Direct Torque Control for Induction Motor Drives: A Model Predictive Control Approach based on Feasibility

Tobias Geyer and Georgios Papafotiou

Automatic Control Laboratory, Swiss Federal Institute of Technology (ETH)
CH-8092 Zurich, Switzerland, `geyer,papafotiou@control.ee.ethz.ch`

Abstract. In this paper, we present a new approach to the Direct Torque Control (DTC) problem of three-phase induction motor drives. This approach is based on Model Predictive Control (MPC) exploiting the specific structure of the DTC problem and using a systematic design procedure. Specifically, by observing that the DTC objectives, which require the controlled variables to remain within certain bounds, are related to feasibility rather than optimality, and by using a blocking control inputs regime for the whole prediction horizon we derive a low complexity controller. The derived controller is an explicit state-feedback control law that can be implemented as a look-up table. Even though the controller is derived here for a DTC drive featuring a two-level inverter, the control scheme can be extended to also tackle three-level inverters. Simulation results demonstrate that the proposed controller leads to performance improvements despite its simple structure.

1 Introduction

Enabled by significant technological developments in the area of power electronics, variable speed induction motor drives have evolved to a state of the art technology within the last decades. These systems, in which DC-AC inverters are used to drive induction motors as variable frequency three-phase voltage or current sources, are used in a wide spectrum of industrial applications. One of the methods for controlling the induction motor's torque and speed is Direct Torque Control (DTC), which was first introduced in 1985 by Takahashi and Noguchi [13] and is nowadays a industrial standard for induction motor drives [14, 11].

The basic characteristic of DTC is that the positions of the inverter switches are directly determined rather than indirectly, thus refraining from using a modulation technique like Pulse Width (PWM) or Space Vector (SVM) modulation. In the generic scheme, the control objective is to keep the motor's torque and the amplitude of the stator flux within pre-specified bounds. The inverter is triggered by hysteresis controllers to switch whenever these bounds are violated. The choice of the new switch positions is made using a pre-designed look-up table that has been derived using geometric insight in the problem and additional heuristics.

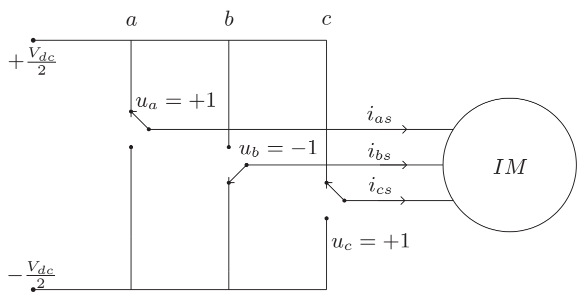
The main reason that makes the design of the switching table difficult is the fact that the DTC drive constitutes a hybrid system, i.e. a system incorporating both continuous and discrete dynamics - in particular discrete-valued manipulated variables. Additionally, constraints on states, inputs and outputs are present imposing further complications on the controller design, since the underlying mathematical problems are intrinsically complex and hard to solve.

Recently, we have proposed in [9, 10] a systematic procedure for the design of the DTC switching table by reformulating the control problem as a *Model Predictive Control* (MPC) [8] problem for a two- and a three-level inverter. Modelling the DTC drive as a hybrid system, introducing integer variables for the inverter switch positions that represent the manipulated variables of the control problem and expressing the control objectives in a cost function led to a constrained finite time optimal control problem. By solving the underlying optimization problem on-line and comparing the results with the behavior of ABB's ACS6000 drive [1] featuring a three-level inverter, we have demonstrated a potential performance improvement in the range of 20 %. Subsequently, moving towards the practical implementation of the method, we have pre-computed off-line the optimal control problem for all feasible states and thus derived the explicit state-feedback control law. The latter was done for a DTC drive featuring a two-level inverter and for a specific operating point.

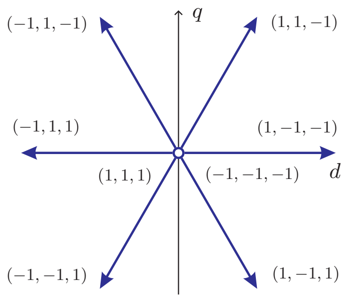
Nevertheless, the complexity of the derived state-feedback controller prohibits the practical implementation on the currently employed controller hardware. On the other hand, two observations suggest the existence of a low complexity controller resulting from a systematic design procedure. Firstly, albeit their very simple controller structure, the existing DTC schemes have proven to yield a satisfactory control performance. Secondly, the post analysis of the derived state-feedback control law reveals a simple and robust pattern in the solution to the optimal control problem.

These observations have motivated the control scheme presented in this paper which is based on the following fundamental property of DTC. The control objectives only weakly relate to optimality but rather to feasibility, in the sense that the main objective is to find a control input that keeps the controlled variables within their bounds, i.e. a control input that is feasible. The second, weaker objective is to select among the set of feasible control inputs the one that minimizes the average switching frequency. The latter can be approximated by the number of switch transitions over the (short) horizon.

We therefore propose an MPC scheme based on feasibility with a prediction horizon N and an internal model of the DTC drive for the predictions. We propose to switch only at the current time-step and to disregard switching within the prediction horizon, which is equivalent to a move blocking strategy. This greatly reduces the number of control input sequences from 8^N to 8 and allows us to evaluate a small number of input sequences by moving forward in time. For each input sequence, we determine the number of steps the controlled variables are kept within their bounds, i.e. remain feasible. Next we define the number of switch transitions divided by the number of predicted time-steps an input



(a) The equivalent representation of a three-phase two-level inverter driving an induction motor



(b) The voltage vectors on the dq plane with switch positions

Fig. 1. Physical setup and voltage vectors

remains feasible as a cost function emulating the switching frequency. In a last step, the control input is chosen that minimizes the cost function. We refer to this concept as the *Feasibility Approach*. The simplicity of the control methodology (with the only design parameter N) translates into a state-feedback control law with a complexity that is of an order of magnitude lower than the one of its counterpart obtained through solving the optimal control problem [9].

The paper is organized as follows. Starting with the derivation of a low complexity piecewise affine model for the DTC drive in Section 2, we pose in Section 3 the control objectives. In Section 4, we first present the Feasibility Approach as a control scheme that is evaluated on-line, and subsequently, we show how the control problem can be pre-solved off-line and translated into a state-feedback control law. Simulation results for the case of a two-level inverter are shown in Section 5, while Section 6 summarizes the results and discusses the extendability of the control approach to DTC drives featuring three-level inverters.

Due to the page limitation the paper had to be shortened by a few pages (mostly Section 2). The full paper is available as technical report [4].

2 Modelling

2.1 Physical Setup

For the modelling of the DTC drive, all variables are transformed from the three-phase system (abc) to an orthogonal dq0 reference frame with a direct (d), a quadrature (q) and a zero (0) axis, that can be either stationary or rotating [6]. For the needs of this paper, the transformation of a vector $\xi_{abc} = [\xi_a \ \xi_b \ \xi_c]^T$ from the three-phase system to the vector $\xi_{dq0} = [\xi_d \ \xi_q \ \xi_0]^T$ in the dq0 frame is carried out through $\xi_{dq0} = P(\varphi)\xi_{abc}$, where φ is the angle between the a-axis of the three-phase system and the d-axis of the reference frame, and $P(\varphi)$ is the Park transformation [6].

An equivalent representation of a three-phase two-level inverter driving an induction motor is shown in Fig. 1(a). At each phase, the inverter can produce two different voltages $-\frac{V_{dc}}{2}, \frac{V_{dc}}{2}$, where V_{dc} denotes the voltage of the dc-link. The switch positions of the inverter can therefore be fully described using the three integer variables u_a, u_b, $u_c \in \{-1, 1\}$, where each variable corresponds to one phase of the inverter, and the values $-1, 1$ correspond to the phase potentials $-\frac{V_{dc}}{2}, \frac{V_{dc}}{2}$, respectively.

There are $2^3 = 8$ different vectors of the form $u_{abc} = [u_a \; u_b \; u_c]^T$. Using the Park transformation these vectors can be transformed into the dq0 frame resulting in vectors of the form $u_{dq0} = [u_d \; u_q \; u_0]^T$. The latter are shown in Fig. 1(b), where they are mapped into the (two-dimensional) dq plane. Even though they are commonly referred to as voltage vectors, this term describes the switch positions rather than the actual voltages applied to the machine terminals.

The dynamics of the squirrel-cage rotor induction motor are commonly modelled in a dq0 reference frame that can be either stationary or rotating. The standard modelling approach, which can be found in detail in [6], yields a 5-dimensional nonlinear state-space model, that uses as state variables the d- and q-components of the stator and rotor flux linkages per second ψ_{ds}, ψ_{qs}, ψ_{dr} and ψ_{qr}, respectively, and the rotor's rotational speed ω_r. The 0-axis components are neglected, since they do not contribute to the electromagnetic torque and are decoupled from the dynamics in the d- and q-axis. The model parameters are the stator and rotor resistances r_s and r_r, the stator, rotor and mutual inductive reactances x_{ls}, x_{lr} and x_m, respectively, the inertia constant H expressed in seconds, and the mechanical load torque T_ℓ.

In this standard dynamical model of the induction motor, the saturation of the machine's magnetic material, the changes of the rotor resistance due to the skin effect and the temperature changes of the stator resistance are ignored. A more elaborate presentation of the induction motor's modelling procedure is out of the scope of this paper. For details, the reader is referred to [6].

2.2 Low Complexity Modelling

In [9, 10], we have derived a low-complexity model of the DTC drive taking into account that the stator flux dynamics are significantly faster than the dynamics of the rotor flux and the rotational speed, and that the length of the stator flux vector and the electromagnetic torque are invariant under a rotation of the flux vectors. This model has the state vector

$$x(k) = \left[\psi_{ds}^{\vartheta}(k) \; \psi_{qs}^{\vartheta}(k) \; \cos(\varphi(k))\right]^T , \tag{1}$$

where $\psi_{ds}^{\vartheta}(k)$ and $\psi_{qs}^{\vartheta}(k)$ denote the d- and q-component of the rotated and mapped stator flux vector, and $\varphi(k)$ captures the position of the rotating reference frame with $\varphi(k+1) = \varphi(k) + \omega_r T_s$. The output vector

$$y(k) = \left[T_e(k) \; \Psi_s^2(k)\right]^T \tag{2}$$

comprises the electromagnetic torque and the squared length of the stator flux vector, and the input vector is composed of the integer variables u_a, u_b and u_c

$$u(k) = u_{abc}(k) = \left[u_a(k) \; u_b(k) \; u_c(k)\right]^T \in \{-1, 1\}^3 . \tag{3}$$

For a summary of the low-complexity modelling, the reader is referred to [4], whereas the complete modelling can be found in [9].

2.3 Piecewise Affine Model

In a subsequent step, we have computed in [9] a piecewise affine (PWA) model for a DTC drive featuring a two-level inverter. PWA models [12] are defined by partitioning the state-space into polyhedra and associating with each polyhedron an affine state-update and output function

$$x(k+1) = f_{j(k)}(x(k), u(k)) \tag{4a}$$

$$y(k) = g_{j(k)}(x(k)) \tag{4b}$$

$$\text{with } j(k) \text{ such that } \begin{bmatrix} x(k) \\ u(k) \end{bmatrix} \in \mathcal{P}_{j(k)}, \tag{4c}$$

where $x(k)$, $u(k)$, $y(k)$ denote at time k the real and binary states, inputs and outputs, respectively, the polyhedra $\mathcal{P}_{j(k)}$ define a set of polyhedra $\{\mathcal{P}_j\}_{j\in\mathcal{J}}$ on the state-input space, and the real time-invariant functions $f_{j(k)}$ and $g_{j(k)}$ are affine in the states and inputs, with $j(k) \in \mathcal{J}$, $\mathcal{J}$ finite. For simplicity, we will later drop the index $j(k)$ and (4c), and use $x(k+1) = f(x(k), u(k))$ and $y(k) = g(x(k))$ to denote the PWA system (4). Note that the PWA system (4) has no throughput, i.e. $y(k)$ is independent of $u(k)$.

To derive such a PWA model, all nonlinearities need to be replaced by PWA approximation over a bounded set of (feasible) states $\mathcal{X}^0$. The set $\mathcal{X}^0$ can be easily determined by translating the output hysteresis bounds imposed by the control objectives into constraints on the state-space. Introducing the lower and upper bounds on the electromagnetic torque $T_{e,min}$ and $T_{e,max}$, respectively, and noting that in the low-complexity model the torque is a linear expression of the second state, the torque bounds can be directly translated into linear bounds on $x_2(k)$

$$\frac{D}{x_m \psi_{dr}^{\vartheta}} T_{e,min} \le \; x_2(k) \; \le \frac{D}{x_m \psi_{dr}^{\vartheta}} T_{e,max} \,, \tag{5}$$

where ψ_{dr}^{ϑ} is equal to the length of the rotor flux, which is treated as a parameter in the low-complexity model. Similarly for the stator flux, its lower and upper bounds $\Psi_{s,min}^2$ and $\Psi_{s,max}^2$ turn into the quadratic state constraint

$$\Psi_{s,min}^2 \le \; x_1^2(k) + x_2^2(k) \; \le \Psi_{s,max}^2 . \tag{6}$$

To account for measurement noise and small disturbances causing the torque or the stator flux to slightly violate the imposed bounds, we relax (5) and (6) by 20 % of the corresponding bound width.

The bounds on the third state are derived from the angle $\varphi(k)$. To ensure that the model remains feasible for at least N time-steps when starting with a $\varphi(k)$ close to $\frac{\pi}{3}$, the bounds on $\varphi(k)$ are set to $0 \le \varphi(k) \le \frac{\pi}{3} + N\omega_r T_s$, which translate into the following bounds on $x_3(k)$

$$\cos(\frac{\pi}{3} + N\omega_r T_s) \le x_3(k) \le 1 \,. \tag{7}$$

Summing up, the constraints (5), (6) and (7) define the set of states $\mathcal{X}^0$ for which the PWA model is to be defined. Thus, the nonlinearities of the DTC drive need to be approximated for $x \in \mathcal{X}^0$ as shown in [9, 10].

Starting from a model description in the HYbrid Systems DEscription Language HYSDEL [15], and fixing the operating point, namely the parameters ω_r and ψ_{dr}^{ϑ}, the model can be transformed into PWA form with the mode enumeration algorithm [5]. This procedure yields a PWA model defined on a polyhedral partition with 48 polyhedra in the six-dimensional state-input space.

3 Control Problem

The most prominent control objective concerning the induction motor is to keep the electromechanical torque within bounds around its reference. In order to avoid the saturation or demagnetization of the motor, the amplitude of the stator flux has to be kept between certain pre-specified bounds around the reference which are in general time-invariant. The control objective concerning the inverter is to minimize the average switching frequency.

4 Feasibility Approach

Traditionally, based on the imposed bounds, the next voltage vector to be applied to the induction motor is selected by evaluating a look-up table every $T_s = 25\,\mu$s. The goal of this paper is to replace the look-up table by a new DTC scheme that is based on a systematic design procedure. This controller needs to address the above formulated objectives, i.e. to minimize the average switching frequency while keeping the controlled variables (torque and length of the stator flux) within the given bounds.

Similar to [9, 10], this controller is based on predictive control with a receding horizon policy. Minimizing the average switching frequency leads to a prediction horizon with an infinite number of steps. As such a problem in the context of hybrid systems is computationally not tractable, we need to approximate this objective. In [9, 10], we have done this by restricting the prediction horizon N to a small number of steps (three or four) and by formulating an objective function that postpones switching and penalizes the violation of the bounds using soft constraints. In particular, we have allowed for switch transitions within the prediction interval. Dynamic programming [3] allowed us to compute off-line the explicit state-feedback control law for the whole state-space.

4.1 On-line Computation of the Control Input

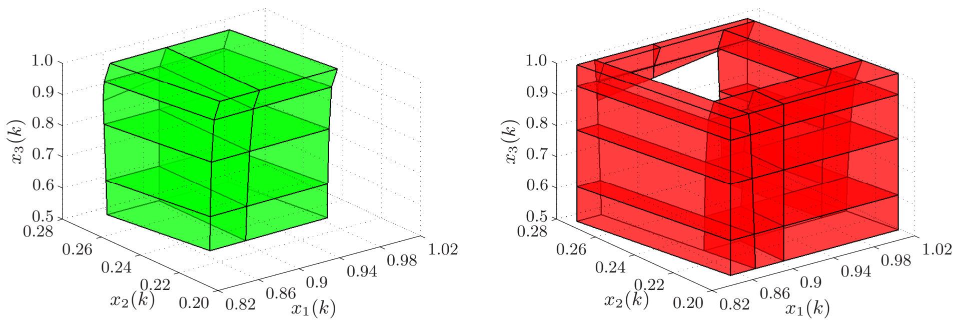
On the other hand, the underlying optimization problem of the above stated control problem is not so much based on optimality but rather on feasibility, meaning that the controlled variables have to be kept within their bounds, i.e. feasible. This insight greatly simplifies the control problem. Furthermore, we propose to switch only at the current time-step k and to disregard switching within the prediction horizon, which is equivalent to a move blocking strategy. This greatly reduces the number of control input sequences from 8^N to 8 and allows us to evaluate a small number of control sequences by moving forward in time. As a result, dynamic programming moving backwards in time becomes obsolete.

More formally, let $u(k-1)$ denote the last voltage vector. If $u(k-1)$ is also feasible at time-instant k, i.e. all controlled variables are predicted to lie within their bounds at time-instant $k+1$, a reasonable choice is to apply it again, i.e. $u(k) = u(k-1)$. If not, however, the controller must choose another voltage vector. For each of the remaining seven voltage vectors, one can easily compute through open-loop predictions the number of time-steps this voltage vector would keep the controlled variables within their bounds. This step reduces the optimal control problem to a feasibility problem. The voltage vector is chosen that minimizes the average switching frequency over the prediction interval, i.e. the number of switch transitions over the number of time-steps, thus re-introducing the notion of optimality. This control concept, to which we refer as the Feasibility Approach, is summarized in Algorithm 1, where f and g refer to the PWA model (4). An output vector $y(k)$ is said to be feasible, if the corresponding bounds are met, and $U = \{-1, 1\}^3$ denotes the set of available voltage vectors.

Algorithm 1

```
function $u(k) = \mathsf{Algo1}\,(\,x(k),\, u(k-1)\,)$
    $x(k+1) = f(x(k), u(k-1))$
    if $y(k+1) = g(x(k+1))$ feasible
        $u(k) = u(k-1)$
    else
        for all $u(k) \in U \setminus u(k-1)$
            $n_u = -1$
            repeat
                $n_u = n_u + 1$
                $x(k+n_u+1) = f(x(k+n_u), u(k))$
            until $\big(y(k+n_u+1) = g(x(k+n_u+1))$ infeasible$\big)$ or $\big(n_u = N\big)$
        endfor
        $u(k) = \arg\min_{u(k)} \frac{||u(k)-u(k-1)||}{n_u}$
    endif
```

Compared to MPC, this control policy is by definition significantly simpler, as only eight control sequences (or control strategies) need to be compared with each other. Unlike in MPC, switch transitions within the prediction interval are not considered, and can only be performed at the current time-instant k. Furthermore, the length of the prediction horizon is time-varying, ranging from one



(a) The core (b) The ring

Fig. 2. Core and ring for $u(k-1) = [1 \ \ -1 \ \ -1]$

step to 10 or even 20 steps. As the next section will show, an explicit form of the proposed controller can be computed easily. Even more important, the explicit form has a low complexity but maintains or improves the control performance with respect to MPC.

4.2 Off-line Computation of the State-feedback Control Law

We restrict the computation of the explicit state-feedback control law to the set of states $\mathcal{X}^0$, which we have obtained by relaxing the bounds on the torque and the flux by 20 %. Furthermore, we fix the operating point, namely the rotor speed ω_r and the length of the rotor flux ψ_{dr}^{ϑ}, and set the lower and upper bounds on the outputs (torque and stator flux). Next, we derive the PWA model defined on $\mathcal{X}^0$. Rewriting (5) and (6), let $\mathcal{C}$ denote the set of states whose corresponding outputs are feasible

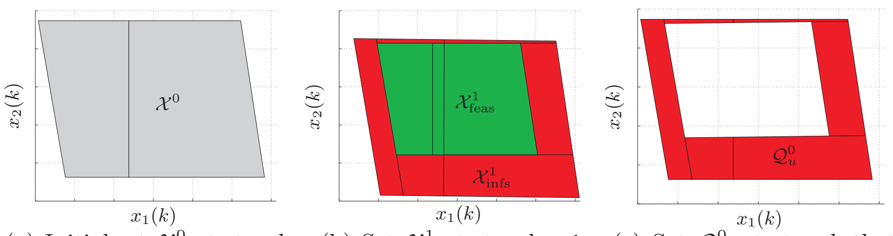
$$\mathcal{C} = \{x \in \mathcal{X}^0 \mid \begin{bmatrix} T_{e,min} \\ \Psi_{s,min}^2 \end{bmatrix} \leq g(x) \leq \begin{bmatrix} T_{e,max} \\ \Psi_{s,max}^2 \end{bmatrix}, \tag{8}$$

where we have replaced the quadratic expression in (6) by the PWA approximation for the stator flux.

Before presenting the computation of the state-feedback control law in three stages, we introduce the following notation. Let n denote the time-step within the prediction horizon N, $\mathcal{X}_{\text{feas}}^n$ the set of states at time-step $k+n$ corresponding to feasible outputs $y(k+\ell)$ for all $\ell \in \{1,\dots,n\}$, $\mathcal{X}_{\text{infs}}^n$ the set of states at time-step $k+n$ with feasible outputs $y(k+\ell)$ for all $\ell \in \{1,\dots,n-1\}$, but infeasible outputs $y(k+n)$, and $\mathcal{Q}_u^n$ the set of states at time-step k that keep the outputs for n time-steps feasible when applying the voltage vector u.

Stage I First, we determine the set of states $x(k) \in \mathcal{X}^0$ for which the controlled variables are feasible at time-step $k+1$ when applying $u(k) = u(k-1)$. We denote this set of states as the *core*

$$\mathcal{Q}_u^c = \{x \in \mathcal{X}^0 \mid f(x,u) \in \mathcal{C}\}, \tag{9}$$



(a) Initial set $\mathcal{X}^0$ at step k (b) Set $\mathcal{X}^1$ at step $k+1$ (c) Set $\mathcal{Q}^0_u$ at step k that yields infeasible outputs at $k+1$

Fig. 3. First step of Algorithm 2 in the x_1x_2 plane for $u(k) = [1 \;\; -1 \;\; -1]$

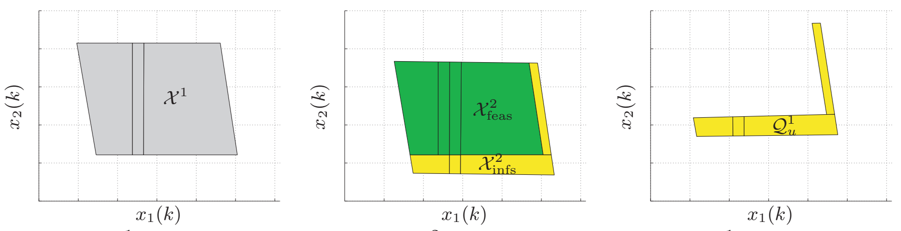
and its complement in $\mathcal{X}^0$ as the *ring*

$$\mathcal{Q}^r_u = \mathcal{X}^0 \setminus \mathcal{Q}^c_u \, . \tag{10}$$

Example 1 To visualize the algorithm, consider as an example a two-level inverter driving an induction machine with the rated voltage 3.3 kV and the rated real power 1.587 MW. All parameters can be found in [9] in Tables 3 and 4. The operating point is given by the rotor speed $\omega_r = 0.8$ p.u., the load torque $T_\ell = 0.8$ p.u., the torque bounds $T_{e,min} = 0.72$ p.u. and $T_{e,max} = 0.88$ p.u., and the flux bounds $\Psi^2_{s,min} = 0.82$ p.u. and $\Psi^2_{s,max} = 1.04$ p.u.. After deriving the PWA model on $\mathcal{X}^0$ (enlarged by 20 % as in Section 2.3), and determining the set $\mathcal{C}$, the core and the ring can be easily computed as shown in Fig. 2 for the voltage vector $u(k-1) = [1 \;\; -1 \;\; -1]$. This operation takes on a Pentium IV roughly 1 s. ∎

Stage II For each new voltage vector $u(k) \in U \setminus u(k-1)$, the following procedure is performed for the initial set[1] $\mathcal{X}^0$. Initially, we set $n = 0$. Next, we map the polyhedra $\mathcal{X}^n$ from time-step $k+n$ to $k+n+1$ yielding $\mathcal{X}^{n+1}$. The states corresponding to infeasible outputs form the set $\mathcal{X}^{n+1}_{\text{infs}}$. Consequently, we map $\mathcal{X}^{n+1}_{\text{infs}}$ back to the time-step k and associate with them the number of time-steps n. We denote these polyhedra by $\mathcal{Q}^n_u$, where u corresponds to the chosen voltage vector $u(k)$, and n denotes the number of time-steps this voltage vector $u(k)$ can be applied to the set of states before any of the outputs violates a bound. If there remain any feasible states, we move one time-step forward in the future by increasing n by one and repeat the above procedure.

[1] Conceptually, this stage of the algorithm should be initialized with the ring $\mathcal{Q}^r_u$ rather than $\mathcal{X}^0$. Let us note though that since the facets of the initial set are mapped forward and backward in time, in the worst case, the complexity of the algorithm both in terms of the computation time and the number of resulting polyhedra $\{\mathcal{Q}^n_u\}^N_{n=0}$ is exponential in the number of facets of the initial set. Therefore, as $\mathcal{X}^0$ is by definition a very simple polytopic set with only a few facets, whereas the ring is a non-convex set with possibly many facets, we initialize Algorithm 2 with $\mathcal{X}^0$ rather than the ring.



(a) Set $\mathcal{X}^1$ at step $k+1$ (b) Set $\mathcal{X}^2$ at step $k+2$ (c) Set $\mathcal{Q}^1_u$ at step k that yields feasible outputs at step $k+1$, but infeasible outputs at $k+2$

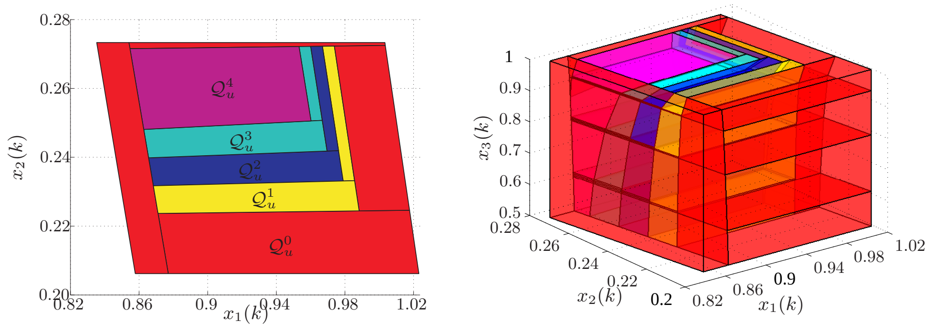
Fig. 4. Second step of Algorithm 2 in the x_1x_2 plane for $u(k) = [1 \;\; -1 \;\; -1]$

This yields for each new voltage vector a polyhedral partition of the ring $\{\mathcal{Q}^n_u\}^N_{n=0}$, where each polyhedron is associated with a unique number indicating for how many time-steps the respective voltage vector can be applied before any of the controlled variables violates a bound.

Next, the algorithm is summarized, where the two subfunctions `mapForw` and `mapBack` are affine transformations of polyhedra using the PWA model (4) for a fixed voltage vector $u(k)$. Specifically, `mapForw` yields $\mathcal{X}^{n+1} = \{f(x,u) \mid x \in \mathcal{X}^n, u = u(k)\}$, and `mapBack` yields $\mathcal{Q}^n_u = \{x \mid (f_u \circ \ldots \circ f_u)(x) \in \mathcal{X}^{n+1}_{\text{infs}}\}$, where we have set $f_u(x) = f(x,u)$ and concatenated f_u n times. Note that `mapForw` maps a set of states by one time-step forward in time, whereas `mapBack` maps a set of states by n time-steps backwards. The subscript feas (infs) refers to sets of states corresponding to feasible (infeasible) outputs.

Algorithm 2

```
function $\{\mathcal{Q}^n_u\}^N_{n=0}$ = Algo2 ( $\mathcal{C}$, $\mathcal{X}^0$, $u$, $N$ )
    $n = 0$
    while $\mathcal{X}^n \neq \emptyset$ and $n < N$
        $\mathcal{X}^{n+1}$ = mapForw ( $\mathcal{X}^n$, $u$ )
        $\mathcal{X}^{n+1}_{\text{feas}} = \mathcal{X}^{n+1} \cap \mathcal{C}$
        $\mathcal{X}^{n+1}_{\text{infs}} = \mathcal{X}^{n+1} \setminus \mathcal{X}^{n+1}_{\text{feas}}$
        $\mathcal{Q}^n_u$ = mapBack ( $\mathcal{X}^{n+1}_{\text{infs}}$, $u$ )
        $\mathcal{X}^{n+1} = \mathcal{X}^{n+1}_{\text{feas}}$
        $n = n + 1$
    endwhile
    $\mathcal{Q}^n_u$ = mapBack ( $\mathcal{X}^n$, $u$ )
```



(a) Polyhedral partition in the x_1x_2 plane for $x_3 = 0.95$

(b) Polyhedral partition in the three-dimensional state space

Fig. 5. The resulting polyhedral partition $\{\mathcal{Q}_u^n\}_{n=0}^N$ of Algorithm 2 for $u(k) = [1\ {-1}\ {-1}]$ and $N = 4$, where the colors correspond to the number of steps n

*Example 1 (*continued*)* Setting $N = 4$, we proceed with Example 1. Fig. 3 visualizes the first step ($n = 0$) of Algorithm 2 in the x_1x_2 plane, where the same scaling is used for all three figures. Starting with the initial set of states $\mathcal{X}^0$ in Fig. 3(a), the voltage vector $u(k) = [1\ \ {-1}\ \ {-1}]$ maps $\mathcal{X}^0$ from time-step k to $k+1$ as shown in Fig. 3(b). The set $\mathcal{X}^1$ comprises two parts. $\mathcal{X}^1_{\text{feas}}$ ($\mathcal{X}^1_{\text{infs}}$) contains the states corresponding to feasible (infeasible) outputs. This set $\mathcal{X}^1_{\text{infs}}$ is consequently mapped back from time-step $k+1$ to k resulting in $\mathcal{Q}_u^0$ and indicating that this set is zero-step feasible for the chosen $u(k)$. Furthermore, we set $\mathcal{X}^1 = \mathcal{X}^1_{\text{feas}}$.

The second step ($n = 1$) is shown in Fig. 4 starting from the set $\mathcal{X}^1$ at time-step $k+1$ in Fig. 4(a). Applying $u(k) = [1\ \ {-1}\ \ {-1}]$ to this set maps it from time-step $k+1$ to $k+2$ as shown in Fig. 4(b). Again, $\mathcal{X}^2_{\text{feas}}$ ($\mathcal{X}^2_{\text{infs}}$) contains the states corresponding to feasible (infeasible) outputs. The states in $\mathcal{X}^2_{\text{infs}}$ are mapped back for two steps from $k+2$ to k yielding $\mathcal{Q}_u^1$ which is shown in Fig. 4(c) and refers to states which are one-step feasible for $u(k)$.

Repeating the above procedure for $n = 2, 3, 4$ and collecting the sets $\mathcal{Q}_u^n$ yields the polyhedral partition $\{\mathcal{Q}_u^n\}_{n=0}^4$ shown in Fig. 5. The outer polyhedra correspond to outputs that are feasible for zero time-steps when applying $u(k) = [1\ {-1}\ {-1}]$, while the inner polyhedra are feasible for one, two, three and four time-steps as $x_2(k)$ is increasing. Note that $\{\mathcal{Q}_u^n\}_{n=0}^4$ is by construction a polyhedral partition of the set $\mathcal{X}^0$.

The computation time for the second stage for the given example is approximately 2 min on a Pentium IV. ■

Summing up, Stages I and II yield a semi-explicit control law that is evaluated by following Algorithm 1, with the main difference that the number of steps n_u is not calculated by mapping operations but rather by set membership tests evaluating if the given state lies in the respective polyhedron. Specifically, if for the given $u(k-1)$, the state $x(k)$ lies in the core, reapply the last voltage vector again. Else determine for each new voltage vector the polyhedron in $\{\mathcal{Q}_u^n\}_{n=0}^N$ containing $x(k)$, evaluate the associated number of time-steps n_u, and find the voltage vector $u(k)$ with the lowest cost as defined in Algorithm 1. This is formalized in Algorithm 3.

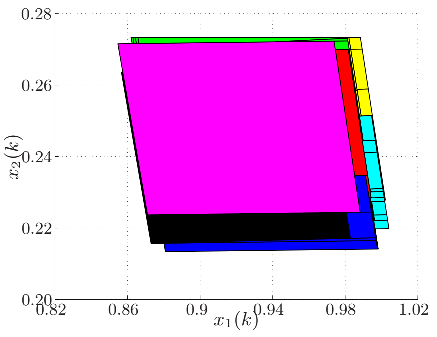
Algorithm 3

```
function $u(k) =$ Algo3 ( $x(k)$, $u(k-1)$ )
    if $x(k) \in \mathcal{Q}_u^c$
        $u(k) = u(k-1)$
    else
        for all $u(k) \in U \setminus u(k-1)$
            determine $n_u$ such that $x(k) \in \mathcal{Q}_u^{n_u}$
        endfor
        $u(k) = \arg\min_{u(k)} \frac{||u(k)-u(k-1)||}{n_u}$
    endif
```

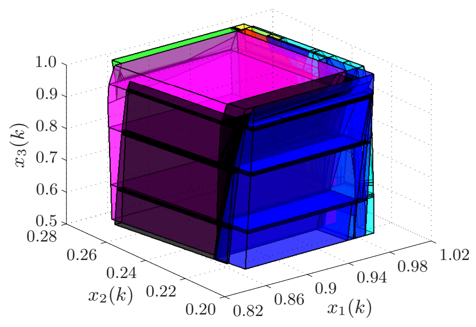
Regarding the computational burden for the on-line computation of the control input, in the worst case, one core needs to be evaluated and the seven polyhedral partitions of $U \setminus u(k-1)$ which feature in general a low number of polyhedra.

Stage III In the third stage we pre-compute Algorithm 3 and derive the fully explicit control law as a function of the last voltage vector $u(k-1)$ and the current state $x(k)$. For $u(k-1) \in U$, we evaluate for each polyhedron in $\{\mathcal{Q}_u^n\}_{n=0}^N$ the cost and associate with it the voltage vector $u(k)$. Next, the core $\mathcal{Q}_u^c$ is added with zero cost and the voltage vector $u(k) = u(k-1)$. Finally, we compare the cost expressions and iteratively remove (parts of) polyhedra with inferior costs[2]. A detailed exposition and analysis of this algorithm can be found in [2]. This yields one polyhedral partition, where each polyhedron refers to a voltage vector $u(k)$ (and not to a number of time-steps). This procedure is repeated for all the eight former voltage vectors $u(k-1)$ yielding eight different fully explicit state-feedback control laws. As a result, the computational burden of evaluating the control law is reduced, as $u(k-1)$ directly defines the one polyhedral partition

[2] As the cost expressions used in Algorithms 1 and 3 are rational, where the nominator (in the case of a two-level inverter) is restricted to the integers two, four and six, and the denominator to $0, \dots, N$, the costs take only a few different values. This increases the possibility that at a given time two or more voltage vectors have the same associated cost leading to ambiguities in the choice of the next voltage vector. In such cases, we suggest to remove the ambiguity by imposing an additional heuristic selection criterion. Examples for such rules are to select the vector that keeps the controlled variables feasible for the maximal number of steps, or to favor zero vectors. Obviously, these ambiguities occur less frequently when the maximal horizon N is increased.



(a) Polyhedral partition in the x_1x_2 plane for $x_3 = 0.95$



(b) Polyhedral partition in the three-dimensional state-space

Fig. 6. Polyhedral partitions of the state-feedback control law resulting from Stage III for $u(k-1) = [1\ \ -1\ \ -1]$, where each color corresponds to a voltage vector $u(k) \in U$

that needs to be searched through in order to obtain $u(k)$. However, the memory requirements are higher since the polyhedral partitions of the fully explicit control law are in general more complex than the one of the semi-explicit control law.

*Example 1 (*continued*)* Applying Stage III to Example 1 yields for $u(k-1) = [1\ \ -1\ \ -1]$ the explicit control law shown in Fig. 6. Each color corresponds to a voltage vector $u(k) \in U$. In particular, the large polyhedron in the center of the three-dimensional state space refers to $u(k) = u(k-1)$. The explicit control law comprises a total of eight control laws similar to Fig. 6, where each one corresponds to a formerly applied voltage vector $u(k-1) \in U$.

The computation was performed using the function `mpt_removeOverlaps` of the Multi-Parametric Toolbox [7]. The computation time was 15 min. ■

5 Simulation Results

The simulation results presented in this section were derived for a DTC drive featuring a two-level inverter. The parameters of the drive are the same as in [9], and the operating point we consider is as in Example 1. As mentioned before, the only design parameter which influences the calculation of the state-feedback controller and consequently the performance of the drive, is the maximal horizon N over which the feasibility of each voltage vector is considered.

In the following, we evaluate the performance of the proposed Feasibility Approach in terms of the average inverter switching frequency. As a benchmark, we employ the Optimal DTC scheme presented in [9]. The corresponding state-feedback controller [9], which was derived for a prediction horizon of two

N	switching frequency [Hz]	total number of polyhedra in semi-explicit control law	total number of polyhedra in fully explicit control law
2	632	292	1192
3	606	440	1891
4	572	625	2226
5	540	860	2907
6	510	1051	3362
7	495	1256	3737
8	547	1467	4443
9	574	1694	4758

Table 1. Performance and complexity of the state-feedback control law

and features a total of 47'000 polyhedra, yields for the above setup an average switching frequency of 525 Hz. Note that in the Optimal DTC scheme switching is allowed at every time-step within the prediction horizon, and that the comparison is based on the same case study as in [9]. In particular, the same drive parameters, operating point and bounds imposed on the torque and the stator flux are used.

The results obtained with the Feasibility Approach are summarized in Table 1 for eight different values of the maximal horizon N. For the horizon used in [9], i.e. $N = 2$, the switching frequency is significantly increased with respect to the benchmark. This is to be expected, since the move blocking strategy (no switching of the control input within the horizon) reduces the degrees of freedom of the control algorithm. However, setting the maximal horizon to $N = 5$ yields a switching frequency that is comparable to the one obtained with the Optimal DTC approach, and the choices of $N = 6$ and $N = 7$ reduce the switching frequency. Most important, this performance improvement is achieved despite the complexity reduction of the state-feedback controller by an order of magnitude with respect to its Optimal DTC counterpart.

Focusing on the case of $N = 7$, the relative switching frequency improvement with respect to the benchmark amounts to 5.7 %. Furthermore, we should point out that using the Feasibility Approach the bounds on the torque and the stator flux are very strictly respected. The Optimal DTC scheme, however, allows for small violations of the bounds; the degree of the violations can be adjusted using a design parameter (penalty on the soft constraints for the bounds) that affects the switching frequency. As tightening the bounds increases the switching frequency, the expected performance improvement is even more pronounced.

For completeness, one should note that the switching frequency does not monotonically decrease with N. This phenomenon has also been observed with the Optimal DTC scheme and is currently under investigation. In particular, a further increase of the horizon to $N = 8$ or $N = 9$ leads to a performance deterioration with respect to $N = 7$.

6 Conclusions and Outlook

In this paper, we have presented the derivation and performance analysis of a state-feedback controller based on MPC for the DTC problem of induction motors driven by a two-level inverter. The proposed controller features a significantly lower complexity (by an order of magnitude) than its counterpart in [9] for the same fixed operating point. It is derived through a simple and systematic design procedure and maintains and even improves the favorable control performance properties obtained by the use of predictive control.

The controller presented in this paper could be extended in the following two ways. Firstly, by considering changes in the operating point. This necessitates the parametrization of the drive's PWA model over the rotational speed ω_r and the torque and flux bounds. Concerning the bounds, only one parameter is needed for the median of the torque bounds. The flux bounds and the width of the torque bounds can be assumed to be in general time-invariant. Obviously, the complexity of the resulting controller would be increased. Yet it is to be expected that the low complexity with respect to an accordingly extended optimal controller in [9] is maintained.

Secondly, this paper can be extended by applying the presented method to a DTC drive with a three-level inverter. As a result, two additional control objectives, namely the regulation of the inverter's neutral point potential and the even distribution of the switching effort between the upper and the lower half of the inverter, arise. A straightforward approach would be to accurately model the nonlinear dynamics of the neutral point potential. To avoid such a substantially more complex PWA model, a favorable approach is to refrain from deriving the fully explicit controller and to rather use the semi-explicit realization in combination with time-varying weights on the voltage vectors. An outer loop should monitor the neutral point potential and set the weights accordingly to favor the selection of voltage vectors that keep the potential within given bounds around zero. The same approach can be also used for the even distribution of the switching effort. Since these control objectives are roughly and heuristically defined, they do not require to be strictly met thus rendering the above approach a sufficient approximation.

The full version of this paper is available as technical report [4] extending the modelling in Section 2.

7 Acknowledgements

This work was supported by ABB Switzerland Ltd., and by the IST research project IST-2001-33520 *Control and Computation* of the EU. The authors would like to thank Christian Stulz, Pieder Joerg and Petri Schroderus of ABB ATDD, Turgi, Switzerland, and Andreas Poncet of ABB Corporate Research, Baden-Dättwil, Switzerland, for their continuous advice. Most importantly, we would like to acknowledge the contribution of Manfred Morari from ETH Zürich, Switzerland.

References

1. ABB Asea Brown Boveri Ltd. Product webpage of ACS 6000. online document. `www.abb.com/motors&drives`.
2. M. Baotić and F.D. Torrisi. Polycover. Technical Report AUT03-11, Automatic Control Laboratory ETH Zurich, `http://control.ee.ethz.ch/`, 2003.
3. D.P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 1995.
4. T. Geyer, G. Papafotiou, and M. Morari. Direct torque control for induction motor drives: A model predictive control approach based on feasibility. Technical Report AUT04-09, Automatic Control Laboratory ETH Zurich, `http://control.ee.ethz.ch/`, 2004.
5. T. Geyer, F.D. Torrisi, and M. Morari. Efficient mode enumeration of compositional hybrid systems. In A. Pnueli and O. Maler, editors, *Hybrid Systems: Computation and Control*, volume 2623 of *Lecture Notes in Computer Science*, pages 216–232. Springer-Verlag, 2003.
6. P.C. Krause. *Analysis of Electric Machinery*. McGraw-Hill, NY, 1986.
7. M. Kvasnica, P. Grieder, M. Baotić, and M. Morari. Multi parametric toolbox (MPT). In R. Alur and G. Pappas, editors, *Hybrid Systems: Computation and Control*, volume 2993 of *Lecture Notes in Computer Science*, pages 448–462. Springer-Verlag, 2004. `http://control.ee.ethz.ch/~mpt`.
8. J.M. Maciejowski. *Predictive Control*. Prentice Hall, 2002.
9. G. Papafotiou, T. Geyer, and M. Morari. Optimal direct torque control of three-phase symmetric induction motors. Technical Report AUT03-07, Automatic Control Laboratory ETH Zurich, `http://control.ee.ethz.ch/`, 2003.
10. G. Papafotiou, T. Geyer, and M. Morari. Optimal direct torque control of three-phase symmetric induction motors. In *Proceedings of the 43th IEEE Conference on Decision and Control*, Atlantis, Bahamas, December 2004.
11. P. Pohjalainen, P. Tiitinen, and J. Lulu. The next generation motor control method - direct torque control, DTC. In *Proceedings of the European Power Electronics Chapter Symposium*, volume 1, pages 115–120, Lausanne, Switzerland, 1994.
12. E.D. Sontag. Nonlinear regulation: The piecewise linear approach. *IEEE Transactions on Automatic Control*, 26(2):346–358, April 1981.
13. I. Takahashi and T. Noguchi. A new quick response and high efficiency control strategy for the induction motor. *IEEE Transactions on Industry Applications*, 22(2):820–827, September/October 1986.
14. I. Takahashi and Y. Ohmori. High-performance direct torque control of an induction motor. *IEEE Transactions on Industry Applications*, 25(2):257–264, March/April 1989.
15. F.D. Torrisi and A. Bemporad. Hysdel — a tool for generating computational hybrid models for analysis and synthesis problems. *IEEE Transactions on Control Systems Technology*, 12(2):235–249, March 2004.