Integration of Inverter Constraints in Geometrical Quantification of the Optimal Solution to an MPC Controller

Johan Raath
Department of Electrical, Electronic and Computer Engineering
Central University of Technology
South Africa
Email: jraath@cut.ac.za

Du Toit Mouton
Department of Electrical and Electronic Engineering
University of Stellenbosch
South Africa
Email: dtmouton@sun.ac.za

Tobias Geyer
ABB Corporate Research
ABB Switzerland Ltd,
Power Electronic Systems
Switzerland
Email: t.geyer@ieee.org

***Abstract*—This paper considers a model predictive controller with reference tracking that manipulates the integer switch positions of a power converter. It is shown that the optimal switch position can be determined with a reduced online computational burden. Instead of solving the optimization problem in each sampling period, the optimization problem can be solved in a new coordinate system, partially offline by means of a polyhedral partition of the solution space. The optimal switch position can then be found during online operation by using a binary search tree. This concept is exemplified for a three-level, single-phase converter with an RL load.**

I. Introduction

Advances in the fields of mathematical optimization and the increased computational power of controller hardware have made it possible to consider model predictive control (MPC) in power electronic systems with short sampling intervals [1]. MPC using larger horizons has the potential to deliver significant performance benefits, but requires more computations at each sampling instant to solve the associated optimization problem [2], [3]. The online computational burden of MPC can be lessened by finding a solution to the MPC problem offline [4], [5]. Storing the solution/control law in a look-up table eliminates the need for solving an optimization problem online.

The main purpose of the research presented here is to find an alternative to sphere decoding with a *fixed* online computational burden so as to implement MPC practically for a multilevel inverter. A discrete time-invariant system with receding horizon and a finite control set is considered. Similar to [6] a control law is formulated by means of the polyhedral partition of the state space. In [6] exact analytical expressions are developed offline for the partition to avoid the need for online optimization in finding the control law. This study partially finds the solution offline in the form of a binary search tree (BST) in an attempt to reduce the online evaluation of the partitioned space. A binary search tree of minimum depth can be achieved only if the partitioned state-space is of the lowest possible complexity. This paper presents an algorithm for reducing the complexity of the partitioned state-space by utilizing the Delaunay triangulation.

The paper is organized as follows: Section 2 introduces the model of a multilevel inverter with RL load and the mathematical background to the MPC problem. In section 3 the partitioned state-space is geometrically presented. The approach to complexity reduction along with the proposed algorithm are presented in section 4. Section 5 describes the advantages obtainable from using a binary search tree. Simulation of an MPC controlled inverter with the proposed principles is demonstrated in section 6 and section 7 concludes the paper.

II. Model Predictive Control

Model predictive control is a method in which the control action is determined by solving a finite horizon open-loop optimal control problem at each sampling instant, using the current state of the system as initial state, searching for an optimal control sequence over the set horizon and then applying the first control action in this sequence to the system. With reference current tracking the general aim is to control the inverter switches in such a manner so as to generate an output current i in the load that tracks a reference current i_r as closely as possible with minimal internal switching losses in the inverter.

A. Modeling

In order for the MPC controller to predict the possible currents in the load, a mathematical model for the system needs to be derived. Consider a single-phase Neutral Point Clamped (NPC) inverter with the neutral point assumed constant. The topology is shown in Fig. 1. The inverter leg can deliver three voltage levels of $-0.5V_{DC}$, $0V_{DC}$ and $+0.5V_{DC}$ across the load. These output levels can be represented by the integer

978-1-5090-1815-4/16/$31.00©2016 IEEE

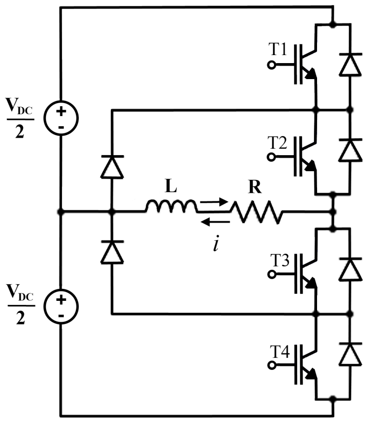

Fig. 1. Single-phase Neutral Point Clamped inverter

values $u \in \{-1, 0, +1\}$ that define the state of the switch positions in the inverter leg. The voltage applied to the RL load is thus equal to $v(t) = 0.5V_{DC} \cdot u(t)$. The RL load equation in the continuous time domain,

$$v(t) = Ri(t) + L\frac{di(t)}{dt}.$$

Converting the differential to the discrete time domain results in the predictive load current model $i(k+1)$ with input vector $u(k)$ and state vector $i(k)$ as noted in,

$$i(k+1) = Ai(k) + Bu(k) \tag{1}$$

with, $A = e^{-T_s/\tau}$, $B = \frac{V_{DC}}{2R}(1 - A)$ and $\tau = \frac{L}{R}$.

B. Cost function

To find the optimal control input to the inverter, all the the predicted currents over a finite horizon and the respective switching states are subjected to a quadratic cost function,

$$J = \sum_{l=k}^{k+N-1} \|i_r(l+1) - i(l+1)\|_2^2 + \lambda_u \|\Delta_u(l)\|_2^2. \tag{2}$$

It is similar to the cost function defined by [2] and consist of two terms, $\|i_r(l+1) - i(l+1)\|_2^2$ to quantify tracking error from the reference current i_r, and $\|u(l) - u(l-1)\|_2^2$ the switching cost. A tuning factor λ_u is used to adjust the weight of the switching cost. Shoot through in the inverter leg is avoided by adhering to the switching constraint $\|u(l) - u(l-1)\|_2^2 \leq 1$. J is a function of the switching sequence $U = [u(k), u(k+1), ...u(k+N-1)]^T$ which leads to an exponential increase in possible switching sequences over horizon N to evaluate.

C. Optimization problem

The optimization problem for finding the optimum switching sequence U_{opt} over the finite horizon can be stated formally as,

$$U_{opt}(k) = \arg\min_{U(k)} J \tag{3}$$

subject to

$$\begin{aligned} i(l+1) &= Ai(l) + Bu(l) \\ u(l) &\in \{-1, 0, +1\} \\ \|\Delta_u(l)\|_2^2 &\leq 1 \\ \forall l &= k, ..., k+N-1 \end{aligned} \tag{4}$$

A solution to the optimization problem (3) can be found by rewriting the cost function in terms of the unconstrained optimal solution $U_{unc}(k)$ as derived in [2] and [6],

$$J = \|HU(k) - HU_{unc}(k)\|_2^2. \tag{5}$$

H is a transformation matrix that transforms the switching sequence $U(k)$ and the unconstrained optimal $U_{unc}(k)$ to the H-coordinate solution space. The optimization problem (3) with cost function (5) then translates into the nearest neighbor search of the N-dimensional vector $HU_{unc}(k)$ to the set of N-dimensional input vectors $HU(k)$ in $\mathbb{R}^N$ Euclidean space.

III. QUANTIZATION

Quantization of HU_{unc} to the nearest HU vector can be accomplished using the exhaustive search method which enumerates all possibilities and verifies if the switching constraint is satisfied [1]. This method is simple but computationally expensive for long horizons and can thus only be used for short horizons. Another option is sphere decoding which has been shown to be effective in extended horizon MPC problems [2]. This research is aimed at exploring the H-coordinate space and computing a geometrical solution in the format of a binary search tree (BST) to solve the associated MPC problem.

The first step in our approach is the partitioning of the H-coordinate space into a Voronoi diagram consisting of convex polyhedra or so called Voronoi cells V_i. A Voronoi diagram is a structure that is extremely efficient in exploring a local neighborhood in a geometric space [7]. The diagram starts out with a given set of points, also referred to as sites or seeds, in $\mathbb{R}^N$ Euclidean space. Next, the space is uniquely partitioned into disjoint polyhedra such that each polyhedron is assigned to one single site and covers all the points in space that are closer to that specific site than to any other site [8]. In our case, the sites are the HU switching sequences arranged in the N-dimensional, H-coordinate space. The corresponding Voronoi diagram can be defined as the following set of polyhedra,

$$V_i = \{x : \|x - HU_i\| < \|x - HU_j\|\} \tag{6}$$

for

$$i = 1, 2, ..., 3^N, \forall j \neq i.$$

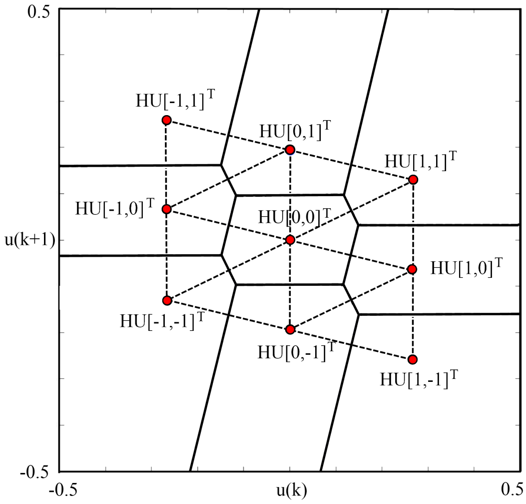

Fig. 2. Voronoi diagram and Delaunay triangulation of HU-sequences ($N = 2$).

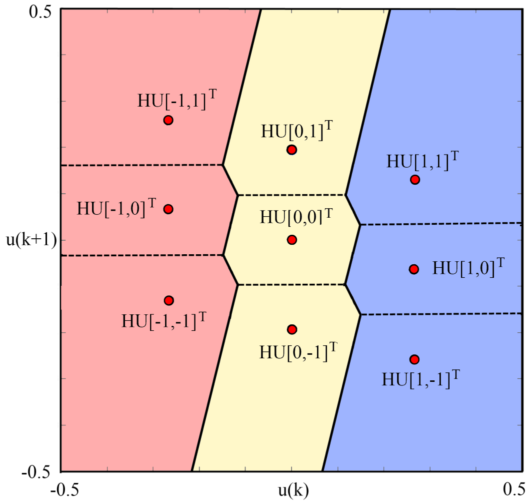

Fig. 3. Voronoi diagram of unified polyhedra ($N = 2$).

TABLE I
UNCONSTRAINED HU-SEQUENCES AND DEFINING HYPERPLANES.

Horizon (N)	1	2	3	4	5	6
HU-sequences	3	9	27	81	243	729
Hyperplanes	2	16	98	544	2882	14896

Fig. 2 illustrates the Voronoi diagram for a typical horizon $N = 2$ example. The quantification of vector HU_{unc} in the H-coordinate space can be done by determining in which of the polyhedra it resides. The most immediate way of solving this point location problem is to carry out a sequential search through all the polyhedra. For the horizon $N = 2$ example, it translates into the linear investigation ($A^T x - b$) of the 16 hyperplanes that define the 9 polyhedra. This exhaustive method is computationally expensive and not viable for applications in higher dimensions [9]. Table I highlights the explosive nature of the number of hyperplanes required to define the Voronoi diagram as the horizon is extended.

IV. COMPLEXITY REDUCTION

Extension of the MPC horizon results in higher dimensionality of the H-transformed coordinate space with increased geometrical complexity of the Voronoi diagram. The standard approach to complexity reduction is to unify adjacent polyhedral partitions with similar control laws $u(k)$ [10]. From Fig. 2 it can be noticed that some of the $HU(k)$-sites have equal first terms and thus represent the same control law. Unification of the 9 polyhedra into three subsets, each representing one of the control laws $u \in \{-1, 0, +1\}$ reduces the number of hyperplanes to be investigated in the point location of HU_{unc} from 16 to 10 for the horizon $N = 2$ case. Fig. 3 displays this unification graphically.

Additional reduction in complexity is achieved by applying the hard switching constraints of the inverter. Switching sequences with transitions from state $+1$ to state -1 and vice versa are not allowed and eliminated as a possible solution. Imposing the switching constraint from $u(k-1)$ to $u(k)$ demands an individual Voronoi diagram for each $u(k-1) \in \{-1, 0, +1\}$ possibility. Also in each Voronoi diagram the switching constraint is applied to the respective HU-sequences within the horizon. Figures 4 to 6 illustrate the Voronoi diagrams for the different possibilities of $u(k-1)$ with the eliminated HU-sequences indicated with an "x". For example, if $u(k-1) = 0$ then switching to any control law $u(k) \in \{-1, 0, +1\}$ will satisfy the switching constraint and is allowed. Within the horizon, sequences $HU[+1, -1]$ and $HU[-1, +1]$ do not adhere to the switching constraint and are eliminated. The number of allowed HU-sequences are reduced to 7 and the number of hyperplanes defining the unified polyhedra now amounts to 8. Similarly the Voronoi diagrams for $u(k-1) \in \{-1, +1\}$ are made up of 5 HU-sequences and 4 defining hyperplanes in each case. The rest of the article will refer only to the $u(k-1) = 0$ condition since its Voronoi diagram is of highest complexity thus being worst case in both the offline tree building and online point location processes. Table II lists the number of HU-sequences and defining hyperplanes for $u(k-1) = 0$. A significant reduction can be observed if the numbers are compared to the unconstrained sequences and hyperplanes of Table I.

A. Algorithm

This paper proposes an alternative algorithm with a direct approach in extracting from the Voronoi diagram, *only* the hyperplanes defining the hull of unified polyhedra. Other than following the traditional approach of determining Voronoi regions, applying complexity reduction and extracting common facets, we utilize the Delaunay triangulation and unique spatial arrangement of the HU-sequences to extract the defining hyperplanes.

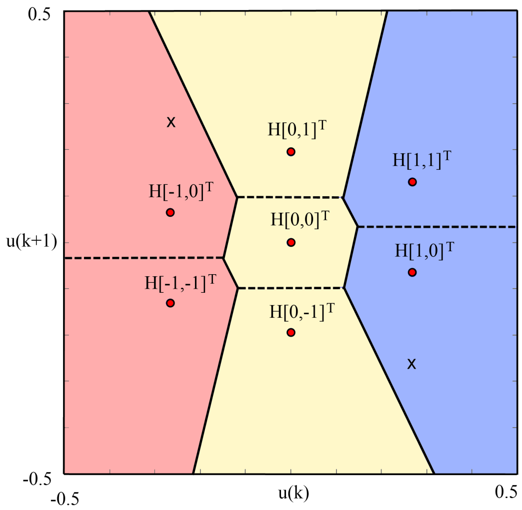

Fig. 4. Voronoi diagram for $u(k-1) = 0$ with constrained sequences ($N = 2$).

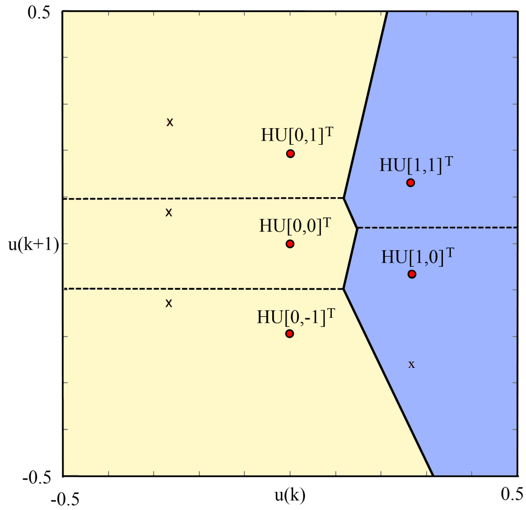

Fig. 6. Voronoi diagram for $u(k-1) = +1$ with constrained sequences ($N = 2$).

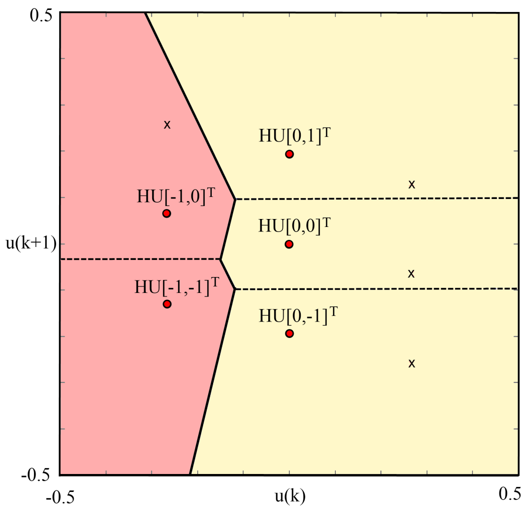

Fig. 5. Voronoi diagram for $u(k-1) = -1$ with constrained sequences ($N = 2$).

The Delaunay triangulation has a unique property in it being the dual graph of the Voronoi diagram and vice versa [11]. This duality translates into a Delaunay edge (line-segment connecting two sites) being orthogonal to, and bisected by the Voronoi hyperplane shared by the respective sites. This can be observed in Fig. 2 where the Delaunay triangulation edges are shown in dotted lines. The principle is used in many algorithms for obtaining the Voronoi diagram from its dual [12]. We apply the same principle but only determine the minimal number of hyperplanes defining the hull of the unified polyhedra, hence eliminating unnecessary computations. We achieve complexity reduction by removing certain edges from the Delaunay triangulation before calculating the respective Voronoi hyperplanes. Delaunay edges that connect sites with the same control law values $u(k)$ are removed since their dual

TABLE II
CONSTRAINED HU SEQUENCES AND DEFINING HYPERPLANES FOR $u(k-1) = 0$

Horizon (N)	1	2	3	4	5	6
HU-sequences	3	7	17	41	99	239
Hyperplanes	2	8	32	126	496	1952

(Voronoi hyperplane) would be of no significance in solving the point location problem. The proposed procedure for the direct extraction of the defining hyperplanes is described in Algorithm 1.

Algorithm 1 Hyperplane extraction algorithm

Step 1 Find the Delaunay triangulation of HU-sequences.
Step 2 For all Delaunay edges (line segments),
Index edges connecting sites with non-similar control laws $u(k)$, realizing border-spanning edges.
Step 3 For each border-spanning edge,
Assign the site with $u(k) \neq 0$ as the normal vector to a hyperplane,
Assign the mid-point of the edge as a point in the hyperplane,
Define the hyperplane in point-normal format.
Step 4 Index the hyperplanes in border defining sets.

V. BINARY SEARCH TREE

Although the number of hyperplanes obtained from algorithm 1 are minimal in defining the unified polyhedra it is not computationally viable to follow the exhaustive search method for solving the point location problem. The number of hyperplanes to investigate can be reduced by constructing a binary search tree (BST). A binary search tree relies on the principle of binary space partitioning where an N-dimensional space is recursively divided by $(N-1)$-dimensional hyperplanes until the partitioning satisfies one or more requirements [13].

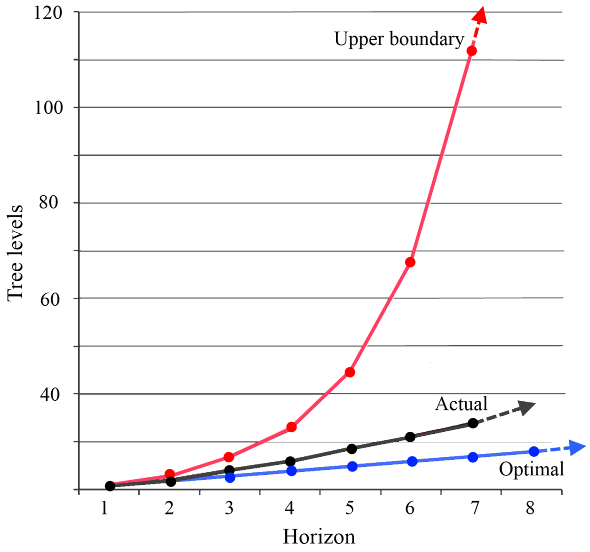

Fig. 7. Binary search tree levels for N-dimensional 3-level hypercube.

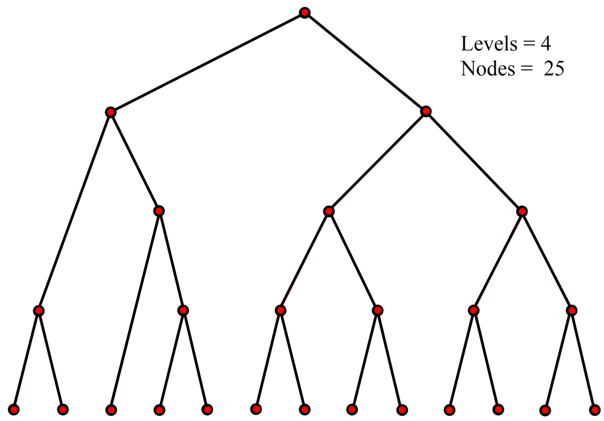

Fig. 8. Tree stucture for $u(k-1) = 0$ and horizon $N = 2$.

TABLE III
BINARY SEARCH TREE LEVELS FOR $u(k-1) = 0$.

Horizon(N)	1	2	3	4	5	6
HU-sequences	3	7	17	41	99	239
BST levels(depth)	2	4	8	12	17	22

A binary search tree is build offline and stored in a look-up table for online use. During online use the tree is traversed where at each tree level one specific hyperplane is evaluated until a leaf node (solution) is reached. Based on this principle the number of tree levels L to traverse for an ideal spatial arrangement of n number of sites is $L = log_2(n)$ [5].

In this research a three-level inverter is used constituting three control options $\{-1, 0, +1\}$ per horizon or dimension in the single-phase case. As seen in Fig. 2 the resulting spatial arrangement of the HU-sequences is a skewed N-dimensional hypercube consisting of 3^N sequences. An orthogonal spatial arrangement of the HU-sequences will lead to a Voronoi diagram of least complexity. Such an orthogonal arrangement has $(2 \times N)$ hyperplanes defining the Voronoi diagram. The most complex polyhedron in this Voronoi diagram namely the bounded polyhedron enclosing the origin is also defined by the same $(2 \times N)$ hyperplanes. Thus in a representative binary search tree a maximum of $L = log_{1.732}(n)$ levels needs to be traversed (hyperplanes evaluated) for achieving point location. The orthogonal arrangement is ideal since it makes optimal use of binary space partitioning and can be taken as the optimal or lower bound for our three-level hypercube.

The typical spatial arrangement of HU-sequences is skewed and non-orthogonal. In finding the expected upper bound number of tree levels for the skewed three-level hypercube, we consider the minimum number of hyperplanes defining the most complex polyhedra in the associated Voronoi diagram. Similar to the orthogonal case it is the bounded polyhedron enclosing the origin. The number of hyperplanes defining this polyhedron were determined experimentally and are plotted in in Fig. 7 together with the the expected lower bound. The upper bound indicates a non-linear growth of tree levels into higher dimensions and can be adjudicated to the almost exponential increase in both HU-sequences and polyhedral complexity of the Voronoi diagram.

In constructing a binary search tree for the partitioned space an adapted version of the binary search tree algorithm proposed in [9] was used. Utilizing the hyperplanes obtained from Algorithm 1, tree levels for the $u(k-1) = 0$ case were obtained as listed in Table III and plotted as the actual curve in Fig. 7. The number of hyperplanes obtained from Algorithm 1 has a direct influence on the offline tree building process. For horizons $N > 6$ the tree building process became extremely burdened by the sheer magnitude of hyperplanes that must be tested and selected at each individual tree node. During online operation the horizon $N = 3$ case as an example requires 8 N-dimensional linear calculations (hyperplane evaluations) in every sampling period. Compared to the exhaustive search method which would require 27 N-dimensional quadratic distance calculations the binary search tree results seems favorable and more so as the horizon increase. Fig. 8 graphically displays a typical tree structure for the horizon $N = 2$ example consisting of 4 levels and 25 nodes with the leaf nodes displayed as the bottom row.

VI. SYSTEM IMPLEMENTATION

The proposed principles in complexity reduction were implemented in a MATLAB simulation model of the single-phase NPC three-level inverter to compare the validity of the output voltage and current waveforms to those of the benchmark exhaustive method. The simulated system consists of two modules, an *offline-* and an *online-* procedure respectively. Consider a typical example for horizon $N = 2$ with steady-state conditions and the following parameters. A sampling interval of $T_S = 25\mu$s, load- resistance of $R = 2\Omega$, and inductance $L = 2m$H. The rated r.m.s. output voltage of the inverter is $V_{AC} = 3.3k$V with an input dc-link voltage of $V_{DC} = 5.2k$V. Base quantities are used to establish a per unit system and the current reference is assumed to be 0.8pu amplitude. A tuning factor λ_u of 0.02 was selected.

The *offline* procedure Algorithm 2 produces the H-transformation matrix together with the three binary search trees for each of the possible switching states for $u(k-1)$. All the system parameters are captured in the H-transformation matrix. If any of the parameters change the offline procedure must be repeated.

Algorithm 2 *Offline* algorithm

Step 1 Obtain the load model parameters.
Step 2 Calculate H-transformation matrix.
Step 3 For each switching position $u(k-1) \in \{-1, 0, +1\}$,
Define the allowed switching sequences over horizon N,
Find the hyperplanes defining the polyhedral partition of the solution space (Algorithm 1),
Construct the relevant binary search tree.
Step 4 Store BST information in a sparse matrices.

The *online* procedure Algorithm 3 requires the reference current $i_r(k)$, state current $i(k)$ and H-transformation matrix for calculation of the unconstrained optimal HU_{unc}. Selecting the relevant binary search tree for $u(k-1)$ and traversing the tree in quantifying HU_{unc} leads to the constrained optimum control $u_{opt}(k)$ for application to the inverter leg.

Algorithm 3 *Online* algorithm

Step 1 Sample state $i(k)$ and reference current $i_r(k)$.
Step 2 Calculate U_{unc} and HU_{unc}.
Step 3 For $u(k-1)$ select the relevant binary search tree,
Traverse the tree to a leaf node depicting the optimal sequence $U_{opt}(k)$.
Step 4 Apply the control law $u_{opt}(k)$ to the inverter.

For the parameters stated above the simulated output voltage and current waveforms are displayed in Fig. 9. The waveforms were found to be identical to the waveforms produced from using the exhaustive method.

VII. Conclusion

We have presented an algorithm for extracting the hyperplanes defining the control regions in the partitioned state-space of an MPC controller for a single-phase, three-level NPC inverter. The algorithm is simple, efficient and extends into higher dimensions. The extracted hyperplanes were used in building a binary search tree offline and utilized online in solution of the MPC problem. The results obtained in terms of tree levels for extended horizons are favorable. Applying the inverter switching constraints reduced the solution space complexity and number of defining hyperplanes. The complexity incurred by opting for a *three-level* inverter rendered

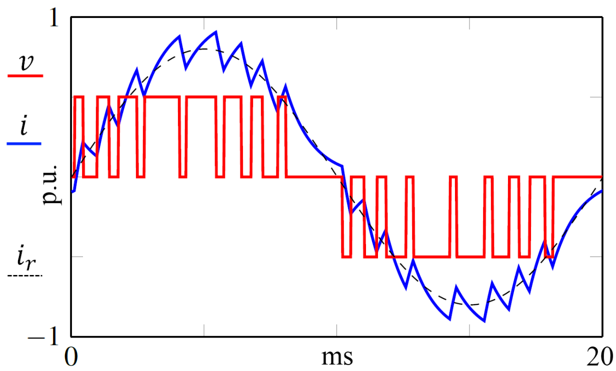

Fig. 9. Output voltage and reference tracking load current

the solution space useful to dimension six. In a three-phase, *three-level* inverter application a prediction horizon of $N = 2$ will necessitate an equivalent $3N$-dimensional solution space. Although the prediction horizons achieved in this paper are limited, it allows one to understand the controller actions with integrated inverter constraints. This approach lends itself to multi-parametric problems of limited complexity where a controller with fixed computational burden is required.

References

[1] J. Rodriguez, M. P. Kazmierkowski, J. R. Espinoza, P. Zanchetta, H. Abu-Rub, H. Young, et al., "State of the art of finite control set model predictive control in power electronics", *Industrial Informatics, IEEE Transactions on*, vol. 9, pp. 1003-1016, 2013.
[2] T. Geyer and D. E. Quevedo, "Multistep finite control set model predictive control for power electronics", *Power Electronics, IEEE Transactions on*, vol. 29, pp. 6836-6846, 2014.
[3] D. Quevedo, R. Aguilera, and T. Geyer, "Predictive Control in Power Electronics and Drives: Basic Concepts, Theory, and Methods", *in Advanced and Intelligent Control in Power Electronics and Drives*, Vol. 531, T. Orowska-Kowalska, F. Blaabjerg, and J. Rodrguez, Eds., ed: Springer International Publishing, pp. 181-226, 2014.
[4] Bemporad, A., Morari, M., Dua, V., and Pistikopoulos, E. N., "The explicit linear quadratic regulator for constrained systems", *Automatica*, Vol. 38, No. 1, pp. 320, 2002.
[5] P. Tondel, T.A. Johansen, and A. Bemporad, "An algorithm for multi-parametric quadratic programming and explicit MPC solutions", *Automatica*, Vol. 39, No. 3, pp. 489497, 2003.
[6] D. E. Quevedo, G. C. Goodwin, and J. A. De Dona, "Finite constraint set receding horizon quadratic control," *International journal of robust and nonlinear control*, vol. 14, pp. 355-377, 2004.
[7] F. Aurenhammer, "Voronoi diagrams: a survey of a fundamental geometric data structure", *ACM Computing Surveys (CSUR)*, 1991.
[8] G. M. Voronoi, "Nouvelles applications des parametres continus a la theorie des formes quadratiques. deuxieme Memoire: Recherches sur les parallelloedres primitifs", *J. Reine Angew. Math.*, 134:198-287, 1908.
[9] P. Tondel, T. A. Johansen, and A. Bemporad, "Evaluation of piecewise affine control via binary search tree", *Automatica*, Vol. 39, pp. 945-950, 2003.
[10] T. Geyer, F. D. Torrisi, and M. Morari, "Optimal complexity reduction of polyhedral piecewise affine systems", *Automatica*, vol. 44, pp. 1728-1740, 2008.
[11] B. Delaunay, "Sur la sphere vide. A la memoire de Georges Voronoi", *Izv. Akad. Nauk SSSR, Otdelenie Matematicheskih i Estestvennyh Nauk*, 7:793-800, 1934.
[12] D. F. Watson, "Computing the n-dimensional Delaunay tessellation with application to Voronoi polytopes", *The Computer Journal*, 24(2):167-172, 1981.
[13] W. C. Thibault and B. F. Naylor, "Set operations on polyhedra using binary space partitioning trees", *ACM SIGGRAPH computer graphics*, pp. 153-162, 1987.