the platoons.

- The *higher-level controllers* (such as area, regional, and supraregional controllers) provide network-wide coordination of the lower-level and middle-level controllers. In particular, the area controllers provide area-wide dynamic route guidance for the platoons, and they supervise and coordinate the activities of the roadside controllers in their area by providing set-points and control targets. In turn, a group of area controllers could be supervised or controlled by a regional controller, and so on.

The lower levels in this hierarchy deal with faster time scales (typically in the milliseconds range for the vehicle controllers up to the seconds range for the roadside controllers), whereas for the higher-level layers the frequency of updating can range from few times per minute (for the area controllers) to a few times per hour (for the supraregional controllers).

2.2.3 Vehicle and traffic modeling at the roadside level

There exists a wide range of traffic models [14]. An important factor that determines the choice of the model to be used in MPC is the trade-off between accuracy and computational complexity since at each time step the model will be simulated repeatedly within the on-line optimization algorithm. As a consequence, very detailed microscopic traffic simulation models are usually not suited as MPC prediction model. Instead, simplified or more aggregate models are usually applied. Therefore, we now describe simplified traffic models for vehicles and for platoons that can be used as (part of the) prediction model within the MPC-based roadside controller. We consider both human drivers and IV models. In Section 2.2.5 we will then also present more aggregate models that can be used by the area controllers.

Traffic flow modeling

In this section, we deal with the longitudinal aspects of the driver tasks, which can be classified as follows:

- free-flow behavior,
- car-following behavior,
- stop-and-go behavior.

In free-flow behavior, the vehicles can travel at their desired speed (corresponding to the speed limit, e.g., 120 km/h). As the traffic demand increases, the vehicles start to follow their predecessors at closer distances and at reduced speeds (50–80 km/h). Once the capacity of the highway is being utilized at its maximum, then the vehicles move with stop-and-go movements (0–40 km/h).

Vehicle models

We use general kinematics motion equations to describe the dynamics of the vehicles, which, after discretization leads to:

$$x_i(\ell) = x_i(\ell-1) + v_i(\ell-1)T_{\text{sim}} + 0.5a_i(\ell-1)T_{\text{sim}}^2 \tag{2.1}$$

$$v_i(\ell) = v_i(\ell-1) + a_i(\ell-1)T_{\text{sim}} \tag{2.2}$$

where ℓ is the simulation step counter, T_{sim} the simulation time step, x_i the longitudinal position of vehicle i, v_i the speed of vehicle i, and a_i the acceleration of vehicle i. The acceleration used in (2.1)–(2.2) is calculated according to the current driving situation as will be explained below. Also, the acceleration is limited between a maximum acceleration and a maximum (in absolute value) comfortable deceleration.

We first consider models for human drivers. Next, we discuss models for the intelligent vehicles and for the platoons. We conclude with a description of a phenomenon called capacity drop.

Longitudinal models for human drivers

The time headway T_{head} of a vehicle is defined as the time difference between the passing of the rear ends of the vehicle's predecessor and the vehicle itself at a certain location. When there is no predecessor or when the time headway to the predecessor is larger than the critical time headway (e.g., 10 s), then the vehicle is said to be in free-flow mode. Once the vehicle travels with a smaller time headway than the critical time headway to its predecessor, then the vehicle is said to be in car-following mode.

Free-flow model

The acceleration for free-flow driving conditions is determined by the delayed difference between the current speed and the reference speed:

$$a_i(\ell) = K(v_{\text{ref},i}(\ell-\sigma) - v_i(\ell-\sigma)) \tag{2.3}$$

where K is the proportional constant, $v_{\text{ref},i}$ is the reference speed, and σ is the reaction delay[1]. The reference speed can either be issued by roadside infrastructure or it can be driver's desired maximum speed.

Car-following model

As described in [11] there exist various types of car-following models such as stimulus response models [41], collision avoidance models [33], psychophysical models [42], and cellular automata models [43].

We will use a stimulus response model to describe the behavior of human drivers as this model is most often used and also easy to implement. Stimulus response models are based on the hypothesis that each vehicle accelerates or decelerates as a function of the relative speed and distance between the vehicle and its predecessor. In particular, the Gazis-Herman-Rothery (GHR) model [24] states that after a reaction delay, the follower vehicle i accelerates or decelerates in proportion to the speed of the

[1] We assume here that the reaction time T_{react}, which typically has a value of 1–1.2 s, is an integer multiple of the simulation time step T_{sim}. So, $T_{\text{react}} = \sigma T_{\text{sim}}$ with σ an integer.

vehicle itself, to the relative speed with respect to its predecessor (vehicle $i+1$), and to the inverse of distance headway between them. The reference acceleration is thus given by

$$a_i(\ell) = C v_i^{\beta}(\ell) \frac{(v_{i+1}(\ell-d) - v_i(\ell-d))}{(x_{i+1}(\ell-d) - x_i(\ell-d))^{\gamma}} \tag{2.4}$$

where C, β, and γ are the model parameters (possibly with different values depending on whether the vehicle is in a congested and uncongested driving situation), and d is the driver delay[2].

Longitudinal models for intelligent vehicles

In our approach, intelligent vehicles will use ACC and ISA measures and are arranged in platoons. We now discuss how the accelerations for the platoon leaders and for the follower vehicles within a platoon are calculated.

Platoon leader model

Platoon leaders have an enforced-ISA system and the calculation of the acceleration for the platoon leader is based on a simple proportional controller:

$$a_i(\ell) = K_1(v_{\mathrm{ISA}}(\ell) - v_i(\ell)) \tag{2.5}$$

where K_1 is the proportional constant, and v_{ISA} is the reference ISA speed provided by the roadside controller.

Follower vehicle model

The follower vehicles will use their on-board ACC system to maintain short intraplatoon distances. The ACC algorithm consists of a combined speed and distance controller:

$$a_i(\ell) = K_2(h_{\mathrm{ref},i}(\ell) - (x_{i+1}(\ell) - x_i(\ell))) + K_3(v_{i+1}(\ell) - v_i(\ell)) \tag{2.6}$$

where K_2 and K_3 are constants, and $h_{\mathrm{ref},i}$ is the reference distance headway for vehicle i. Note that the speed controller is based on the same principle as the one used in the platoon leader model, but with the platoon leader's speed as the reference speed. The distance controller calculates the safe distance headway as follows:

$$h_{\mathrm{ref},i}(\ell) = S_0 + v_i(\ell) T_{\mathrm{head},i} + L_i \tag{2.7}$$

where S_0 is the minimum safe distance that is to be maintained at zero speed, $T_{\mathrm{head},i}$ is the time headway for vehicle i, and L_i is the length of vehicle i.

Platoon-based prediction model

On a more aggregate level, we can also consider a platoon of vehicles as a single entity without taking the detailed interactions among the individual vehicles within a platoon into account. So essentially we consider a platoon as one "big vehicle" with a length that is a function of the speed of the platoon (due to the dependence of the intervehicle spacing managed by the ACC on the speed (cf. (2.7))), and

[2]Here we assume again that T_{delay}, which typically has a value of 1–1.2 s, is an integer multiple of T_{sim}. So, $T_{\mathrm{delay}} = dT_{\mathrm{sim}}$ with d an integer.

of the number and lengths of the vehicles in the platoon. The dynamics equations for the speed and position of the platoon are the same as those of a platoon leader presented above. Consider platoon p and assume for the sake of simplicity that the vehicles in the platoon are numbered 1 (last vehicle), 2 (one but last vehicle), ..., n_p (platoon leader). The speed dependent length $L_{\text{plat},p}(\ell)$ of platoon p is then given by

$$L_{\text{plat},p} = (n_p - 1)(S_0 + S_1 v_{n_p}(\ell)) + \sum_{i=1}^{n_p} L_i \ , \tag{2.8}$$

where $S_0 + S_1 v_{n_p}(\ell)$ is the speed-dependent intervehicle spacing between the vehicles in the platoon, with S_0 the minimum safe distance that is to be maintained at zero speed, S_1 a model constant, v_{n_p} the speed of the platoon (leader), and L_i the length of vehicle i.

Capacity drop

In general, traffic congestion occurs when the available network resources are not sufficient to handle the traffic demand (recurrent congestion), or due to irregular occurrences, such as traffic incidents (non-recurrent congestion). In practice, traffic jams or congestion result in capacity drop [25]. This phenomenon causes the expected maximum outflow from the jammed traffic to be less than in the case of free-flow traffic. This is mainly caused by the delay in reaction time and increased intervehicle distance (time headway) when vehicles start to exit from a traffic jam. For human drivers the capacity drop is typically of the order of 2–7 %. With fully automated vehicles the capacity drop can be reduced to almost 0 %.

2.2.4 Roadside controllers

In this section we propose an MPC method for the roadside controllers to determine optimal speeds, lane allocations, and on-ramp release times for the platoons. For the sake of simplicity of the exposition we will mainly focus on intelligent speed adaptation (ISA), but the proposed approach also applies to other control measures.

MPC for ISA

We now explain how MPC can be applied for speed control in IVHS. MPC makes use of discrete-time models. Let T_c be the control sampling interval, i.e., the (constant) time interval between two updates of the control signal settings. At each time step k (corresponding to the time instant $t = kT_c$), the roadside controller first measures or determines the current state $x(k)$ of the system. Recall that the roadside control works with platoons as basic entities. So in our case the state of the system includes the positions and speeds of the platoon leaders and the lengths of the platoons. Next, the controller uses an optimization algorithm in combination with a model of the system to determine the control inputs $u(k), \ldots, u(k+N_p-1)$ that optimize a performance criterion $J(k)$ over a time interval $[kT_c, (k+N_p)T_c]$, where N_p is called the prediction horizon. In our case the control signal u will consist of the speed limits for the platoon leaders.

In the previous section we have already presented some models that are especially suited for use in MPC for IVHS. Note however that MPC is a modular approach so that in case a given prediction model does not perform well, it can easily be replaced by another prediction model.

Possible performance criteria $J(k)$ are the total time spent in a traffic network, the total throughput, the total fuel consumption, safety, or a combination of these. In the following we will in particular

consider the total time spent (TTS) by all the vehicles in the network:

$$J_{\mathrm{TTS}}(k) = \sum_{j=0}^{N_{\mathrm{p}}} n_{\mathrm{veh}}(k+j)T_{\mathrm{c}} \ , \tag{2.9}$$

where $n_{\mathrm{veh}}(k+j)$ is the number of vehicles that are present in the network at time $t = (k+j)T_{\mathrm{c}}$. Moreover, in order to prevent oscillations and frequent shifting in the control signals, one often adds a penalty on variations in the control signal u, which results in the total performance function

$$J_{\mathrm{tot}}(k) = J(k) + \alpha \sum_{j=0}^{N_{\mathrm{p}}} \|u(k+j) - u(k+j-1)\|_2 \ , \tag{2.10}$$

where $\alpha > 0$ is a weighting factor.

The MPC controller also explicitly takes into account operational constraints such as minimum separation between the platoons, minimum and maximum speeds, minimum headways, etc. To reduce the computational complexity of the problem, one often introduces a constraint of the form $u(k+j) = u(k+j-1)$ for $j = N_{\mathrm{c}}, \ldots, N_{\mathrm{p}} - 1$, where N_{c} ($< N_{\mathrm{p}}$) is called the control horizon.

In MPC the control actions are applied in a receding horizon fashion. This is done by applying only the first control sample $u(k)$ of the optimal control sequence to the system. Next, the prediction horizon is shifted one step forward, and the prediction and optimization procedure over the shifted horizon are repeated using new system measurements.

Optimization methods

Solving the MPC optimization problem (i.e., computing the optimal control actions) is the most demanding operation in the MPC approach. In our case the MPC approach gives rise to nonlinear nonconvex optimization problems that have to be solved on-line. So a proper choice of optimization techniques that suit the nature of the problem has to be made. In our case global or multi-start local optimization methods are required such as multi-start sequential quadratic programming [44], pattern search [4], genetic algorithms [15], or simulated annealing [19].

2.2.5 Area controllers

In this section we describe a control approach for the area controllers and in particular on how optimal routes can be determined for the platoons.

Approach

In principle, the optimal route choice control problem in IVHS consists in assigning an optimal route to each individual platoon in the network. However, this results in a huge nonlinear integer optimization problem with high computational complexity and requirements, making the problem intractable in practice. So, since considering each individual platoon is too computationally intensive, we will consider streams of platoons instead (characterized by (real-valued) demands and flows expressed in vehicles per hour). The routing problem will be recast as the problem of determining the flows on each link.

Once these flows are determined, they can be implemented by roadside controllers at the links and at the nodes. So the area controllers provide flow targets to the roadside controllers, which then have to control the platoons that are under their supervision in such a way that these targets are met as

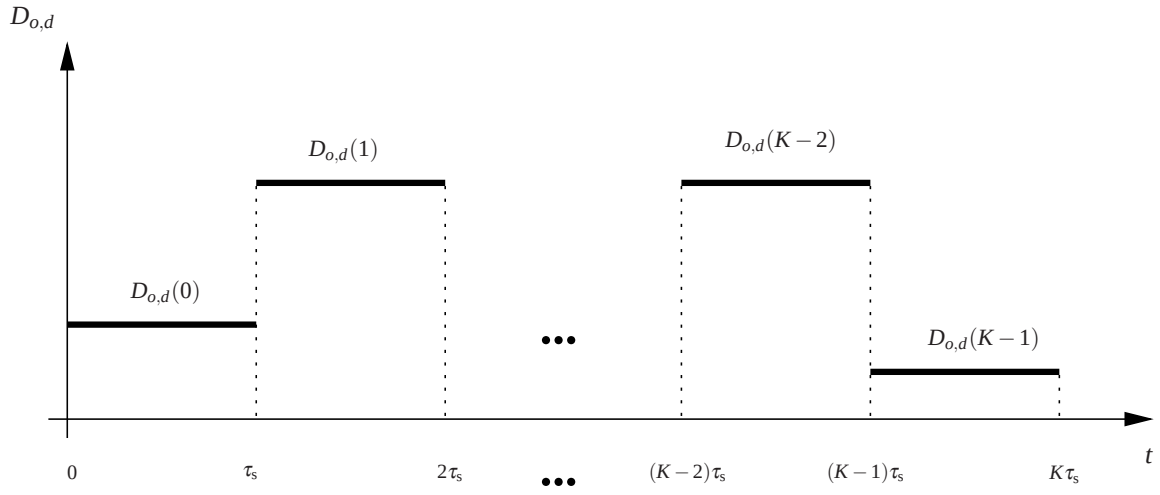


Figure 2.2: Piecewise constant time-varying demand profile $D_{o,d}$ for the dynamic case.

well as possible. This corresponds to slowing down or speeding up platoons in the links if necessary (in combination with lane allocation and on-ramp access timing), and to steering them in a certain direction depending on the splitting rates for the flows.

Set-up

We consider the following set-up. We have a transportation network with a set of origin nodes $\mathscr{O}$, a set of destination nodes $\mathscr{D}$, and a set of internal nodes $\mathscr{I}$. Define the set of all nodes as $\mathscr{V} = \mathscr{O} \cup \mathscr{I} \cup \mathscr{D}$. Nodes can be connected by one or more (unidirectional) links. The set of all links is denoted by L.

For each origin-destination pair $(o,d) \in \mathscr{O} \times \mathscr{D}$ we define the set $L_{o,d} \subseteq L$ of links that belong to some route going from o to d. For every link $l \in L$ we define the set $\mathscr{S}_{\mathrm{od},l}$ of origin-destination pairs $(o,d) \in \mathscr{O} \times \mathscr{D}$ such that l belongs to some route going from o to d.

For each pair $(o,d) \in \mathscr{O} \times \mathscr{D}$, there is a constant demand $D_{o,d}$ (in the static case) or a dynamic, piecewise constant demand pattern $D_{o,d}(\cdot)$ as shown in Figure 2.2 with $D_{o,d}(k)$ the demand of vehicles at origin o with destination d in the time interval $[k\tau_s, (k+1)\tau_s)$ for $k = 0, \ldots, K-1$ with K the simulation horizon and τ_s the simulation time step (we assume that beyond $T = K\tau_s$ the demand is 0).

For each link $l \in L$ in the network[3] there is a maximal capacity C_l. We assume that there is a fixed average speed v_l on each link l. Let τ_l denote the travel time on link l: $\tau_l = \frac{\ell_l}{v_l}$ where ℓ_l is the length of link l. We denote the set of incoming links for node $v \in \mathscr{V}$ by L_v^{in}, and the set of outgoing links by L_v^{out}. Note that for origins $o \in \mathscr{O}$ we have $L_o^{\mathrm{in}} = \emptyset$ and for destinations $d \in \mathscr{D}$ we have $L_d^{\mathrm{out}} = \emptyset$.

The aim is now to assign actual (real-valued) flows $x_{l,o,d}$ (in the static case) or $x_{l,o,d}(k)$ (in the dynamic case) for every pair $(o,d) \in \mathscr{O} \times \mathscr{D}$ and every $l \in L_{o,d}$, in such a way that the capacity of the links is not exceeded and such that the given performance criterion (e.g., total time spent) is minimized. In the dynamic case $x_{l,o,d}(k)$ denotes the flow of vehicles from origin o to destination d that enter link l in the time interval $[k\tau_s, (k+1)\tau_s)$.

For the optimal route choice problem we now consider four cases with a gradually increasing complexity:

- Static case with sufficient network capacity,

[3]This approach can easily be extended to the case where also the internal nodes $v \in \mathscr{I}$ have a finite capacity.

- Static case with queues at the boundaries of the network only,
- Dynamic case with queues at the boundaries of the network only,
- Dynamic case with queues inside the network.

Static case with sufficient network capacity

Here we assume that there is a constant demand for each origin-destination pair and that the total network capacity is such that the entire demand can be processed, so that there will be no queues at the boundaries or inside the network. Let us now describe the equations to model this situation.

For every origin node $o \in \mathscr{O}$ we have:

$$\sum_{l \in L_o^{\text{out}} \cap L_{o,d}} x_{l,o,d} = D_{o,d} \quad \text{for each } d \in \mathscr{D}. \tag{2.11}$$

For every internal node $v \in \mathscr{I}$ and for every pair $(o,d) \in \mathscr{O} \times \mathscr{D}$ we have

$$\sum_{l \in L_v^{\text{in}} \cap L_{o,d}} x_{l,o,d} = \sum_{l \in L_v^{\text{out}} \cap L_{o,d}} x_{l,o,d} \ . \tag{2.12}$$

We also have the following condition for every link l:

$$\sum_{(o,d) \in \mathscr{S}_{\text{od},l}} x_{l,o,d} \leqslant C_l \ . \tag{2.13}$$

Finally, the objective function is given as follows[4]:

$$J_{\text{links},k,N} = \sum_{(o,d) \in \mathscr{O} \times \mathscr{D}} \sum_{l \in L_{o,d}} x_{l,o,d} \tau_l T \ , \tag{2.14}$$

which is a measure for the total time the vehicles or platoons spend in the network. In order to minimize $J_{\text{links},k,N}$ we have to solve the following optimization problem:

$$\min J_{\text{links},k,N} \quad \text{s.t. (2.11)–(2.13)} \tag{2.15}$$

Clearly, this is a linear programming problem.

Static case with queues at the boundaries of the network only

In case the capacity of the network is less than the demand, then problem (2.15) will not be feasible. In order to be able to determine the optimal routing in this case, we have to take into account that queues might appear at the origin of the network.

Let us first write down the equations for the flows inside the network.

For every origin node $o \in \mathscr{O}$ we have:

$$\sum_{l \in L_o^{\text{out}} \cap L_{o,d}} x_{l,o,d} \leqslant D_{o,d} \quad \text{for each } d \in \mathscr{D}. \tag{2.16}$$

Equations (2.12) and (2.13) also hold in this case.

[4]Recall that $T = K\tau_{\text{s}}$ is the length of the simulation period.

Let us now describe the behavior of the queues. Since the actual flow out of origin node o for destination d is given by

$$F^{\text{out}}_{o,d} = \sum_{l \in L^{\text{out}}_o \cap L_{o,d}} x_{l,o,d} \ ,$$

the queue length at the origin o for vehicles or platoons going to destination d will increase linearly with a rate $D_{o,d} - F^{\text{out}}_{o,d}$ (note that by (2.16) this rate is always nonnegative). At the end of the simulation period (which has length T) the queue length will be $(D_{o,d} - F^{\text{out}}_{o,d})T$, and hence the average queue length is $\frac{1}{2}(D_{o,d} - F^{\text{out}}_{o,d})T$. So the total time spent in the origin queues is

$$\begin{aligned} J_{\text{queue},k,N} &= \sum_{(o,d) \in \mathscr{O} \times \mathscr{D}} \frac{1}{2}(D_{o,d} - F^{\text{out}}_{o,d})T^2 \\ &= \sum_{(o,d) \in \mathscr{O} \times \mathscr{D}} \frac{1}{2}\left(D_{o,d} - \sum_{l \in L^{\text{out}}_o} x_{l,o,d}\right)T^2 \ . \end{aligned}$$

In order to minimize the total time spent we have to solve the following optimization problem:

$$\min \ J_{\text{links},k,N} + J_{\text{queue},k,N} \quad \text{s.t. (2.12), (2.13), and (2.16).} \tag{2.17}$$

This is also a linear programming problem.

Dynamic case with queues at the boundaries of the network only

Now we consider a piecewise constant demand pattern for every origin-destination pair. Moreover, we assume that the travel time τ_l on link l is an integer multiple of τ_s, say

$$\tau_l = \kappa_l \tau_s \quad \text{with } \kappa_l \text{ an integer.} \tag{2.18}$$

Let $q_{o,d}(k)$ denote the partial queue length of vehicles at origin o going to destination d at time instant $t = k\tau_s$. In principle, the queue lengths should be integers as their unit is "number of vehicles", but we will approximate them using reals.

For the sake of simplicity we also assume that initially the network is empty (i.e., $q_{o,d}(k) = 0$ and $x_{l,o,d}(k) = 0$ for $k \leqslant 0$).

For every origin node $o \in \mathscr{O}$ we now have:

$$\sum_{l \in L^{\text{out}}_o \cap L_{o,d}} x_{l,o,d}(k) \leqslant D_{o,d}(k) + \frac{q_{o,d}(k)}{\tau_s} \quad \text{for each } d \in \mathscr{D}, \tag{2.19}$$

with by definition $D_{o,d}(k) = 0$ for $k \geqslant K$ and $q_{o,d}(k) = 0$ for $k \leqslant 0$. Note that the term $\frac{q_{o,d}(k)}{\tau_s}$ in (2.19) is due to the assumption that whenever possible and feasible the queue is emptied in the next sample period, with length τ_s.

Taking into account that every flow on link l has a delay of κ_l time steps before it reaches the end of the link, we have

$$\sum_{l \in L^{\text{in}}_v \cap L_{o,d}} x_{l,o,d}(k - \tau_l) = \sum_{l \in L^{\text{out}}_v \cap L_{o,d}} x_{l,o,d}(k) \tag{2.20}$$

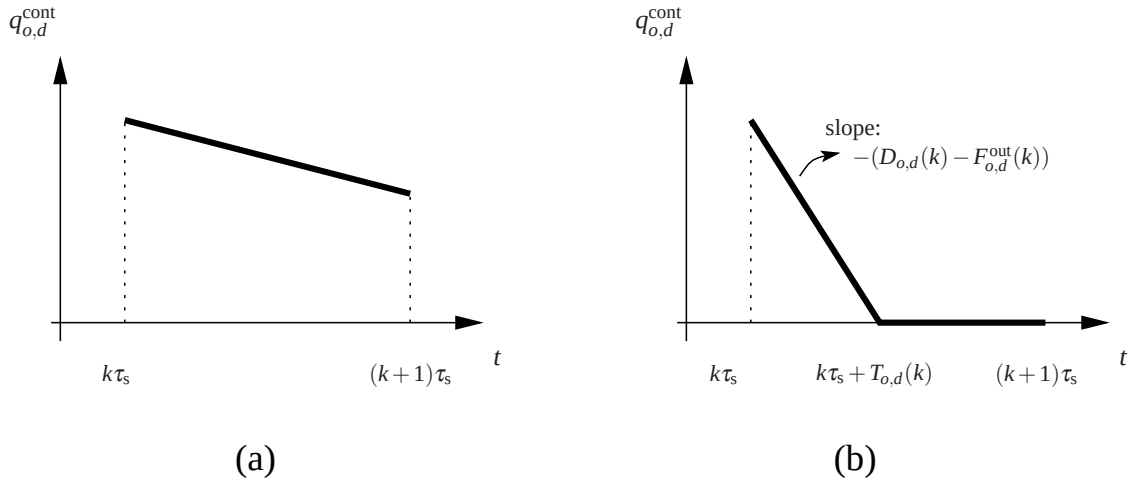


Figure 2.3: Two possible cases for the evolution of the (continuous-time) queue length $q_{o,d}^{\text{cont}}$ in the time interval $[k\tau_s,(k+1)\tau_s)$.

for every internal node $v \in \mathscr{I}$ and for every pair $(o,d) \in \mathscr{O} \times \mathscr{D}$, with $x_{l,o,d}(k) = 0$ for $k \leq 0$. We also have the following condition for every link l:

$$\sum_{(o,d)\in\mathscr{S}_{\text{od},l}} x_{l,o,d}(k) \leqslant C_l \ . \tag{2.21}$$

Let us now describe the behavior of the queues. Since the actual flow out of origin node o for destination d in the time interval $[k\tau_s,(k+1)\tau_s)$ is given by

$$F_{o,d}^{\text{out}}(k) = \sum_{l\in L_o^{\text{out}}\cap L_{o,d}} x_{l,o,d}(k) \ , \tag{2.22}$$

the queue length at the origin o for vehicles going to destination d will increase linearly with a rate$D_{o,d}(k) - F_{o,d}^{\text{out}}(k)$ in the time interval $[k\tau_s,(k+1)\tau_s)$. Hence,

$$q_{o,d}(k+1) = \max\left(0, q_{o,d}(k) + (D_{o,d}(k) - F_{o,d}^{\text{out}}(k))\tau_s\right) \tag{2.23}$$

In order to determine the time $J_{\text{queue},o,d}(k)$ spent in the queue at origin o in the time interval $[k\tau_s,(k+1)\tau_s)$ for traffic going to destination d, we have to distinguish between two cases depending on whether or not the continuous-time queue length $q_{o,d}^{\text{cont}}$ becomes equal to zero *inside*[5] the interval $[k\tau_s,(k+1)\tau_s]$ (see Cases (a) and (b) of Figure 2.3). For Case (b) we define

$$T_{o,d}(k) = \frac{q_{o,d}(k)}{F_{o,d}^{\text{out}}(k) - D_{o,d}(k)} \tag{2.24}$$

as the time offset after $k\tau_s$ at which the queue length becomes zero. Then we have

$$J_{\text{queue},o,d}(k) = \begin{cases} \dfrac{1}{2}(q_{o,d}(k) + q_{o,d}(k+1))\tau_s & \text{for Case (a),} \\ \dfrac{1}{2} q_{o,d}(k) T_{o,d}(k) & \text{for Case (b).} \end{cases} \tag{2.25}$$

[5] So we are only Case (b) if $q_{o,d}^{\text{cont}}$ becomes equal to zero for some time t with $k\tau_s < t < (k+1)\tau_s$, i.e., if $q_{o,d}(k) > 0$ and $q_{o,d}(k) + (D_{o,d}(k) - F_{o,d}^{\text{out}}(k))\tau_s < 0$. All other situations belong to Case (a).

Due to the denominator term in (2.24) $J_{\text{queue},o,d}(k)$ is in general a nonlinear function. Now assume that we simulate the network until time step $K_{\text{sim}} \geqslant K$ (e.g., until all queues and all flows have become[6] equal to zero). Then we have

$$J_{\text{queue},k,N} = \sum_{k=0}^{K_{\text{sim}}-1} \sum_{(o,d)\in\mathscr{O}\times\mathscr{D}} J_{\text{queue},o,d}(k) \ .$$

The time spent in the links is now given by

$$J_{\text{links},k,N} = \sum_{k=0}^{K_{\text{sim}}-1} \sum_{(o,d)\in\mathscr{O}\times\mathscr{D}} \sum_{l\in L_{o,d}} x_{l,o,d}(k)\kappa_l\tau_{\text{s}}^2 \ . \tag{2.26}$$

In order to minimize the total time spent we have to solve the following optimization problem:

$$\min \ \left(J_{\text{links},k,N} + J_{\text{queue},k,N}\right) \qquad \text{s.t. (2.19)–(2.23).} \tag{2.27}$$

Due to the presence of constraint (2.23) and the nonlinear expression for $J_{\text{queue},o,d}(k)$ in Case (b) this is a nonlinear, non-convex, and non-smooth optimization problem. In general, these problems are difficult to solve and require multi-start local optimization methods (such as Sequential Quadratic Programming (SQP)) or global optimization methods (such as genetic algorithms, simulated annealing, or pattern search) [44]. However, in the following we will propose an alternative approximate solution approach based on mixed integer linear programming.

Dynamic case with queues inside the network

Now we consider the case with queues inside the network. If there are queues formed, we assume that they are formed at the end of the links and that the queues are vertical. In fact, for the sake of simplicity and in order to obtain linear equations, we assign the queues to the nodes instead of the links.

This case is similar to the previous case, the difference being that (2.20) is now replaced by (cf. also (2.19)):

$$\sum_{l\in L_v^{\text{out}}\cap L_{o,d}} x_{l,o,d}(k) \leqslant \left(\sum_{l\in L_v^{\text{in}}\cap L_{o,d}} x_{l,o,d}(k-\tau_l)\right) + \frac{q_{v,o,d}(k)}{\tau_{\text{s}}}, \tag{2.28}$$

where $q_{v,o,d}(k)$ is the partial queue length at node v for vehicles or platoons going from origin o to destination d at the time instant $t = k\tau_{\text{s}}$. Moreover,

$$q_{v,o,d}(k+1) = \max\left(0, q_{v,o,d}(k) + (F^{\text{in}}_{v,o,d}(k) - F^{\text{out}}_{v,o,d}(k)\right)\tau_{\text{s}}$$

with the flow into and out of the queue being given by

$$F^{\text{in}}_{v,o,d}(k) = \sum_{l\in L_v^{\text{in}}\cap L_{o,d}} x_{l,o,d}(k-\tau_l) \tag{2.29}$$

$$F^{\text{out}}_{v,o,d}(k) = \sum_{l\in L_v^{\text{out}}\cap L_{o,d}} x_{l,o,d}(k) \ . \tag{2.30}$$

[6] If this is not the case we have to add an end-point penalty on the queue lengths and flows at time step K_{sim}.

Similar to the way $J_{\text{queue},o,d}(k)$ has been defined in (2.25) we also define the time $J_{\text{queue},v,o,d}(k)$ spent in the queue at node v in the time interval $[k\tau_s,(k+1)\tau_s)$ for traffic going from origin o to destination d, and we extend the definition of $J_{\text{queue},k,N}$ into

$$J_{\text{queue},k,N} = \sum_{k=0}^{K_{\text{sim}}-1} \sum_{(o,d)\in\mathscr{O}\times\mathscr{D}} \Big(J_{\text{queue},o,d}(k) + \sum_{v\in\mathscr{I}} J_{\text{queue},v,o,d}(k) \Big) \tau_s \ .$$

In order to minimize the total time spent we have to solve the following optimization problem:

$$\min \ \big(J_{\text{links},k,N} + J_{\text{queue},k,N} \big) \quad \text{s.t. (2.19), (2.21)–(2.23), and (2.28)–(2.30),}$$

with $J_{\text{links},k,N}$ still defined by (2.26). This also results in a nonlinear, non-convex, and non-smooth optimization problem. However, in the next section we will show that this problem can also be approximated using mixed integer linear programming.

Approximation based on mixed integer linear programming

Recall that the dynamic optimal route guidance problems previously stated are nonlinear, non-convex, and non-smooth. Now we will show that by introducing an approximation these problems can be transformed into mixed integer linear programming (MILP) problems, for which efficient solvers have been developed [22].

First we consider the case with queues at the origins only, i.e., we consider the optimization problem (2.27). Apart from (2.23) this problem is a linear optimization problem.

Now we explain how we can transform (2.23) into a system of linear equations by introducing some auxiliary boolean variables δ. To this aim we use the following properties [7], where δ represents a binary-valued scalar variable, y a real-valued scalar variable, and f a function defined on a bounded set X with upper and lower bounds M and m for the function values:

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

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

where ε is a small positive number[7] (typically the machine precision),

P2: $y = \delta f$ is equivalent to

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

Depending on the order in which these properties are applied and in which additional auxiliary variables are introduced, we may end up with more or less binary and real variables in the final MILP problem. The number of binary variables — and to a lesser extent the number of real variables

[7] We need this construction to transform a constraint of the form $y > 0$ into $y \geqslant \varepsilon$, as in (mixed integer) linear programming problems only non-strict inequalities are allowed.

— should be kept as small as possible since this number has a direct impact of the computational complexity of the final MILP problem.

To reduce the number of real variables in the final MILP problem, we first eliminate $F^{\text{out}}_{o,d}(k)$ and we write (2.23) as

$$q_{o,d}(k+1) = \max\Big(0, q_{o,d}(k) + \big(D_{o,d}(k) - \sum_{l \in L^{\text{out}}_o \cap L_{o,d}} x_{l,o,d}(k)\big)\tau_{\text{s}}\Big). \tag{2.31}$$

Note that this is a nonlinear equation and thus it does not fit the MILP framework. Let $D^{\max}_{o,d} = \max_k D_{o,d}(k)$ be the maximal demand for origin-destination pair (o,d), let $F^{\max}_{o,d} = \sum_{l \in L^{\text{out}}_o \cap L_{o,d}} C_l$ be the maximal possible flow out of origin node o towards destination d, and let $q^{\max}_{o,d} = D^{\max}_{o,d}\tau_{\text{s}} K_{\text{sim}}$ be the maximal origin queue length at origin o for traffic going to destination d. If we define $m^{\text{low}}_{o,d} = -F^{\max}_{o,d}\tau_{\text{s}}$ and $m^{\text{upp}}_{o,d} = q^{\max}_{o,d} + D^{\max}_{o,d}\tau_{\text{s}}$, then we always have

$$m^{\text{low}}_{o,d} \leqslant q_{o,d}(k) + \Big(D_{o,d}(k) - \sum_{l \in L^{\text{out}}_o \cap L_{o,d}} x_{l,o,d}(k)\Big)\tau_{\text{s}} \leqslant m^{\text{upp}}_{o,d} \ .$$

Next, we introduce binary variables $\delta_{o,d}(k)$ such that

$$\delta_{o,d}(k) = 1 \text{ if and only if } q_{o,d}(k) + \big(D_{o,d}(k) - \sum_{l \in L^{\text{out}}_o \cap L_{o,d}} x_{l,o,d}(k)\big)\tau_{\text{s}} \geqslant 0 \ .$$

Using Property **P1** with the bounds $m^{\text{low}}_{o,d}$ and $m^{\text{upp}}_{o,d}$ this condition can be transformed into a system of linear inequalities. Now we have (cf. (2.31))

$$q_{o,d}(k+1) = \delta_{o,d}(k)\Big(q_{o,d}(k) + \big(D_{o,d}(k) - \sum_{l \in L^{\text{out}}_o \cap L_{o,d}} x_{l,o,d}(k)\big)\tau_{\text{s}}\Big).$$

This expression 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.
So by introducing some auxiliary variables $\delta_{o,d}(k)$ we can transform the original nonlinear equation (2.23) into a system of additional linear equations and inequalities.

Recall that $J_{\text{queue},o,d}(k)$ is in general a nonlinear function due to the occurrence of Case (b) of Figure 2.3. However, if we also use the expression of Case (a) for Case (b), then we can approximate $J_{\text{queue},o,d}(k)$ as[8]

$$J_{\text{queue},o,d}(k) = \frac{1}{2}(q_{o,d}(k) + q_{o,d}(k+1))\tau_{\text{s}} \ ,$$

which is a linear expression. This implies that the overall objective function $J_{\text{links},k,N} + J_{\text{queue},k,N}$ is now linear. So the problem (2.27) can be approximated by an MILP problem.

Several efficient branch-and-bound MILP solvers [22] are available for MILP problems. Moreover, there exist several commercial and free solvers for MILP problems such as, e.g., CPLEX,

[8] This is exact for Case (a) and an approximation for Case (b). However, especially if τ_{s} is small enough, the error we then make is negligible.

Xpress-MP, GLPK, or lp_solve (see [3, 39] 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, non-convex, non-smooth optimization problem (2.27). Moreover, if the computation time is limited (as is often the case in on-line real-time traffic 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 — even although it solves an approximated problem — might even perform better than the solution returned by the prematurely terminated global and multi-start local optimization method. In general, we can say that the MILP solution often provides a good trade-off between optimality and computational efficiency.

Using a similar reasoning as above we can also transform the routing problem with queues inside the network into an MILP problem. Note however that in this case the number of binary variables may become quite large.

2.2.6 Conclusions

In this section we have considered a hierarchical control framework for intelligent vehicle highway systems (IVHS), with a special focus on the models used at the various control levels. We have presented how model predictive control (MPC) can be used to determine optimal speeds for platoons by the roadside controllers. We have also considered the optimal route guidance problem for IVHS. In particular, we have proposed an optimal route guidance approach for platoons by an area controller based on a simplified flow model. Since the resulting optimization problem could still be too involved for on-line, real-time implementation in the case of dynamic demands, we have explored an approximation resulting in a mixed integer linear programming problem, for which efficient solvers exist.

2.3 Hierarchical model predictive control for baggage handling systems

In this section we propose a hierarchical control framework for state-of-the-art baggage handling systems where the luggage is transported by fast destination coded vehicles (DCVs). In this control framework switch controllers provide position instructions for each switch in the network. A collection of switch controllers is then supervised by a network controller that mainly takes care of the route choice instructions for DCVs. In general, the route choice control problem 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 for which 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 compare the hierarchical control with centralized switch control. The results indicate that the proposed hierarchical control offers a balanced trade-off between optimality and computational efficiency.

In the proposed approach two different types of models are used, depending on the time scale involved. For simulations and for the lower control levels we use a fast event-based model, while for the higher level controller we use a model based on queues and flows that can ultimately be recast into a mixed-integer linear programming description.

This section is organized as follows. First we introduce the background of the routing problem in Section 2.3.1. Next, we discuss DCV-based baggage handling systems in Section 2.3.2. Section 2.3.3 presents the detailed event-based model for the DCV-based baggage handling system. The actual control approach is presented in Section 2.3.4, followed by a description of the higher-level route choice controller in Section 2.3.5, including the flow-and-queue-based model used by the higher-level controller. Section 2.3.6 then discusses the lower-level switch controller, which is based on the event-driven model of Section 2.3.3. A case study is considered in Section 2.3.7 and conclusions are presented in Section 2.3.8.

2.3.1 Introduction

State-of-the-art baggage handling systems in airports transport luggage at high speeds using destination coded vehicles (DCVs). These vehicles transport the bags in an automated way on a "mini" railway network. 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 with a given amount of 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.

Currently, the DCVs are routed through the system using routing schemes based on preferred routes. These routing schemes can be adapted to respond on the occurrence of predefined events. However, as argued in [17], the patterns of loads on the system are highly variable, depending on e.g. season, time of the day, type of aircraft at each gate, number of passengers for each flight. Therefore, in the research we conduct we do not consider predefined preferred routes. Instead we develop advanced control methods to determine the optimal routing in case of dynamic demand.

The route assignment problem has been addressed in e.g. [23, 31]. But, in our case we do not deal with a shortest-path or shortest-time problem, since we need the bags at their corresponding end point within a given time window. In [20] is presented an analogy between the DCV routing problems and data transmissions via internet. Also, [26] presents a multi-agent approach for routing DCVs. However, this multi-agent system 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 route choice control 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 [37]. However, as shown in [49], for a fast event-based model of this system, 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 [50], we have also implemented centralized and decentralized model predictive control (MPC), and also a decentralized heuristic approach. As the results confirmed, centralized MPC requires high computation time to determine a solution. The use of decentralized control lowers the computation time, but at the cost of suboptimality.

In this chapter, we propose a hierarchical control framework where the higher level controllers use MPC. The large computation time obtained in previous work comes from solving the nonlinear, mixed integer optimization problems that have multiple local minima, and therefore, are difficult to solve. So, in this paper we investigate whether the computational effort required to compute the optimal route choice can be lowered by using mixed integer linear programming (MILP).

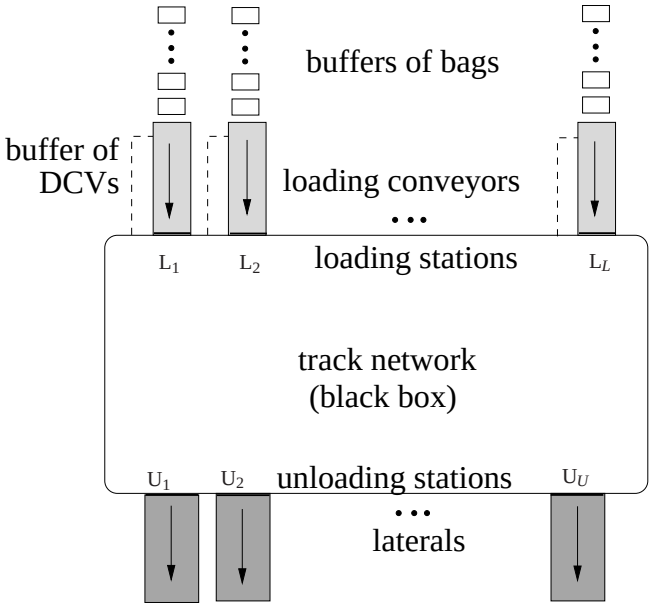


Figure 2.4: Baggage handling system using DCVs.

2.3.2 DCV-based baggage handling systems

The track network of a DCV-based baggage handling system 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. 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.

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 (from a given loading station to the corresponding unloading station) has to be determined subject to the operational and safety constraints detailed in [49] such that all the bags to be handled arrive at their end points within the corresponding time window.

2.3.3 Event-driven model

Operation of the system

The baggage handling process begins after the bags have passed the check-in. Then they enter the conveyor network, being routed to loading conveyors towards loading stations. Depending on the availability of empty DCVs, at each loading station a queue of bags may be formed. In the following we focus on the transporting-using-DCVs part of the process. Therefore, one may consider that each loading station has a buffer of bags waiting to be handled as sketched in Figure 2.4. The baggage handling system operates as follows: given a finite sequence of bags (identified by their unique code) and a buffer of empty DCVs for each loading station, together with the network of tracks, the optimal route and the optimal velocity profile of each DCV have to be computed subject to operational and safety constraints such that the system optimum is assured.

We consider a baggage handling system with L loading stations and U unloading stations as depicted in Figure 2.4. Accordingly, we have L FIFO (First In First Out) buffers of bags waiting to enter the system.

Modeling assumptions

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

1. a sufficient number of DCVs are present in the system.
2. the capacity of the network is large enough so that no overflow will occur.
3. each loading station has a finite buffer of bags waiting to be handled.
4. all buffers have the same maximum capacity $b_{\max}$.
5. assume there are X bags with random destinations to be handled. They are numbered $1, 2, \cdots, X$. When using a baggage handling system with L loading stations, we split this stream $\mathbf{b} = [1\ 2\ \cdots\ X]^{\mathrm{T}}$ with $X \leq Lb_{\max}$, in L new streams $\mathbf{b_1} = [1\ 2\ \cdots\ l]^{\mathrm{T}}$, $\mathbf{b_2} = [l+1\ l+2\ \cdots\ 2l]^{\mathrm{T}}$, $\cdots$, $\mathbf{b_L} = [(L-1)l+1\ (L-1)l+2\ \cdots\ X]^{\mathrm{T}}$ with $l = \left\lfloor \frac{X}{L} \right\rfloor$, where $\lfloor x \rfloor$ denotes the largest integer less than or equal to x.
6. the "mini" railway network has single-direction tracks.
7. a route switch at a junction can be performed in a negligible time span.
8. the speed of a DCV is piecewise constant.
9. the laterals have infinite capacity.
10. the destinations to which the bags have to be transported are allocated to the laterals when the process starts.

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

Model

There are four types of events that can occur:

- loading a new bag into the system.
- unloading a bag that meets the corresponding lateral.
- updating the route switches at the junction that the DCV has to pass.
- updating the speed of a DCV.

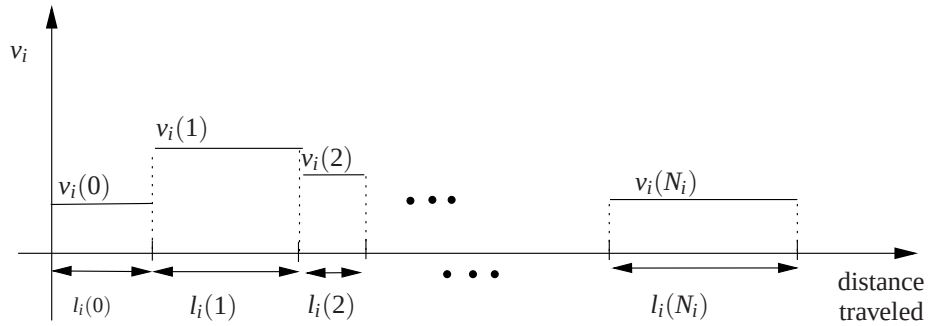


Figure 2.5: Speed evolution of the DCV$_i$.

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

Define N_i as the number of junctions that a DCV has to pass in order to reach its destination. A track segment is the portion of the track on which a DCV is running either between a loading station and a junction, or between two junctions, or between a junction and an unloading station. Let DCV$_i$ be the DCV that transports the ith bag that entered the system. The following situation has been assumed: given the velocity sequence of the DCV$_i$ say $\mathbf{v_i} = [v_i(0)\ v_i(1)\ \cdots\ v_i(N_i)]^{\mathrm{T}}$ and the sequence of segment lengths $\mathbf{l_i} = [l_i(0)\ l_i(1)\ \cdots\ l_i(N_i)]^{\mathrm{T}}$, on each segment j of length $l_i(j)$ the velocity of DCV$_i$ equals $v_i(j)$ as illustrated in Figure 2.5. The velocity of the DCV$_i$ that passed segment j, $j = 0, 1, \cdots, N_i - 1$ is updated at time instant $t_{j+1} = t_j + \frac{l_i(j)}{v_i(j)}$ with t_0 the initial time.

The model of the baggage handling system is given by the algorithm below, where the loading stations are denoted by L_1, L_2, $\cdots$, L_L, and the unloading stations are denoted by U_1, U_2, $\cdots$, U_U. We also define S as the number of junctions of the track network and $X_{\text{current}}(t)$ as the number of bags that entered the baggage handling system up to the current time instant t.

Algorithm 1. Baggage handling

$t \leftarrow t_0$
while there are bags to be handled **do**
  for $\ell = 1$ to L **do**
    $t_{\text{load}}(\ell) \leftarrow$ time that will pass until the next loading event from L_ℓ's point of view
  end for
  for $\ell = 1$ to U **do**
    $t_{\text{unload}}(\ell) \leftarrow$ time that will pass until the next unloading event from U_ℓ's point of view
  end for
  for $s = 1$ to S **do**
    $t_{\text{switch}}(s) \leftarrow$ time that will pass until the next route switch event from the junction s's point of view
  end for
  for $i = 1$ to $X_{\text{current}}(t)$ **do**
    if bag i is not at a lateral **then**
      $t_{\text{speed\_update}}(i) \leftarrow$ time that will pass until the

next speed-update event from the point
of view of the DCV$_i$

15: **end if**
16: **end for**
17: $t_{\min} \leftarrow \min(\min_{\ell=1,\cdots,L} t_{\text{load}}(\ell), \min_{\ell=1,\cdots,U} t_{\text{unload}}(\ell),$
$\min_{s=1,\cdots,S} t_{\text{switch}}(s), \min_{i=1,\cdots,X_{\text{current}}(t)} t_{\text{speed\_update}}(i))$
18: $t \leftarrow t + t_{\min}$
19: update the state of the system
20: **if** $t_{\min} = \min_{i=1,\cdots,X_{\text{current}}(t)} t_{\text{speed\_update}}(i)$ **then**
21: update the speed of the DCV$_i$
22: **end if**
23: **end while**

If multiple events occur at the same time, then we take all these events into account when updating the state of the system at step 19.

Operational constraints

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

- the velocity of each DCV is bounded between 0 and $v_{\max}$.
- a bag can be loaded onto a DCV only if there is an empty DCV under the loading station.
- a DCV can transport only one bag.
- collisions between DCVs have to be avoided on each track segment and at each intersection.

2.3.4 Hierarchical control approach

In order to efficiently compute the route choice of each DCV we propose a hierarchical control framework that consists of a multi-level control structure as shown in Figure 2.6 with the following layers:

- The *network controller* provides the route choice for DCVs by determining reference flow trajectories over time for each link in the network. These flow trajectories are computed so that the performance of the 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 trajectory and the future flow trajectory is minimal.
- The *DCV controller* present in each vehicle detects the speed and position of the vehicle in front of it 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.

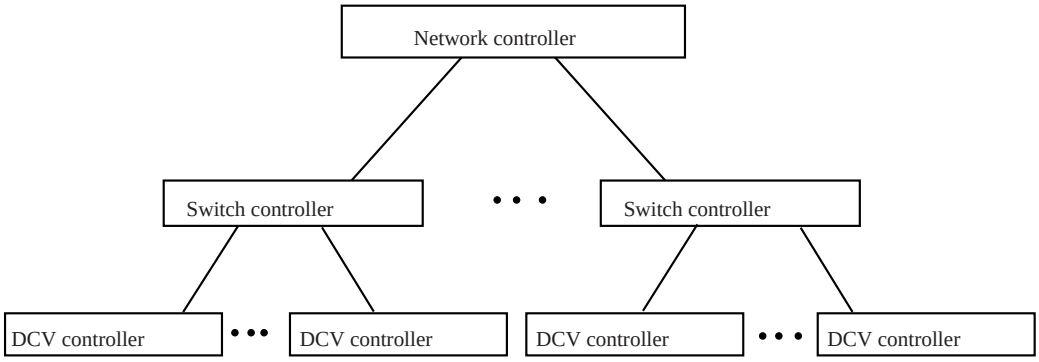


Figure 2.6: Hierarchical control for DCV-based baggage handling systems.

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.

2.3.5 Route choice control

In this section we focus on the network controller. The switch controller will be discussed in Section 2.3.6.

Approach

The predictive switch control problem results in a huge nonlinear integer optimization problem with high computational complexity and requirements, making the problem in fact intractable in practice [50]. 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 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 under its supervision, 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.

Set-up

We consider the following set-up. We have a transportation network with a set of origin nodes $\mathscr{O}$, a set of destination nodes $\mathscr{D}$, and a set of internal nodes $\mathscr{I}$. 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 L denote the set of all links.

Let the time instant t_k be defined as $t_k = k\tau^{\mathrm{nc}}$ with τ^{nc} the sampling time for the network controller. Then, for each pair $(o,d) \in \mathscr{O} \times \mathscr{D}$, there is a dynamic, piecewise constant demand pattern $D_{o,d}(\cdot)$ 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 $k = 0,\ldots,K-1$ with K the demand horizon (we assume that beyond t_K the demand is 0). Let L_d be the set of links that belong to some route going to d, $L_d \subseteq L$. We also denote the set of incoming links for node $v \in \mathscr{V}$ by L_v^{in}, and the set of outgoing links by L_v^{out}. Note that for origins $o \in \mathscr{O}$ we have $L_o^{\mathrm{in}} = \emptyset$ and for destinations $d \in \mathscr{D}$ we have $L_d^{\mathrm{out}} = \emptyset$. Also, without loss of generality, we assume each origin node to have only one outgoing link ($|L_o^{\mathrm{out}}| = 1$) and the destination nodes have only one incoming

link ($|L_d^{\text{in}}| = 1$) where $|\Lambda|$ represents the cardinality of the set Λ. For each destination $d \in \mathscr{D}$ and for each link $l \in L_d$ in the network we will define a real-valued flow $u_{l,d}(k)$. The flow $u_{l,d}(k)$ 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 step k, flows $u_{l,d}(k)$ for every destination $d \in \mathscr{D}$ and for every link $l \in L_d$ 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 $[t_k, t_{k+N}]$. 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.

Route choice model

In this section we determine the model of the DCV flows through the network. Let τ_l denote the free-flow travel time on link l. The free-flow travel time represents the time period that a DCV requires to travel on a track segment in case of no congestion, using, hence, maximum speed. We assume the travel time τ_l to be an integer multiple of τ^{nc}.

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 of the partial queue of DCVs at origin o going to destination d at time instant t_k. 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:

$$u_{l,d}(k) \leqslant D_{o,d}(k) + \frac{q_{o,d}(k)}{\tau^{\text{nc}}} \text{ for } l \in L_o^{\text{out}} \cap L_d \tag{2.32}$$

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

$$q_{o,d}(k+1) = \max(0, q_{o,d}(k) + (D_{o,d}(k) - \sum_{l \in L_o^{\text{out}} \cap L_d} u_{l,d}(k))\tau^{\text{nc}}) \tag{2.33}$$

But queues can form also inside the network. We assume that the DCVs run with maximum speed along the track segment and, if necessary, they wait before crossing the junction in a vertical queue. Let $q_{v,d}(k)$ denote the length of the vertical queue at junction $v \in \mathscr{I}$, for destination $d \in \mathscr{D}$, at time instant t_k. Note that, we do not consider outflow restrictions on queues to destination d for a junction v connected via a link to destination d ($q_{v,d}(k) = 0$ for all k).

Taking into account that every flow on link l has a delay of $\frac{\tau_l}{\tau^{\text{nc}}}$ 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:

$$F_{v,d}^{\text{out}}(k) \leqslant F_{v,d}^{\text{in}}(k) + \frac{q_{v,d}(k)}{\tau^{\text{nc}}} \tag{2.34}$$

where $F_{v,d}^{\text{in}}(k)$ is the flow into the queue at junction v, $F_{v,d}^{\text{in}}(k) = \sum_{l \in L_v^{\text{in}} \cap L_d} u_{l,d}(k - \frac{\tau_l}{\tau^{\text{nc}}})$ and where $F_{v,d}^{\text{out}}(k)$ is the flow out of the queue at junction v, $F_{v,d}^{\text{out}}(k) = \sum_{l \in L_v^{\text{out}} \cap L_d} u_{l,d}(k)$.

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:

$$q_{v,d}(k+1) = \max(0, q_{v,d}(k) + (F_{v,d}^{\text{in}}(k) - F_{v,d}^{\text{out}}(k))\tau^{\text{nc}}) \tag{2.35}$$

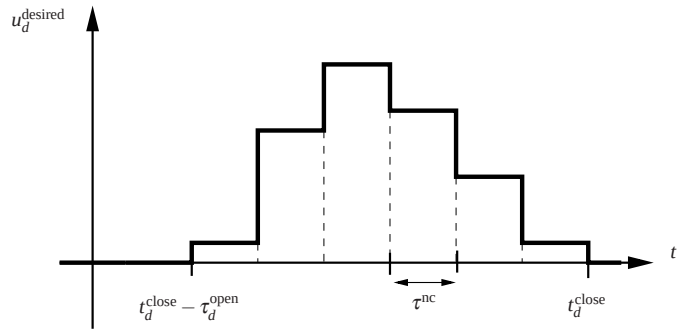


Figure 2.7: Desired arrival profile at destination d.

We also have the following condition for every link l:

$$\sum_{d \in \mathscr{D}} u_{l,d}(k) \leqslant U_l^{\max} \quad . \tag{2.36}$$

where $U_l^{\max}$ is the maximum flow on link l.

Next we define the performance index to be used for computing the optimal routing at step k for a prediction period $[t_k, t_{k+N})$.

The objective is to have each bag arriving at its end point within a given time interval $[t_d^{\text{close}} - \tau_d^{\text{open}}, t_d^{\text{close}}]$ where t_d^{close} is the time instant when the end point d closes and the last bags are loaded onto the plane, 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}. Without loss of generality, in this paper we consider that each destination has only one flight assigned to it. However, this can be easily extended to the general case, but where a presorting will be performed.

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 2.7, where the area under u_d^{desired} equals the total number of bags to be sent to destination d out of the total demand. Note that outside the time window $[t_d^{\text{close}} - \tau_d^{\text{open}}, t_d^{\text{close}})$ no bags should enter the incoming link of destination d outside the given time window. Consequently, $u_d^{\text{desired}}(k) = 0$ for all $k < k_d^{\text{open}}$ and all $k > k_d^{\text{close}}$ with $k_d^{\text{open}} = \frac{t_d^{\text{close}} - \tau^{\text{open}}}{\tau_{\text{s}}}$ and $k_d^{\text{close}} = \frac{t_d^{\text{close}}}{\tau_{\text{s}}}$.

Hence, one can define the following penalty for flow profiles $J^{\text{pen}}(k) = \sum_{d \in \mathscr{D}} \lambda_d |u_d^{\text{desired}}(k) - u_{l,d}(k + \frac{\tau_d^{\text{dest}}}{\tau_{\text{s}}})|$ where τ_d^{dest} is the free-flow travel time of link $l \in L_d^{\text{in}}$ and $\lambda_d > 0$ is a penalty that expresses the importance of the flight.

Note that using as MPC performance criterion $\sum_{i=k}^{k+N-1} J^{\text{pen}}(i)$ for each time step k, 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 $\tau_{l,d}^{\text{link}}$ be the typical time required for a DCV that just entered link l to reach destination d. Then one can define the following penalty: $J_{l,d}^{\text{flow}}(k) = u_{l,d}(k)$ if $k_d^{\text{open}} - \frac{\tau_{l,d}^{\text{link}}}{\tau_{\text{s}}} \leq k \leq k_d^{\text{close}} - \frac{\tau_{l,d}^{\text{link}}}{\tau_{\text{s}}}$ and $J_{l,d}^{\text{flow}}(k) = 0$ otherwise. 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. Then we define the new penalty: $J_{v,d}^{\text{overdue}}(k) = d_{v,d}^{\min} q_{v,d}(k)$ if $k \geq k_d^{\text{close}} - \frac{\tau_{v,d}^{\text{junc}}}{\tau_s}$ and $J_{v,d}^{\text{overdue}}(k) = 0$ otherwise, where $d_{v,d}^{\min}$ represents the length of the shortest route from junction v to destination d.

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

$$J_{k,N} = \sum_{i=k}^{k+N-1} \Bigg(J^{\text{pen}}(i) - \alpha \sum_{d \in \mathscr{D}} \sum_{l \in (L \setminus L^{\text{dest}}) \cap L_d} J_{l,d}^{\text{flow}}(i) + \beta \sum_{d \in \mathscr{D}} \sum_{v \in \mathscr{I}} J_{v,d}^{\text{overdue}}(i) \Bigg)$$

with $\alpha \ll 1$ and $\beta \ll 1$ nonnegative weighting parameters.

Then the nonlinear MPC optimization problem is defined as follows:

$$\min_{\mathbf{u}(k)} J_{k,N} \quad \text{s.t. (3)–(5).} \tag{2.37}$$

where $\mathbf{u}(k)$ is the control sequence consisting of all the flows $u_{l,d}(k) \ldots u_{l,d}(k+N-1)$ with $d \in \mathscr{D}$ and $l \in L_d$.

Equivalent MILP model

In this section we transform the dynamic optimal route choice problem (2.37) into an MILP problem. Recalling the development described in Section 2.2.5, as an example we will show how equation (2.33) of the nonlinear route choice model can be transformed into a set of linear equations and inequalities by introducing some auxiliary variables. For the other equations of the route choice model we apply a similar procedure.

Equation (2.33) is nonlinear and thus it does not fit the MILP framework. Therefore, we will first introduce the binary variables $\delta_{o,d}(k)$ such that

$$\begin{aligned} &\delta_{o,d}(k) = 1 \text{ if and only if} \\ &\quad q_{o,d}(k) + \big(D_{o,d}(k) - \sum_{l \in L_o^{\text{out}} \cap L_d} u_{l,d}(k)\big)\tau^{\text{nc}} \leqslant 0 \end{aligned} \tag{2.38}$$

and rewrite (2.33) as follows:

$$\begin{aligned} q_{o,d}(k+1) = &\big(1 - \delta_{o,d}(k)\big) \cdot \Big(q_{o,d}(k) + \\ &\big(D_{o,d}(k) - \sum_{l \in L_o^{\text{out}} \cap L_d} u_{l,d}(k)\big)\tau^{\text{nc}} \Big). \end{aligned} \tag{2.39}$$

Condition (2.38) is equivalent to:

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

where $f(k) = q_{o,d}(k) + \big(D_{o,d}(k) - u_{l,d}(k)\big)\tau^{\text{nc}}$ with $l \in L_o^{\text{out}} \cap L_d$, $U_l^{\max}$ is the maximal possible flow out of origin node o towards destination d, $q_{o,d}^{\max}$ is the maximal queue length at origin o for traffic

going to destination d, and where $D_{o,d}^{\max} = \max_k D_{o,d}(k)$ is the maximal demand for origin-destination pair (o,d).

Equation (2.39) is still nonlinear since it contains a multiplication of a binary variable $\delta_{o,d}(k)$ with a real-valued (linear) function. However, as already described in Section 2.2.5, it can be transformed into a system of linear inequalities.

Next we transform the nonlinear terms of (2.37) into sets of equality and inequality constraints. For example the problem

$$\min_{\mathbf{u}(k)} \sum_{d\in\mathscr{D}} \lambda_d \sum_{i=k}^{k+N-1} |u_d^{\text{desired}}(k) - \sum_{l\in L_d^{\text{in}}} u_{l,d}(k)|$$

can be written as:

$$\min \lambda_d \sum_{i=k}^{k+N-1} \sum_{d\in\mathscr{D}} u_d^{\text{diff}}(i)$$

s.t.

$$u_d^{\text{diff}}(i) \geqslant u_d^{\text{desired}}(i) - \sum_{l\in L_d^{\text{in}}} u_{l,d}(i+\tau_d^{\text{dest}}) \text{ for } i = 1,\ldots,N$$

$$u_d^{\text{diff}}(i) \geqslant -u_d^{\text{desired}}(i) + \sum_{l\in L_d^{\text{in}}} u_{l,d}(i+\tau_d^{\text{dest}}) \text{ for } i = 1,\ldots,N.$$

which is a linear programming problem.

So, the overall objective function $J_{k,N}$ can be written as a linear one. Hence, the problem (2.37) can be written as an MILP problem.

2.3.6 Switch control

In this section we focus on the switch controller for the proposed hierarchy.

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 $u_{l,d}^{\text{opt}}(k)$ with $d \in \mathscr{D}$ and $l \in L \cap L_d$. 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 $k = 0,\ldots,K_{\text{sim}}$. Next we will refer to one junction $v \in \mathscr{I}$ only. For all other junctions, the switch control actions are determined similarly.

Let $s_v^{\text{in}}(k^{\text{sc}})$ denote the position of the switch-in at junction $v \in \mathscr{I}$ during the time interval $\left[t_{k^{\text{sc}}}^{\text{sw}}, t_{k^{\text{sc}}+1}^{\text{sw}}\right)$, where $t_{k^{\text{sc}}}^{\text{sw}} = t_k + k^{\text{sc}}\tau^{\text{sc}}$ with k^{sc} an integer and τ^{sc} the switch controller sampling time ($t_k = t_0^{\text{sc}}$). Similarly, we define $s_v^{\text{out}}(k^{\text{sc}})$, the position of the switch-out at junction $v \in \mathscr{I}$ during the time interval $\left[t_{k^{\text{sc}}}^{\text{sw}}, t_{k^{\text{sc}}+1}^{\text{sw}}\right)$.

We want to determine the switch control sequence at most until time instant t_{k+1}. However, the prediction period has at most $N_{\max}^{\text{sc}}$ steps. As a consequence, the prediction period for the MPC switch problem at step k^{sc} is defined as $\left[t_{k^{\text{sc}}}^{\text{sw}}, t_{\text{end},k}^{\text{sw}}\right)$ with $t_{\text{end},k}^{\text{sw}} = \min(t_{k+1}, t_{k^{\text{sc}}+N^{\text{sc}}}^{\text{sw}})$.

Hence, at each MPC step k^{sc}, the switch controller solves the optimization problem: $\min_{\mathbf{s}_v} \; J_{k^{\text{sc}},N^{\text{sc}}}^{\text{sw},v,k}(\mathbf{s}_v)$ where

- N^{sc} is the length of the prediction horizon ($N^{\text{sc}} = \frac{t_{\text{end},k}^{\text{sw}}}{\tau^{\text{sc}}}$),
- $\mathbf{s}_v = [s_v^{\text{in}}(k^{\text{sc}}) \ldots s_v^{\text{in}}(k^{\text{sc}}+N^{\text{sc}}-1) \ldots s_v^{\text{out}}(k^{\text{sc}}) \ldots \; s_v^{\text{out}}(k^{\text{sc}}+N^{\text{sc}}-1)]^{\text{T}}$,
- $J_{k^{\text{sc}},N^{\text{sc}}}^{\text{sw},v,k}$ is the local performance index defined next.

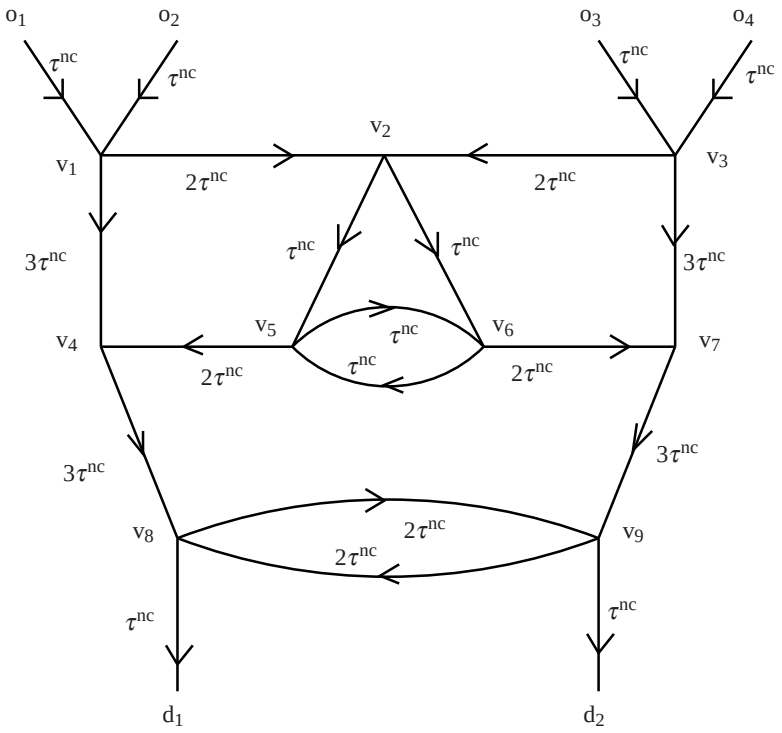


Figure 2.8: Case study for a DCV-based baggage handling system.

Let $X^{\text{opt}}_{v,l,k}$ denote the optimal number of DCVs to enter the outgoing link l of junction v during the period $\left[t^{\text{sw}}_{k^{\text{sc}}}, t^{\text{sw}}_{\text{end},k}\right)$. Then for $l \in L^{\text{out}}_v \cap L_d$ the variable $X^{\text{opt}}_{v,l,k}$ is given by $X^{\text{opt}}_{v,l,k} = (t^{\text{sw}}_{\text{end},k} - t^{\text{sw}}_{k^{\text{sc}}}) \sum_{d \in \mathscr{D}} u^{\text{opt}}_{l,d}(k)$. Next let $X_{v,l,k^{\text{sc}}}$ be the actual number of DCVs entering link l during the prediction period. The variable $X_{v,l,k^{\text{sc}}}$ is determined via simulation for the fast event-driven model of Section 2.3.3 above. Then, at time step k^{sc}, the local performance index is defined as follows:

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

where $n^{\text{sw\_in}}$ and $n^{\text{sw\_out}}$ represent the number of toggles of the switch-in and of the switch-out respectively during the prediction period $\left[t^{\text{sw}}_{k^{\text{sc}}}, t^{\text{sw}}_{\text{end},k}\right)$, which are obtained from simulation, and where γ is a nonnegative weighting parameter.

2.3.7 Case study

In this section we present a simple case study involving a basic set-up to illustrate the network-level control approach for DCV-based baggage handling systems proposed in this section. 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.

Set-up and scenarios

We consider the network of tracks depicted in Figure 2.8 with 4 loading stations, 2 unloading stations, 9 junctions, and 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. 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 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.

We consider 6 different scenarios where 2500 bags have to be handled for different initial states of the system, queues on different links, different piecewise constant demand profiles over the first 180 s of the simulation, and different weighting parameters. We simulate a period of 40 min. 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. In these scenarios we have also considered the occurrence of queues at origin. Assuming that we start the simulation at time instant $t_0 = 0$ s, we consider the time window to be $[800, 1400]$ for destination d_1, and $[1000, 1600]$ for destination d_2.

Results

In this section we compare the results obtained when using the proposed hierarchical control framework and the centralized switch control of [50]. 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 a *genetic* algorithm of the Matlab optimization toolbox *Genetic Algorithm and Direct Search* implemented via the function `ga`, using multiple runs. The same genetic algorithm has been used to solve the optimization problem of the centralized switch control. Essentially the centralized switch control boils down to solving a problem like (2.37) but with $r(k+1), r(k+2), \ldots, r(k+N)$ as optimization variables for each MPC step k, where $r(i)$ is the route of the ith DCV that entered the network (for details see [50]). As prediction horizon we have considered $N = 11$ for the network controller and $N^{\text{sc}} = 15$ for the switch controller of the hierarchical control, and $N = 40$ for the centralized MPC switch control. Note that due to computational requirements reasons, for the switch control of both frameworks 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 (1 hour) for solving an optimization problem corresponding to the centralized switch control.

Based on simulations we now compare, for the given scenarios, the results obtained for the proposed control frameworks. The results of the simulations are reported in Figure 2.9. For this comparison we consider the total performance of the system to be defined as $J = \sum_{d \in \mathscr{D}} \sum_{i=1}^{X_d} |t_{i,d} - t_{i,d}^{\text{desired}}|$ with $t_{i,d}$ the time when the ith bag crossing the junction directly connected to destination d actually crosses that junction, $t_{i,d}^{\text{desired}}$ is the desired crossing time for the same DCV, and X_d the total number of bags to be sent to destination d during the simulation period. The time sequence $t_{1,d}^{\text{desired}}, \ldots, t_{X_d,d}^{\text{desired}}$ with $d \in \mathscr{D}$ is computed such that at each control time step k of the network controller, the $\tau^{\text{nc}} u_d^{\text{desired}}(k)$ bags arrive at equidistant time instants during the period $[t_k, t_{k+1})$.

Using simulations we have obtained an average performance over all scenarios of $9.92 \cdot 10^5$ s for the hierarchical control framework versus a performance of $6.44 \cdot 10^6$ s. So, simulation results confirm that computing the route choice using the hierarchical control framework gives better performance than using the centralized switch control. Hence, the hierarchical control with MILP flow solutions performs better than the centralized switch control, the solution of which was returned by

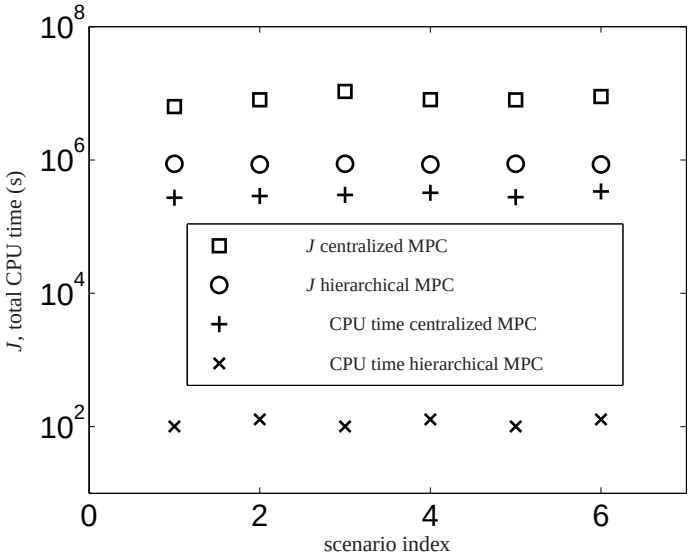


Figure 2.9: Closed-loop results (the smaller J the better system performance).

the prematurely terminated global and multi-start local optimization method.

However, even with these computational restrictions, the total computation time of the centralized switch control (over 62 hours) is much larger than the one of the hierarchical control (an average of 246 s per junction, plus 12 s for solving the MILP optimization problems).

Hence, the proposed hierarchical control outperforms the centralized switch control of [50].

2.3.8 Conclusions

In this section 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 uses a higher-level, aggregated model of the system and 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, which use a more detailed event-based model, 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 minimal. The problem of computing optimal routes for DCVs is a nonlinear, non-convex, mixed integer optimization problem, and very expensive to solve in terms of computational efforts. Therefore, we have 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 control with centralized switch control. Results indicate that the proposed hierarchical control outperforms the centralized switch control where the multi-start local optimization method has been terminated prematurely.

Chapter 3

Design of hierarchical control systems with reconfiguration capabilities

In this chapter the design of a two layer hierarchical controller is considered for cascade systems. The high layer provides for a slow dynamics regulator, computing the reference signals the plant would ideally need to be suitably controlled. In turn, at the low layer, a number of faster actuation control loops are in charge of tracking such references as accurately as they can, in accordance to their dynamics, which for simplicity is supposed here to be first order. Because of several inaccuracies, stemming from the real behavior of the actuation equipment, a discrepancy between the ideal control actions determined at the high level and those effectively afforded to the plant arises, leading to a robustness problem for the overall control system. To tackle this problem, the upper level controller is designed by resorting to a robust MPC approach. In so doing, a convergence result for the overall closed-loop system is derived. The structure here considered can be viewed as a particular case (cascade systems) of the more general structure for hierarchical control previously described in Deliverable D2.2.
In order to emphasize the reconfiguration capabilities of optimization-based predictive controllers in response to changes in the subsystems (actuators), it is then shown how the proposed MPC algorithm may be readily extended to cope with the self reconfiguration of the controller, owing to an actuator replacement/addition. It can thus take a significant role also within the "Plug and Play" research community, which studies the problem of control reconfiguration when a new device, in general a sensor or an actuator, is plugged/substituted into an already functioning control system (see the very recent works [8, 9, 32]).
The results reported in this chapter will be partially presented in [55], where also some simulation experiments will be reported and discussed.

Notation. In the mathematical developments of the proposed algorithm, we will consider two time scales: in particular, we will denote the fast discrete-time index by h, while we will represent the slow discrete-time index by k. By $\|\cdot\|$ we denote the Euclidean vector or induced matrix norm. For $x \in \mathbb{R}^n$ and $\mathbb{R}^{n\times n} \ni P > 0$, we let $\|x\|_P = \sqrt{x'Px}$

3.1 Problem formulation

Let us consider a discrete-time linear system modeling a plant operated by m actuators, whose dynamical behavior in the fast time scale is given by

$$\mathscr{P} : x^f(h+1) = A^f x^f(h) + \sum_{i=1}^{m} b_i^f u_i^f(h) \tag{3.1}$$

where $x^f \in \mathbb{R}^{n_x}$ is the measurable state while $u_i^f \in \mathbb{R}$ stands for the action provided by the i-th actuator.

Throughout this chapter, we suppose that simple (e.g., PI-like) control loops acting on all the systems placed at the low level (i.e., the actuators) are *a-priori* designed and already working. Moreover, we assume that the stemming closed-loop systems can be depicted by first order unitary gain SISO models of the form

$$\mathscr{S}_{\text{act}\,i} : \begin{cases} \zeta_i(h+1) = f_i\zeta_i(h) + (1-f_i)u_i(h), \quad \zeta_i(0) = \zeta_{i0} \\ \hat{u}_i(h) = \zeta_i(h) \\ |f_i| < 1 \end{cases} \tag{3.2}$$

$\forall i = 1,\dots,m$, whose output variables $\hat{u}_i$'s coincide with the inputs u_i^f's of system (3.1).

The control objective consists of achieving a stabilizing control law for the cascade interconnection of systems (3.1) and (3.2). To this end, we propose a two-level hierarchical regulator. The high level provides for a controller working at a slow time scale and computing the reference signal u_i to be tracked by each actuator control loop. In turn, at the low level, all the actuators concur to drive the plant, tracking their own reference signal in accordance to their closed loop dynamics (3.2). For this reason, one in general has $\hat{u}_i \neq u_i$, at least in transient conditions, so consequently a robustness problem arises. To cope with this problem, we consider the discrepancy between the ideal control actions and those effectively afforded to the process as a disturbance term the high level controller has to be robust to.

3.1.1 Model of the plant in the slow time scale

As for system (3.1), we consider the control constraints

$$u_i^f \in \mathscr{U}_i = [-\alpha_i, \alpha_i],\ \alpha_i > 0 \quad \forall i = 1,\dots,m. \tag{3.3}$$

In addition, we let $\mathscr{U} = \mathscr{U}_1 \times \cdots \times \mathscr{U}_m \subset \mathbb{R}^m$ and

$$B^f = \begin{bmatrix} b_1^f & b_2^f & \cdots & b_m^f \end{bmatrix} \in \mathbb{R}^{n_x \times m}. \tag{3.4}$$

Assumption 1 *The pair (A^f, B^f) is stabilizable.* ♦

For some fixed integer $\tau \geq 1$, let us decompose the control variables u_i^f's of system (3.1) in the form

$$u_i^f(h) = \bar{u}_i(h) + \big(u_i^f(h) - \bar{u}_i(h)\big),$$

$\bar{u}_i(h) \in \mathscr{U}_i$ being some piecewise constant signals, i.e., $\forall k \in \mathbb{N}$ and $\forall j = 0,\dots,\tau-1$, it holds that $\bar{u}_i(\tau k + j) = \bar{u}_i(\tau k)$. Then system (3.1) can be rewritten as

$$x^f(h+1) = A^f x^f(h) + \sum_{i=1}^{m} b_i^f \bar{u}_i(h) + \sum_{i=1}^{m} b_i^f w_i^f(h) \tag{3.5}$$

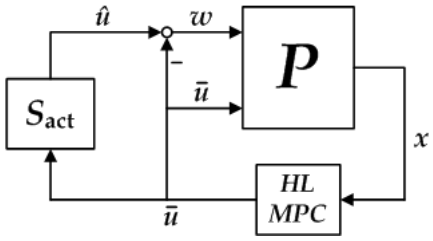


Figure 3.1: Hierarchical control scheme (P: plant, *HL MPC*: high level MPC, S_{act}: actuators).

where

$$w_i^f(h) = u_i^f(h) - \bar{u}_i(h)$$

is considered as a matched disturbance term.

Letting $x(k) = x^f(\tau k)$, $u_i(k) = \bar{u}_i(\tau k)$ and

$$A = (A^f)^\tau, \qquad b_i = \sum_{j=0}^{\tau-1} (A^f)^{\tau-j-1} b_i^f, \qquad w_i(k) = \sum_{j=0}^{\tau-1} (A^f)^{\tau-j-1} b_i^f w_i^f(\tau k + j), \tag{3.6}$$

the sampled version of (3.5) in the slow sampling rate is

$$x(k+1) = Ax(k) + \sum_{i=1}^{m} b_i u_i(k) + \sum_{i=1}^{m} w_i(k) \tag{3.7}$$

which, in turn, translates in the vector form

$$\mathscr{P}_{\text{slow}} : \; x(k+1) = Ax(k) + B_1 u(k) + B_2 w(k) \tag{3.8}$$

once the following definitions have been stated:

$$u(k) = \left[\begin{array}{ccc} u_1'(k) & \dots & u_m'(k) \end{array} \right]' \in \mathscr{U} \subset \mathbb{R}^m \tag{3.9a}$$

$$w(k) = \left[\begin{array}{ccc} w_1'(k) & \dots & w_m'(k) \end{array} \right]' \in \mathbb{R}^{mn_x} \tag{3.9b}$$

$$B_1 = \left[\begin{array}{cccc} b_1 & b_2 & \dots & b_m \end{array} \right] \in \mathbb{R}^{n_x \times m} \tag{3.9c}$$

$$B_2 = \left[\begin{array}{cccc} I_{n_x} & I_{n_x} & \dots & I_{n_x} \end{array} \right] \in \mathbb{R}^{n_x \times mn_x}. \tag{3.9d}$$

In conclusion, in view of the linear nature of the problem considered here, the plant to be controlled can be viewed as driven, on one hand, by the ideal control commands coming from the high level controller, $\bar{u}_i(h)$'s (namely, $u_i(k)$'s in the slow sampling rate), and, on the other hand, by the discrepancies between such commands and the effective control signals achieved by the actuators, $w_i^f(h)$'s (namely, $w_i(k)$'s in the slow sampling rate). Moreover, the latter differ from the usual disturbance term affecting the system dynamics in robust control designs, in that they originate from the high level control action itself, rather than being afforded by the "nature", as schematically portrayed in Figure 3.1.

In the light of such a reformulation of the problem, at the high level we address the design of a robustly stabilizing MPC controller for system (3.8) in the face of the disturbance $w(k)$, achieving the piecewise constant signals $\bar{u}_i(h)$, i.e., the references for the low level actuators' loops.

3.1.2 The hierarchical controller

The hierarchical controller involves a high level robust MPC algorithm, providing the references for the actuators at any long sampling instant k. Hence, in order to guarantee that the control signals $\hat{u}_i(h)$'s, accomplished by the low level systems against such references, satisfy the control constraints on the process (3.3), the following hypothesis must be stated.

Assumption 2 *For all* $i = 1, \ldots, m$

1. $\zeta_i(0) \in \mathscr{U}_i$;
2. $u_i \in [-\varepsilon_i, \varepsilon_i]$ *where*

$$\varepsilon_i = \frac{\alpha_i(1-|f_i|)}{1-f_i}.$$

♦

For later use, we define $\tilde{\mathscr{U}}_i = [-\varepsilon_i, \varepsilon_i]$ and

$$\tilde{\mathscr{U}} = \tilde{\mathscr{U}}_1 \times \ldots \times \tilde{\mathscr{U}}_m \subset \mathbb{R}^m.$$

Moreover, the feasibility of the control references computed at the high level needs the following assumption.

Assumption 3 *The high level MPC controller incorporates the actuators' models, i.e., it knows the parameters* f_i*'s along with the initial conditions* $\zeta_i(0)$*'s,* $i = 1, \ldots, m$. ♦

Assumption 3 implies that the high level controller can predict the low level systems' behavior, so consequently being aware of their current state situation at any time instant. We can thus define the augmented version of system (3.8):

$$\chi(k+1) = \tilde{A}\chi(k) + \tilde{B}_1 u(k) + \tilde{B}_2 w(k) \tag{3.10}$$

where the new state variable and the corresponding state space matrices are

$$\begin{gathered} \chi(k) = \begin{bmatrix} x(k) \\ \tilde{u}(k) \end{bmatrix} \in \mathscr{X} = \mathbb{R}^{n_x} \times \mathscr{U}, \\ \tilde{A} = \begin{bmatrix} A & 0_{n_x,m} \\ 0_{m,n_x} & F \end{bmatrix}, \quad \tilde{B}_1 = \begin{bmatrix} B_1 \\ G \end{bmatrix}, \quad \tilde{B}_2 = \begin{bmatrix} B_2 \\ 0_{m,mn_x} \end{bmatrix} \end{gathered} \tag{3.11}$$

in which

$$\tilde{u}(k) = \begin{bmatrix} \tilde{u}_1'(k) & \ldots & \tilde{u}_m'(k) \end{bmatrix}' \tag{3.12a}$$

$$F = \operatorname{diag}(f_1^\tau, \ldots, f_m^\tau) \tag{3.12b}$$

$$G = \operatorname{diag}\big((1-f_1^\tau), \ldots, (1-f_m^\tau)\big). \tag{3.12c}$$

Moreover, the $\tilde{u}_i(k)$'s, $i = 1, \ldots m$, in (3.12a) are the slow time rate counterparts of the fast time rate control actions performed by the actuators (i.e., $\tilde{u}_i(k) = \hat{u}_i(\tau k)$), which stem from output propagation of systems (3.2) as follows

$$\hat{u}_i\big(\tau(k+1)\big) = f_i^\tau \hat{u}_i(\tau k) + (1-f_i^\tau)\bar{u}_i(\tau k) \tag{3.13}$$

$\bar{u}_i(\tau k)$ standing for the i-th piecewise constant reference entry generated by the upper level controller, namely

$$\bar{u}_i(h) = u_i\big(\lfloor \tfrac{h}{\tau} \rfloor\big) \tag{3.14}$$

in which the floor function $\lfloor \cdot \rfloor$ has been used.

Furthermore, associated with the dynamic equation (3.10), we introduce also the output transformation

$$z(k) = \begin{bmatrix} \chi(k) \\ u(k) \end{bmatrix}. \tag{3.15}$$

Finally, to gain versatility in the definition of the cost function the MPC paradigm calls for, we introduce weighted norms in both the state and control spaces. Thus, for fixed symmetric and positive definite matrices $Q_x \in \mathbb{R}^{n_x \times n_x}$ and $Q_i \in \mathbb{R}$, $i = 1, \ldots, m$, we let:

$$\begin{array}{rcl} Q_u & = & \mathrm{diag}\{Q_1, \cdots, Q_m\} \in \mathbb{R}^{m \times m} \\ Q_{ii} & = & \mathrm{diag}\{Q_i, Q_i\} \in \mathbb{R}^{2 \times 2} \\ Q_z & = & \mathrm{diag}\{Q_x, Q_u, Q_u\} \in \mathbb{R}^{(n_x+2m)\times(n_x+2m)} \\ Q_w & = & \mathrm{diag}\{Q_x, \cdots, Q_x\} \in \mathbb{R}^{mn_x \times mn_x} \\ Q_\chi & = & \mathrm{diag}\{Q_x, Q_u\} \in \mathbb{R}^{(n_x+m)\times(n_x+m)}. \end{array}$$

3.2 Design and analysis of the high level controller in the basic actuation configuration

In this section, we deal with the design of the high level MPC controller for the basic low level configuration, i.e., the one including m actuators. In the following, we will discuss how such a controller can be extended to readily allow for a self reconfiguration subsequent to actuators' addition/replacement.

Thus, according both to Assumption 3 and to the output propagation (3.13), the high level controller can easily compute each matched disturbance term $w_i(k)$ appearing in (3.7) as follows

$$\begin{aligned} w_i(k) &= \sum_{j=0}^{\tau-1} (A^f)^{\tau-j-1} b_i^f \left[f_i^j \hat{u}_i(\tau k) - f_i^j \bar{u}_i(\tau k) \right] \\ &= \big(\hat{u}_i(\tau k) - \bar{u}_i(\tau k)\big) \sum_{j=0}^{\tau-1} (A^f)^{\tau-j-1} b_i^f f_i^j \\ &= \big(\tilde{u}_i(k) - u_i(k)\big) \vartheta_i \end{aligned} \tag{3.16}$$

where

$$\vartheta_i = \sum_{j=0}^{\tau-1} (A^f)^{\tau-j-1} b_i^f f_i^j \tag{3.17}$$

is an *a-priori* known vector for every actuator. It turns out that the $w_i(k)$ terms are linear functions of

the discrepancies $\big(\tilde{u}_i(k)-u_i(k)\big)$ and satisfy gain conditions of the form:

$$\begin{aligned}\|w_i(k)\|_{Q_x} &\leq \sqrt{\lambda_{\max}(Q_x)}\|w_i(k)\| \\ &= \sqrt{\lambda_{\max}(Q_x)}\|\vartheta_i\|\cdot\big|\big(\tilde{u}_i(k)-u_i(k)\big)\big| \\ &\leq \sqrt{2\lambda_{\max}(Q_x)}\|\vartheta_i\|\cdot\left\|\begin{bmatrix}\tilde{u}_i(k)\\ u_i(k)\end{bmatrix}\right\| \\ &\leq \sqrt{\frac{2\lambda_{\max}(Q_x)}{Q_i}}\|\vartheta_i\|\cdot\left\|\begin{bmatrix}\tilde{u}_i(k)\\ u_i(k)\end{bmatrix}\right\|_{Q_{ii}} \\ &= \gamma_{\mathrm{d}}(i)\left\|\begin{bmatrix}\tilde{u}_i(k)\\ u_i(k)\end{bmatrix}\right\|_{Q_{ii}}\end{aligned} \tag{3.18}$$

where the $\gamma_{\mathrm{d}}(i)$'s are available to the upper level controller in view of Assumption 3. As a consequence, the high level regulator can be carried out via the small-gain approach, by considering w as a disturbance term satisfying a gain condition of the type $\|w\|_{Q_w} \leq \gamma_{\mathrm{d}}\|z\|_{Q_z}$ (for a suitable γ_{d}).

In the sequel, we will discuss first a robustly stabilizing auxiliary control law and later an MPC controller improving both performance and region of attraction such an auxiliary law achieves.

3.2.1 The auxiliary law

Under Assumption 1, we can construct an auxiliary control law for system (3.10) taking the form

$$u(k) = K_{\mathrm{aux}}\chi(k), \quad K_{\mathrm{aux}} \in \mathbb{R}^{m\times(n_x+m)}. \tag{3.19}$$

To this end, we consider $\gamma > 0$ such that there exists a symmetric and positive definite matrix $P \in \mathbb{R}^{(n_x+m)\times(n_x+m)}$ satisfying the Riccati inequality with constraint

$$\begin{aligned}&-P+\tilde{A}'P\tilde{A}+Q_\chi-\tilde{A}'P\tilde{B}R^{-1}\tilde{B}'P\tilde{A}<0\\ &\tilde{B}_2'P\tilde{B}_2-\gamma^2Q_w<0\end{aligned} \tag{3.20}$$

where

$$\begin{aligned}\tilde{B} &= \begin{bmatrix}\tilde{B}_1 & \tilde{B}_2\end{bmatrix}\\ R &= \begin{bmatrix}\tilde{B}_1'P\tilde{B}_1+Q_u & \tilde{B}_1'P\tilde{B}_2\\ \tilde{B}_2'P\tilde{B}_1 & \tilde{B}_2'P\tilde{B}_2-\gamma^2Q_w\end{bmatrix}\end{aligned}$$

and we let

$$K_{\mathrm{aux}} = -\begin{bmatrix}I_m & 0_{m,mn_x}\end{bmatrix}R^{-1}\tilde{B}'P\tilde{A}.$$

Moreover, we define the function $V_f(\chi)=\chi'P\chi$ and for any $\rho>0$, the set

$$\Omega_\rho = \{\chi\in\mathbb{R}^{n_x+m}\,|\,V_f(\chi)\leq\rho^2\}\subset\mathbb{R}^{n_x+m}.$$

The local robust stabilization properties of the auxiliary control law (3.19) are clarified by the following result.

Proposition 1 *Define*

$$\overline{\gamma}_{\mathrm{d}} = \max_{i,\ldots,m}\gamma_{\mathrm{d}}(i). \tag{3.21}$$

Let $\gamma>0$ be such that $\gamma\cdot\overline{\gamma}_{\mathrm{d}}<1$ and assume that a positive definite solution P for the Riccati inequality (3.20) *exists. Consider system* (3.10) *under the corresponding control law* (3.19) *and, accordingly,*

let $w(k)$ be the vector collecting all the disturbance terms given in (3.16)*. Then, $\exists \rho > 0$ such that the following properties hold $\forall \chi \in \Omega_{\text{aux}}$, $\Omega_{\text{aux}} = \Omega_\rho$:*

$$\Omega_{\text{aux}} \in \mathscr{X} \tag{3.22a}$$

$$K_{\text{aux}}\chi \in \tilde{\mathscr{U}}; \tag{3.22b}$$

$$\|w(k)\|_{Q_w} \leq \overline{\gamma}_{\text{d}} \|z(k)\|_{Q_z} \tag{3.22c}$$

$$\forall \chi(k) \in \Omega_{\text{aux}},\ V_f\big(\chi(k+1)\big) - V_f\big(\chi(k)\big) \leq -\big(\|z(k)\|^2_{Q_z} - \gamma^2 \|w(k)\|^2_{Q_w}\big) \tag{3.22d}$$

$$\Omega_{\text{aux}} \textit{ is positively invariant.} \tag{3.22e}$$

□

Proof 1 *Properties* (3.22a) *and* (3.22b) *are guaranteed by the choice of a sufficiently small $\rho > 0$, since $P > 0$, $\tilde{\mathscr{U}}$ is a neighborhood of the origin and, for the latter one, the control law $u = K_{\text{aux}}\chi$ is continuous.*

As far as property (3.22c) *is concerned, notice that*

$$\|w(k)\|^2_{Q_w} = \sum_{i=1}^{m} \|w_i(k)\|^2_{Q_x} \leq \sum_{i=1}^{m} \overline{\gamma}^2_{\text{d}}(i) \left\| \begin{bmatrix} \tilde{u}_i(k) \\ u_i(k) \end{bmatrix} \right\|^2_{Q_{ii}} \leq \overline{\gamma}^2_{\text{d}} \sum_{i=1}^{m} \left\| \begin{bmatrix} \tilde{u}_i(k) \\ u_i(k) \end{bmatrix} \right\|^2_{Q_{ii}} \leq \overline{\gamma}^2_{\text{d}} \, \|[z(k)]\|^2_{Q_z}$$

where the upper bound (3.18) *has been used.*

The proof of property (3.22d) *can be found in [40].*

Finally, the positive invariance of Ω_{aux} is guaranteed by (3.22) *and the small-gain condition $\gamma \cdot \overline{\gamma}_{\text{d}} < 1$. In fact,* (3.22b) *ensures the satisfaction of the input constraints* (3.3)*, while inequalities* (3.22c), (3.22d) *and $\gamma \cdot \overline{\gamma}_{\text{d}} < 1$ ensure that $V_f\big(\chi(k+1)\big) \leq V_f\big(\chi(k)\big)$.* ■

3.2.2 The MPC controller

In this Section, we improve both the region of attraction Ω_{aux} and the performance provided by the auxiliary control law (3.19), by resorting to the small-gain paradigm. In particular, according to Assumption 3 and the gain condition (3.18), we derive a robustly stabilizing high level MPC control law fulfilling the norm bound $\|w\|_{Q_w} \leq \overline{\gamma}_{\text{d}} \|z\|_{Q_z}$ (where $\overline{\gamma}_{\text{d}}$ is given in (3.21)).

In details, we let $N_p \in \mathbb{N}$, $N_p \geq 1$, be the length of the prediction horizon and $N_c \in \mathbb{N}$, $N_c \leq N_p$, be the length of the control horizon. Moreover, we define

$$\mathscr{F}(k, N_c) = \begin{bmatrix} u_{(k)}(k) & u_{(k)}(k+1) & \dots & u_{(k)}(k+N_c-1) \end{bmatrix}, \tag{3.23}$$

where $u_{(k)}(k+j) \in \tilde{\mathscr{U}}$ is the vector of the predicted control signals to be processed by the MPC algorithm at time k. At any time instant k, the control problem consists of solving the following optimization problem:

$$\min_{\mathscr{F}(k,N_c)} J\big(\chi(k), \mathscr{F}, N_p\big), \tag{3.24}$$

$$J\big(\chi(k), \mathscr{F}, N_p\big) = \sum_{j=0}^{N_p-1} \big(\|z(k+j)\|^2_{Q_z} - \gamma^2 \|w(k+j)\|^2_{Q_w}\big) + V_f\big(\chi(k+N_p)\big),$$

subject to:

(i) system (3.10), (3.15), (3.16) under the control signal (3.23) and, for $j = N_c, \dots, N_p - 1$, $u(k+j) = K_{\text{aux}}\chi(k+j)$;

(ii) the control constraints: $\forall j = 0,\dots,N_c-1, u(k+j) \in \tilde{\mathscr{U}}$;

(iii) the terminal constraint $\chi(k+N_p) \in \Omega_{\text{aux}}$.

If

$$\mathscr{F}^o\big(\chi(k),N_c\big) = \left[\begin{array}{cccc} u^o_{(k)}(k) & u^o_{(k)}(k+1) & \dots & u^o_{(k)}(k+N_c-1) \end{array}\right]$$

is the optimal solution to this problem, according to the Receding Horizon principle, we define the MPC control law as

$$u\big(\chi(k)\big) = u^o_{(k)}(k). \tag{3.25}$$

Theorem 1 *Under the assumptions of Proposition 1, let $X^{MPC}(N_c,N_p)$ be the set of states such that problem* (3.24) *admits a solution. Then $\forall N_p \geq 1$ and $\forall N_c \leq N_p$, one has:*

1. $\Omega_{\text{aux}} \subseteq X^{MPC}(N_c,N_p)$,
2. $X^{MPC}(N_c,N_p)$ *is a positively invariant set,*
3. *the origin is a locally exponentially stable equilibrium point with region of attraction $X^{MPC}(N_c,N_p)$,*

where properties 2) and 3) hold for the closed-loop system (3.10), (3.25), (3.16). □

Proof 2 *The theorem is proved for $N_c \geq 1$ (the case $N_c = 0$ easily follows by Proposition 1).*

1. $\Omega_{\text{aux}} \subseteq X^{MPC}(N_c,N_p)$ *because, by properties* (3.22)*, the auxiliary law is feasible for $\chi \in \Omega_{\text{aux}}$ and Ω_{aux} is positively invariant.*
2. *If $\chi(k) \in X^{MPC}(N_c,N_p)$, then there exists $\mathscr{F}^o$ such that $\chi(k+N_p) \in \Omega_{\text{aux}}$. Thus, at time $k+1$, consider the following control signal:*

$$\hat{\mathscr{F}}(k+1,N_c) = \left[\begin{array}{ccccc} u^o_{(k)}(k+1) & \cdots & u^o_{(k)}(k+N_c-1) & K_{\text{aux}}\chi(k+N_c) \end{array}\right].$$

   This policy is still feasible for $\chi(k+1)$, and hence $X^{MPC}(N_c,N_p)$ is a positively invariant set, because under the auxiliary control law Ω_{aux} is positively invariant.
3. *Let $V\big(\chi(k),N_c,N_p\big) = J\big(\chi(k),\mathscr{F}^o,N_c,N_p\big)$ be the optimal performance starting from $\chi(k)$. In view of the well known Theorem III.2 proved in [36] and since $\|z(k)\|^2_{Q_z} \geq \|\chi(k)\|^2_{Q_x}$, the stability result holds if the following properties are satisfied:*

$$\chi \in X^{MPC}(N_c,N_p) \Rightarrow\Rightarrow \quad V(\chi,N_c,N_p) \geq \big(1-\gamma^2\cdot\bar{\gamma}_{\text{d}}^2\big)\|\chi\|^2_{Q_\chi}; \tag{3.26a}$$

$$\chi \in \Omega_{\text{aux}} \Rightarrow V(\chi,N_c,N_p) \leq \tfrac{\lambda_{\max}(P)}{\lambda_{\min}(Q_\chi)}\|\chi\|^2_{Q_\chi}; \tag{3.26b}$$

$$\chi(k) \in X^{MPC}(N_c,N_p) \Rightarrow V\big(\chi(k+1),N_c,N_p\big) - V\big(\chi(k),N_c,N_p\big) \leq -\big(1-\gamma^2\cdot\bar{\gamma}_{\text{d}}^2\big)\|z(k)\|^2_{Q_z}. \tag{3.26c}$$

   As far as inequality (3.26a) *is concerned, for $\chi(k) \in X^{MPC}(N_c,N_p)$, one has*

$$V\big(\chi(k),N_c,N_p\big) = \sum_{j=0}^{N_p-1}\big(\|z(k+j)\|^2_{Q_z} - \gamma^2\|w(k+j)\|^2_{Q_w}\big) + V_f\big(\chi(k+N_p)\big) \geq$$
$$\overset{\text{(a)}}{\geq}\big(1-\gamma^2\cdot\bar{\gamma}_{\text{d}}^2\big)\|z(k)\|^2_{Q_z} \geq \big(1-\gamma^2\cdot\bar{\gamma}_{\text{d}}^2\big)\big)\|\chi(k)\|^2_{Q_\chi},$$

where inequality (a) holds because, in view of the gain condition expressed by (3.18) *and* (3.21)*, it holds that,*

$$\forall j = 0,\dots,N_p-1, \quad \|z(k+j)\|^2_{Q_z} - \gamma^2\|w(k+j)\|^2_{Q_w} \geq \left(1-\gamma^2\cdot\overline{\gamma}_{\mathrm{d}}^2\right)\|z(k+j)\|^2_{Q_z}.$$

To prove inequality (3.26b)*, we first show that, if* $\chi \in X^{MPC}(N_c,N_p)$*, then*

$$V(\chi,N_c+1,N_p+1) \leq V(\chi,N_c,N_p). \tag{3.27}$$

Indeed, consider the control signal

$$\widetilde{\mathscr{F}}(k,N_c+1) = \left[\begin{array}{cc} \mathscr{F}^o\big(\chi(k),N_c\big) & K_{\mathrm{aux}}\chi(k+N_c) \end{array}\right],$$

then

$$\begin{aligned} J\big(\chi(k),\widetilde{\mathscr{F}}(k,N_c+1),N_c+1,N_p+1\big) = \sum_{j=0}^{N_p-1}\big(\|z(k+j)\|^2_{Q_z} - \gamma^2\|w(k+j)\|^2_{Q_w}\big) + \\ + V_f\big(\chi(k+N_p)\big) + \\ - V_f\big(\chi(k+N_p)\big) + V_f\big(\chi(k+N_p+1)\big) + \\ + \big(\|z(k+N_p)\|^2_{Q_z} - \gamma^2\|w(k+N_p)\|^2_{Q_w}\big). \end{aligned}$$

Since $\chi(k+N_p) \in \Omega_{\mathrm{aux}}$ *and the value of the output* $z(k+N_p)$ *is obtained with the auxiliary control law used at time* $k+N_p$*, using inequality* (3.22d)*, one has*

$$\begin{aligned} J\big(\chi(k),\widetilde{\mathscr{F}}(k,N_c+1),N_c+1,N_p+1\big) \leq \sum_{j=0}^{N_p-1}\big(\|z(k+j)\|^2_{Q_z} - \gamma^2\|w(k+j)\|^2_{Q_w}\big) + \\ + V_f\big(\chi(k+N_p)\big) = V\big(\chi(k),N_c,N_p\big). \end{aligned}$$

Consequently,

$$V\big(\chi(k),N_c+1,N_p+1\big) \leq V\big(\chi(k),N_c,N_p\big),$$

thus proving the (3.27)*.*

Now, $\forall\,\chi(k) \in \Omega_{\mathrm{aux}}$*,*

$$\begin{aligned} V\big(\chi(k),N_c,N_p\big) &\overset{\text{(b)}}{\leq} V\big(\chi(k),0,N_p-N_c\big) \\ &= \sum_{j=0}^{N_p-N_c-1}\big(\|z(k+j)\|^2_{Q_z} - \gamma^2\|w(k+j)\|^2_{Q_w}\big) + V_f\big(\chi(k+N_p-N_c)\big) \\ &\overset{\text{(c)}}{\leq} \sum_{j=0}^{N_p-N_c-1}\Big(V_f\big(\chi(k+j)\big) - V_f\big(\chi(k+j+1)\big)\Big) + V_f\big(\chi(k+N_p-N_c)\big) \\ &= V_f\big(\chi(k)\big) \leq \lambda_{\max}(P)\|\chi(k)\|^2 \leq \frac{\lambda_{\max}(P)}{\lambda_{\min}(Q_\chi)}\|\chi\|^2_{Q_\chi}, \end{aligned}$$

where inequality (b) follows by iterating the (3.27) *(notice that* $\forall\, N_c \leq N_p$*,* $\Omega_{\mathrm{aux}} \subseteq X^{MPC}(N_c,N_p)$*) and inequality (c) holds in view of inequality* (3.22d) *(which can be applied because the length of the control horizon is 0 and, over the prediction horizon, the system evolves under the auxiliary law).*

Finally, let us prove inequality (3.26c)*: observe that*

$$V\big(\chi(k),N_c,N_p\big)=\|z(k)\|^2_{Q_z}-\gamma^2\|w(k)\|^2_{Q_w}+V\big(\chi(k+1),N_c-1,N_p-1\big),$$

then, in view of (3.27)*,*

$$V\big(\chi(k+1),N_c,N_p\big)-V\big(\chi(k),N_c,N_p\big)\leq-\big(\|z(k)\|^2_{Q_z}-\gamma^2\|w(k)\|^2_{Q_w}\big).$$

Hence, once again by (3.18) *and* (3.21)*,*

$$V\big(\chi(k+1),N_c,N_p\big)-V\big(\chi(k),N_c,N_p\big)\leq-\big(1-\gamma^2\cdot\vec{\gamma}_{\rm d}^2\big)\|z(k)\|^2_{Q_z}.$$

■

3.2.3 The overall system: convergence analysis

The following result provides the analysis of the overall control system behavior.

Theorem 2 *Under the assumptions of Theorem 1, consider the closed loop system* (3.1)*,* (3.14) *and* (3.2)*, where the upper level controller is defined in* (3.25)*. Assume that, at time* $h=0$*, the initial states of the actuators* (3.2) *fulfill Assumption 2.1,* $i=1,\dots,m$*. Let*

$$\mu_0=\left[\begin{array}{c}u_{10}\\u_{20}\\\vdots\\u_{m0}\end{array}\right].$$

Assume also that the MPC controller at the upper level is initialized with

$$\chi(0)=\left[\begin{array}{c}x(0)\\\mu_0\end{array}\right]\in\mathscr{X}^{MPC}(N_c,N_p).$$

Then it holds that

$$\left\{\begin{array}{l}\lim_{h\to+\infty}x^f(h)=0\\\lim_{h\to+\infty}\zeta_i(h)=0,\quad\forall i=1,\dots,m.\end{array}\right.$$

□

Proof 3 *Since* $\chi(0)\in\mathscr{X}^{MPC}(N_c,N_p)$ *then, by Theorem 1, it holds that* $\lim_{k\to+\infty}\chi(k)=0$*. This means that*

$$\left\{\begin{array}{l}\lim_{k\to+\infty}x(k)=0\\\lim_{k\to+\infty}u(k)=0,\end{array}\right.$$

and, according to equation (3.14)*,*

$$\lim_{h\to+\infty}\bar{u_i}(h)=0\quad\forall i=1,\dots,m.$$

Hence, in view of the stability of the low level control loops, such a convergence to zero translates into the convergence to zero of the internal state of the actuators, namely

$$\lim_{h\to+\infty}\zeta_i(h)=0,\quad\forall i=1,\dots,m$$

and, as a consequence, $\lim_{h\to+\infty}\hat{u}_i(h)=0$.

Finally, we have to prove that $\lim_{h\to+\infty}x^f(h)=0$. *To this end, it is sufficient to show that,* $\forall l=1,\dots,\tau-1$, *it holds that* $\lim_{k\to+\infty}\|x^f(\tau k+l)\|=0$. *Indeed, combining equations* (3.1) *and* (3.4), *one has*

$$x^f(\tau k+l)=(A^f)^l x^f(\tau k)+\sum_{j=0}^{l-1}(A^f)^{l-j-1}B^f\hat{u}(\tau k+j),$$

where

$$\hat{u}(h)=\begin{bmatrix}\hat{u}_1(h)\\ \hat{u}_2(h)\\ \dots\\ \hat{u}_m(h)\end{bmatrix}.$$

Thus, $\forall l=1,\dots,\tau-1$, *it holds that*

$$\|x^f(\tau k+l)\|\leq\|(A^f)^l\|\cdot\|x^f(\tau k)\|+\sum_{j=0}^{l-1}\|(A^f)^{l-j-1}B^f\|\cdot\|\hat{u}(\tau k+j)\|.$$

As $x^f(\tau k)=x(k)$, *it holds that*

$$\lim_{k\to+\infty}x(k)=0$$

and

$$\forall i=1,\dots,m,\quad\lim_{h\to+\infty}\hat{u}_i(h)=0,$$

so the thesis follows. ■

3.3 Control system reconfiguration

According to the receding horizon paradigm, the high level MPC controller stated in Theorem 1 calls for solving the optimization problem (3.24) at each time step. It can thus easily allow for a "Plug and Play" flexibility for actuators' addition/replacement. To this end, assuming that only one actuator per (slow) time step can be plugged/substituted into the overall system, plug and play fashion is guaranteed as follows.

Actuator addition

As an actuator addition happens, variables χ, u, $\tilde{u}$, w and z contain additional components accounting for such a new actuator. Hence, we introduce the notation $\eta^{(m+1)}$ to denote the variable η related to the new low level configuration, i.e., the one composed of $m+1$ actuators. Accordingly, we let

$$\tilde{\mathscr{U}}^{(m+1)}=\tilde{\mathscr{U}}\times[-\varepsilon_{m+1},\varepsilon_{m+1}]$$

be the corresponding set of admissible control commands.

Let us discuss how the high level controller, hence problem (3.24), can be reconfigured after the actuator addition.

The auxiliary control law (3.19) is largely independent of the low level dynamics. As a consequence, even if it has to feed also the plugged actuator, it can send it a null reference, so keeping unchanged.

In details, letting

$$u^{(m+1)} = K_{\text{aux}}^{(m+1)} \chi^{(m+1)} = \begin{bmatrix} K_{\text{aux}} & 0_{m\times 1} \\ 0_{1\times(n_x+m)} & 0 \end{bmatrix} \chi^{(m+1)} \tag{3.28}$$

and

$$\Omega_{\text{aux}}^{(m+1)} = \Omega_{\text{aux}} \times \{0\}, \tag{3.29}$$

it is straightforward to show that the feasibility of the auxiliary law (3.28) holds in $\Omega_{\text{aux}}^{(m+1)}$ defined in (3.29).

Moreover, within the enlarged setting, problem (3.24) has to be rephrased in terms of the $\eta^{(m+1)}$ variables and the auxiliary law (3.28). Care has only to be payed in the following issues:

1. V_f should be replaced by $V_f^{(m+1)}$, where

$$V_f^{(m+1)}(\chi^{(m+1)}) = \chi^{(m+1)'} P^{(m+1)} \chi^{(m+1)},$$

   $P^{(m+1)}$ being matrix P bordered with a null row and column;

2. constraint ***(ii)*** becomes

$$u^{(m+1)}(k+j) \in \tilde{\mathscr{U}}^{(m+1)}, \quad \forall j = 0,\dots,N_c-1,$$

   while ***(iii)*** translates in

$$\chi^{(m+1)}(k+N_p) \in \Omega_{\text{aux}}^{(m+1)};$$

3. if $\gamma_{\text{d}}(m+1) > \bar{\gamma}_{\text{d}}$, in order to still guarantee the exponential stability of the origin, the optimization problem has to be solved under the further constraint

$$\|w^{(m+1)}(k+j)\|^2_{Q_w^{(m+1)}} \le \bar{\gamma}_{\text{d}}^2 \|z^{(m+1)}(k+j)\|^2_{Q_z^{(m+1)}}, \quad \forall j = 0,\dots,N_c-1. \tag{3.30}$$

With this position, Theorem 1 holds true and the overall control system self reconfigures, only requiring the minor formal adjustments mentioned above.

As far as the feasibility issue of the high level control inputs is concerned, if, at time $k = \bar{k}$, a new actuator is plugged into the system, the enlarged state reads

$$\chi^{(m+1)}(\bar{k}) = \begin{bmatrix} \chi(\bar{k}) \\ \tilde{u}_{m+1}(\bar{k}) \end{bmatrix},$$

where $\tilde{u}_{m+1}(\bar{k}) = \zeta_{m+1}(\tau\bar{k})$, $\zeta_{m+1}(\tau\bar{k})$ being the internal state of such an actuator. If $\tilde{u}_{m+1}(\bar{k}) = 0$, then $\chi^{(m+1)}(\bar{k})$ belongs to the feasibility region of problem (3.24) for the new actuation configuration. As a matter of fact, problem (3.24) for $k < \bar{k}$ can be seen as a particular instance of the same problem for the enlarged system, under the additional constraint

$$\tilde{u}_{m+1}(k+j) = 0, \forall j = 0,\dots,N_c-1. \tag{3.31}$$

We hence make the following assumption.

Assumption 4 *The value of the control action afforded by the new actuator in correspondence to the addition time instant is null, namely* $\tilde{u}_{m+1}(\bar{k}) = 0$. ♦

Finally, considering the stability requirement, we let $V^{(m+1)}_{\text{constr}}\big(\chi^{(m+1)}(k),N_c,N_p\big)$ be the optimal value of the optimization problem for the enlarged system with constraint (3.31) and $V^{(m+1)}\big(\chi^{(m+1)}(k),N_c,N_p\big)$ be the same quantity for the corresponding unconstrained problem. It thus turns out that

$$V\big(\chi(k),N_c,N_p\big)=V^{(m+1)}_{\text{constr}}\big(\chi^{(m+1)}(k),N_c,N_p\big).$$

Moreover, since at time $\bar{k}$ it holds that

$$V^{(m+1)}\big(\chi^{(m+1)}(\bar{k}),N_c,N_p\big)\leq V^{(m+1)}_{\text{constr}}\big(\chi^{(m+1)}(\bar{k}),N_c,N_p\big)$$

then by inequality (3.26c) applied at time $\bar{k}-1$, one obtains

$$V^{(m+1)}\big(\chi^{(m+1)}(\bar{k}),N_c,N_p\big)-V\big(\chi(\bar{k}-1),N_c,N_p\big)\leq-(1-\gamma^2\cdot\bar{\gamma}_{\text{d}}^2)\|z(\bar{k}-1)\|^2_{Q_z}. \tag{3.32}$$

This inequality stands for the counterpart of (3.26c) in the instant of the actuator addition. Consequently, since all the above arguments iteratively apply to any of such occurrences, one can conclude that stability is preserved irrespective of any actuator plugging event.

Actuator replacement

Differently from the previous case, when the *i*-th actuator is replaced with a new one, dimensionality of system (3.10) is not an issue, whilst matrices F in (3.12b) and G in (3.12c), and hence $\tilde{A}$ and $\tilde{B}_1$ in (3.11), change in view of the value f_i^{new} characterizing the new device (namely matrices A in (3.6) and B_1 in (3.9c) remain unaltered). Let us hence discuss the reconfigurability problem of the high level controller.

The auxiliary control law can be maintained. However, its robust stabilization properties given in Proposition 1 still hold, provided that the following conditions are satisfied: first, letting $\gamma_{\text{d}}^{\text{new}}(i)$ be the gain defined in (3.18) for the new actuator, inequality $\gamma\cdot\gamma_{\text{d}}^{\text{new}}(i)<1$ is valid; secondly, the left-hand-side of the Riccati inequality (3.20), evaluated with the same P but with the matrices $\tilde{A}$ and $\tilde{B}_1$ replaced by those associated with the new low level configuration, is negative definite. Notice that, by continuity arguments, the latter property is ensured by the choice of a new actuator such that $|f_i^{\text{new}}-f_i|<\varepsilon$, for a sufficiently small $\varepsilon>0$, which also allows for fulfilling Assumption 2.2.

In order to save the feasibility of problem (3.24) in the replacement time instant $\bar{k}$, the following assumption is considered.

Assumption 5 *The value of the control action afforded by the new actuator at time $\bar{k}$ coincides with the one the old actuator was giving in such an instant, namely $\tilde{u}_i^{\text{new}}(\bar{k})=\tilde{u}_i(\bar{k})$.* ♦

If all the above conditions hold, the result of Theorem 1 is guaranteed without any further manipulations, provided that the weighting matrix Q_i^{new} coincides with the one applied to the replaced actuator. However, it is worth noting that the optimal value $V^{\text{new}}\big(\chi(\bar{k})\big)$ of problem (3.24) in the new configuration may be larger than the optimal value $V\big(\chi(\bar{k})\big)$ in the old configuration. Therefore, the counterpart of relation (3.32) for the replacement case is not ensured. Nevertheless, stability of the overall process can be preserved as long as a sufficiently large time interval between two consecutive replacement events is held on. This corresponds to the well-known concept of dwell-time in switching control [28], thereby a sufficient decrease of the optimal value function is ensured and, as a consequence, its possible increases in the replacement instants are counteracted.

3.4 Conclusions

In this chapter, a two level hierarchical control problem has been addressed. At the high level a robust MPC regulator has been designed in order to compute the ideal control actions needed by the plant to be controlled. A number of already controlled actuators placed at the low level are in charge of driving the plant by tracking such control actions. A convergence result for the overall control system has been derived by resorting to the small gain approach. The algorithm proposed can also be used in a Plug and Play framework as a new actuator is added/replaced into the overall control system.

Chapter 4

Conclusions

In the HD-MPC project, Work Package 2 was intended to be a preliminary step aimed at defining and establishing appropriate control architectures for the development of distributed and hierarchical control and estimation methods. Its main outcomes had to be the basis for the other work packages. For these reasons, WP2 was organized into four main tasks, listed in the following together with the main results achieved.

Task 2.1: Survey
The main approaches to hierarchical and distributed control and estimation proposed so far in the technical literature have been critically examined (see Deliverable D2.1). The survey has been useful to define the different control problems which can be tackled with hierarchical structures (cascade control systems, singularly perturbed systems, Real Time Optimization based on models with different levels of abstraction). Moreover, the approaches proposed in the literature for the design of distributed control systems have been classified according to their goals (e.g., independent subsystems, interacting subsystems) and their nature (e.g., iterative, cooperating).

Task 2.2: Definition of the control architecture
A very general architecture for hierarchical control has been defined and described in Deliverable D2.2. It is based on a three layer structure, where at each layer a different time scale is considered, which allows one to cope with the different cases devised in task 2.1. Moreover, a paradigmatic hierarchical control algorithm has been defined and some inter-layer communication protocols have been proposed. As for the definition of the architecture for distributed control, some preliminary results based on classical partitioning approaches have been proposed in Deliverable D2.1, while some other methods have been surveyed in Chapter 1 of this deliverable (D2.3).

Task 2.3: Extension of the control architecture (to adapt the architecture in response to changes)
This topic has been studied in the final part of the Work Package and has been described in this deliverable (D2.3, Chapter 3). It has been shown how the MPC approach allows one to partially reconfigure a hierarchical control system in front of actuators' addition or replacement (plug and play control) without the need to redesign the overall control structure and maintaining the original convergence properties. This result can be viewed as the basis of further extensions in other WPs, such as WP3 or WP5, in order to consider more general cases of control reconfiguration due to a time varying exchange of information among local subsystems in a distributed control environment.

Task 2.4: Multi-level models (models consistent with the hierarchical levels, multi-resolution models, reduction and aggregation methods)
The derivation of multi-level models, with different kinds of spatial and temporal resolution, has been considered in Deliverable D2.1 and in this report (Chapter 1). Moreover, multi-level models have been used to derive hierarchical control systems in a couple of significant examples where a global approach is not suitable due to the complexity of the underlying optimization problem (see Chapter 2 of this report). It is believed that new partitioning criteria for distributed control and estimation will naturally stem from the synthesis methods studied in other work packages of the project.

Bibliography

[1] M. Agudelo. The application of proper ortoghonal decomposition to the control of tubular reactors. *PhD thesis, Katholieke Universiteit Leuven*, 2009.

[2] P. Astrid. Reduction of process simulation models: a proper orthogonal decomposition approach. *PhD thesis, Technishche Universiteit Eindhoven*, 2004.

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

[4] C. Audet and J. E. Dennis Jr. Analysis of generalized pattern searches. *SIAM Journal on Optimization*, 13(3):889–903, 2007.

[5] L. D. Baskar, B. De Schutter, and J. Hellendoorn. Hierarchical traffic control and management with intelligent vehicles. In *Proceedings of the 2007 IEEE Intelligent Vehicles Symposium (IV'07)*, pages 834–839, Istanbul, Turkey, June 2007.

[6] L.D. Baskar. *Traffic Management and Control in Intelligent Vehicle Highway Systems*. PhD thesis, Delft University of Technology, Delft, The Netherlands, November 2009.

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

[8] J. Bendtsen, K. Trangbaek, and J. Stoustrup. Closed-loop system identification with new sensors. In *Proceedings of the 47th IEEE Conference on Decision and Control*, Cancun, Mexico, December 2008.

[9] J. Bendtsen, K. Trangbaek, and J. Stoustrup. Plug and play process control: improving control performance through sensor addition and pre-filtering. In *Proceedings of the 17th IFAC World Congress*, Seoul, Korea, July 2008.

[10] R. Bishop. *Intelligent Vehicles Technology and Trends*. Artech House, 2005.

[11] M. Brackstone and M. McDonald. Car-following: A historical review. *Transportation Research Part F*, 2(4):181–196, 1999.

[12] M. Broucke and P. Varaiya. The automated highway system: A transportation technology for the 21st century. *Control Engineering Practice*, 5(11):1583–1590, November 1997.

[13] L. S. Comte. New systems: new behaviour? *Transportation Research Part F*, 3(2):95–111, May 2000.

[14] C. F. Daganzo. *Fundamentals of Transportation and Traffic Operations*. Pergamon Press, 1997.

[15] L. Davis. *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York, 1991.

[16] L. C. Davis. Effect of adaptive cruise control systems on traffic flow. *Physical Review E*, 69:1–8, 2004.

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

[18] M.T. Dokucu, M.J. Park, and F.J. Doyle III. Multi-rate model predictive control of particle size distribution in a semibatch emulsion copolymerization reactor. *Journal of Process Control*, 18:105–120, 2008.

[19] R. W. Eglese. Simulated annealing: A tool for operations research. *European Journal of Operational Research*, 46:271–281, 1990.

[20] 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.

[21] R. E. Fenton. IVHS/AHS: Driving into the future. *IEEE Control Systems Magazine*, 14(6):13–20, December 1994.

[22] 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.

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

[24] D. Gazis, R. Herman, and R. Rothery. Nonlinear follow the leader models of traffic flow. *Operations Research*, 9(4):545–567, June 1961.

[25] F. L. Hall and K. Agyemang-Duah. Freeway capacity drop and the definition of capacity. *Transportation Research Record*, (1320):91–98, 1991.

[26] 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.

[27] J. Hennet. A bimodal scheme for multi-stage production and inventory control. *Automatica*, 39:793–805, 2003.

[28] J.P. Hespanha and A.S. Morse. Stability of switched systems with average dwell-time. In *Proceedings of the 38th Conference on Decision and Control*, pages 2655–2660, Phoenix, Arizona, USA, December 1999.

[29] L. Huisman. Control of glass melting processes based on reduced cfd models. *PhD thesis, Technishche Universiteit Eindhoven*, 2005.

[30] R. K. Jurgen. Smart cars and highways go global. *IEEE Spectrum*, 28(5):26–36, May 1991.

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

[32] T. Knudsen, K. Trangbaek, and C.S. Kallesøe. Plug and play process control applied to a district heating system. In *Proceedings of the 17th IFAC World Congress*, Seoul, Korea, July 2008.

[33] E. Kometani and T. Sasaki. Dynamic behaviour of traffic with a nonlinear spacing speed relationship. In *Proceedings of the Symposium for Theory Traffic Flow*, pages 105–109, Research Laboratories, General Motors, New York, 1959.

[34] K. Laabidi, F. Bouani, and M. Ksouri. Multi-criteria optimization in nonlinear predictive control. *Mathematics and Computers in Simulation*, 76:363–374, 2008.

[35] S. Lall, J.E. Marsden, and S. Glavaski. A subspace approach to balanced truncation for model reduction of nonlinear control systems. *Int. J. Robust Nonlinear Control*, 12:519–535, 2002.

[36] M. Lazar, W.P.M.H. Heemels, S. Weiland, and A. Bemporad. Stabilizing model predictive control of hybrid systems. *IEEE Transactions on Automatic Control*, 51(11):1813–1818, 2006.

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

[38] K. Li and P. Ioannou. Modeling of traffic flow of automated vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 5(2):99–113, June 2004.

[39] J. Linderoth and T. Ralphs. Noncommercial software for mixed-integer linear programming. *Optimization Online*, January 2005.

[40] L. Magni, G. De Nicolao, R. Scattolini, and F. Allgöwer. Robust model predictive control of nonlinear discrete-time systems. *International Journal of Robust and Nonlinear Control*, 13:229–246, 2003.

[41] A. D. May. *Traffic Flow Fundamentals*. Prentice-Hall, Englewood Cliffs, New Jersey, 1990.

[42] R. M. Michaels. Perceptual factors in car following. In *Proceedings of the 2nd International Symposium for Theory Road Traffic Flow*, pages 44–59, Paris, France, 1963.

[43] K. Nagel. Particle hopping models and traffic flow theory. *Physical Review E*, 53:4655–4672, 1996.

[44] P. M. Pardalos and M. G. C. Resende. *Handbook of Applied Optimization*. Oxford University Press, Oxford, UK, 2002.

[45] S. Shladover, C.A. Desoer, J.K. Hedrick, M. Tomizuka, J. Walrand, W.B. Zhang, D.H. McMahon, H. Peng, S. Sheikholeslam, and N. McKeown. Automatic vehicle control developments in the PATH program. *IEEE Transactions on Vehicle Technology*, 40(1):114–130, February 1991.

[46] G. Stephanopoulos, K. Karsligil, and M. Dyer. Multi-scale aspects in model-predictive control. *Journal of Process Control*, 10:275–282, 2000.

[47] J. M. Sussman. Intelligent vehicle highway systems: Challenge for the future. *IEEE Micro*, 1(14-18):101–104, June 1993.

[48] A.N. Tarău. *Model-Based Control for Postal Automation and Baggage Handling*. PhD thesis, Delft University of Technology, Delft, The Netherlands, January 2010.

[49] 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.

[50] A.N. Tarău, B. De Schutter, and J. Hellendoorn. Route choice control of automated baggage handling systems. In *Proceedings of the 88th Annual Meeting of the Transportation Research Board*, Washington DC, USA, January 2009. Paper 09-0432.

[51] E. Tatara, B. Inanc, T. Fouad, and Ç Ali. Agent-based control of autocatalytic replicators in networks of reactors. *Computers and Chemical Engineering*, 29:807–815, 2005.

[52] P. Tatjewski. Advanced control and on-line process optimization in multilayer structures. *Annual Reviews in Control*, 32:71–85, 2008.

[53] P. Varaiya. Smart cars on smart roads. *IEEE Transactions on Automatic Control*, 38(2):195–207, February 1993.

[54] F.D. Vargas-Villamil and D.E. Rivera. Multilayer optimization and scheduling using model predictive control: application to reentrant semiconductor manufacturing lines. *Computers and Chemical Engineering*, 24:2009–2021, 2000.

[55] D. De Vito, B. Picasso, and R. Scattolini. On the design of reconfigurable two-layer hierarchical control systems with mpc. In *IEEE American Control Conference*, 2010.

[56] F.Y. Wang and I.T. Cameron. A multi-form modelling approach to the dynamics and control of drum granulation processes. *Power Technology*, 179:2–11, 2007.

[57] Y. Wang, D. Zhou, and F. Gao. Iterative learning model predictive control for multi-phase batch processes. *Journal of Process Control*, 18:543–557, 2008.

[58] S. Weiland. Balancing and hankel norm approximation of dynamical systems. *Course on model reduction*, 2006.

[59] P.A. Wisnewski and F.J. Doyle III. Control structure selection and model predictive control of the weyerhaeuser digester problem. *Journal of Process Control*, 8:487–495, 1998.

[60] W. Wu, J.P. Xua, and K. Hwang. Multi-loop nonlinear predictive control scheme for a simplistic hybrid energy system. *International journal of hydrogen energy*, 34:3953–3964, 2009.

[61] M. Xu, S. Li, and W. Caib. Cascade generalized predictive control strategy for boiler drum level. *ISA Transactions*, 44:399–411, 2005.

[62] D.W. Yu and D.L. Yu. Multi-rate model predictive control of a chemical reactor based on three neural models. *Biochemical Engineering Journal*, 37:86–97, 2007.