On the Micciancio-Voulgaris algorithm to solve the long-horizon direct MPC optimization problem

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

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*—It is widely accepted that model predictive control (MPC) with long prediction horizons yields, in general, a better performance than with short horizons. In the context of power electronic systems, the main advantages include improved closed-loop stability and lower current distortion per switching frequency. A shortcoming of MPC with long prediction horizons is the computational burden associated with solving the optimization problem in real time, which limits the minimum possible sampling interval. The solution to the MPC optimization problem is a polyhedral partition of the state-space. Pre-processing of the state-space and storing representative information thereof offline assists in reducing the online computational burden. The problem structure is a special case in the form of a truncated lattice. Exploiting this characteristic enables representation of the partitioned space to be is stored as a minimal set of Voronoi relevant vectors describing the basic Voronoi cell of a lattice. We evaluate the algorithm proposed by Micciancio and Voulgaris known as the *MV*-algorithm to solve the closest vector problem with pre-processing (CVPP). The performance of the algorithm is evaluated in a simulated three-level neutral point clamped (NPC) voltage source inverter with an RL load.**

I. Introduction

Using MPC with long horizons in power electronic applications has a significant potential to improve the performance of three-phase inverters [1], [2] and [3]. Unfortunately, long horizon MPC requires more computations to obtain the optimum solution which should be reached within a sampling interval if it is to succeed in a real time application. In power electronic systems with short sampling intervals, this computational burden poses a challenge. Finding the solution to the optimization problem translates to finding the predicted control vector with minimum Euclidean distance to the unconstrained optimum in a transformed solution space. This nearest neighbor problem (NNP) is also known as the closest vector problem (CVP) in lattice theory. Three main approaches exist for solving CVP. The first class are enumeration based algorithms which follow the Pohst [4] strategy and have traditionally been used as a practical tool [5]. The second are space saturation algorithms, originally proposed by Kannan [6]. The last approach are Voronoi based algorithms, an approach this research is following in the form of the *MV*-algorithm [7] to find the *exact* solution to the CVP, i.e. the MPC problem. The exact solution is a challenging prospect, as it has been shown to be non-deterministic polynomial-time hard (NP-hard) [8].

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 solve the CVP along with the proposed algorithm is presented in section 4. Section 5 demonstrates the simulation of an MPC controlled inverter with the proposed principles and section 6 concludes the paper.

II. Modeling

A three-phase neutral point clamped (NPC) inverter, of which the neutral point potential is assumed to be constant, can deliver three discrete voltage levels of $-0.5V_{DC}$, 0 and $0.5V_{DC}$ at each phase terminal $x \in \{a, b, c\}$ to an RL-load. These voltage levels can be represented by the integer values $u_x \in \{-1, 0, 1\}$ defining the state of the respective inverter leg. In the discrete-time domain, the load current can be predicted by

$$i(k+1) = Ai(k) + Bu(k), k \in \mathbb{N}, \tag{1}$$

where $u(k)$ denotes the inverter input $u = [u_a u_b u_c]^T$ and $i(k)$ is the three-phase load current transformed to the $\alpha\beta$-coordinate system $i = K[i_a i_b i_c]^T = [i_\alpha i_\beta]^T$, where K is the Clarke transformation matrix. For a more informative description of the derived three-phase model and its matrices, the reader is referred to [3]. The quadratic cost function is defined over a finite horizon N, with i_r being the reference current in the $\alpha\beta$ coordinate system.

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

This cost function is similar to that in [1] which consists of two terms; $\|i_r(l+1) - i(l+1)\|_2^2$ quantifies the current tracking error and $\lambda \|(u(l) - u(l-1))\|_2^2$ the switch-

ing cost with the tuning factor λ to adjust the weight thereof. J is a function of the switching sequence $U(k) = [u^T(k)u^T(k+1)...u^T(k+N-1)]$, resulting in 3^d feasible switching sequences to be evaluated during each sampling period. For a three-phase NPC inverter, we have $d = 3N$. The optimization problem for obtaining the optimum switching sequence U_{opt} can be stated formally as

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

subject to

$$\begin{aligned} i(l+1) &= Ai(l) + Bu(l) \\ u(l) &\in \{-1, 0, 1\}^3 \\ \forall l &= k, ..., k+N-1. \end{aligned} \tag{4}$$

A solution to the optimization problem (3)-(4) can be found by rewriting the cost function in terms of the *unconstrained optimal* solution $U_{unc}(k)$ as derived in [1] and [9]

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

H is an invertible upper triangular matrix that transforms the unconstrained solution U_{unc} and the set of 3^d input vectors U to a d-dimensional solution space. Obtaining U_{opt} in (3) with (5) translates to finding from the set of HU vectors the vector with minimum Euclidean distance to the unconstrained solution HU_{unc} in the transformed H-coordinate solution space. Only the first control input $u_{opt}(k)$ in the optimal switching sequence $U_{opt}(k)$ is applied to the inverter.

III. Solution Space

A. Convex hull

Transforming the input vectors to the H-solution space results in HU vectors arranged in a lattice structure represented by $\Lambda(H) = \Lambda(h_1, h_2, ..., h_d) = \{HU : U \in \mathcal{U}\}$ with $\mathcal{U} = \{-1, 0, +1\}^d$. The basis of the lattice $(h_1, h_2, .., h_d)$ are linearly independent column vectors of H. The span of Λ can be interpreted as an outer boundary or convex *hull* of the set of possible HU solutions. It has the shape of a d-dimensional parallelotope with $2d$, $(d-1)$-dimensional hyperplanes

$$P = \pm \left\{ x \in \mathbb{R}^d : \langle n_i \cdot (x - h_i) \rangle = 0, i = 1, \dots d \right\} \tag{6}$$

with the normal $n_i \in \mathbb{R}^d, n_i \neq \underline{0}$ to the orthogonal base $(x - h_i)$ of the relevant hyperplane. Scaling the normals with

$$n_i = \frac{\langle n_i \cdot h_i \rangle}{\|n\|^2} n_i \tag{7}$$

allows the span of the lattice $\Lambda(H)$ to be defined as the intersection of half spaces

$$P'(\Lambda, \underline{0}) = \left\{ x \in \mathbb{R}^d : |\langle x \cdot n \rangle| \leq \|n\|^2, \forall n, n \neq \underline{0} \right\}. \tag{8}$$

Fig. 1 illustrates this statement by example of a single-phase inverter with horizon $N = 2$ in a two-dimensional $d = 2$ lattice structure. The basis vectors h_i and normals n_i are indicated with the darkened parallelotope highlighting the span of the lattice.

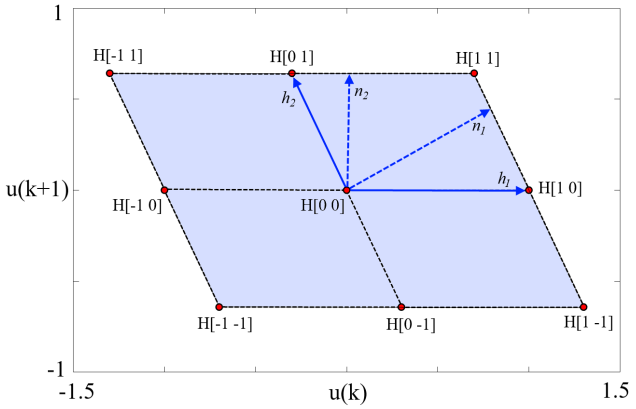

Fig. 1. H-matrix base vectors and HU input vectors for a single-phase inverter with horizon $N = 2$.

B. Polyhedral partitioning

Partitioning of $\mathbb{R}^d$ space into a *Voronoi diagram* results in a number of convex Voronoi cells, each consisting of all points nearest to a respective Voronoi site [10]. The Voronoi cell V of an individual site $s \in HU$ can be defined as

$$V(\Lambda, s) = \left\{ x \in \mathbb{R}^d : \|x - s\| \leq \|x - j\|, \forall j \in \Lambda, j \neq s \right\}. \tag{9}$$

In our three-level inverter application the lattice Λ is truncated with $U \in \mathcal{U}$ and only the Voronoi cell of the origin, $s = \underline{0}$ is bounded. All the other Voronoi cells are unbounded since their respective HU-site resides on the convex hull of the parallelotope. The Voronoi cell enclosing the origin is also known as the *basic Voronoi cell* of a lattice, which is a polytope that is symmetrical in reflection through its Voronoi site $\underline{0}$. *Voronoi relevant vectors* v are lattice points closest to a specific lattice point; in a d-dimensional lattice a maximum of $(2^{d+1} - 2)$ Voronoi relevant vectors can exist [11]. Each Voronoi relevant vector is bisected orthogonally at its midpoint $\frac{1}{2}v$ by a hyperplane that defines the border between the Voronoi site and its Voronoi relevant vector. Due to the symmetrical nature of the basic Voronoi cell, the Voronoi relevant vectors can be introduced as a subset of the lattice $S(\Lambda) \subseteq \Lambda$ and are subsequently used to define the basic Voronoi cell

$$V(\Lambda, \underline{0}) = \left\{ x \in \mathbb{R}^d : \|x\| \leq \|x - v\|, \|x\| \leq \|x + v\|, \forall v \in S(\Lambda), v \neq \underline{0} \right\}. \tag{10}$$

The Voronoi diagram for the two-dimensional example is illustrated in Fig. 3. The shaded polytope refers to the basic Voronoi cell, and the Voronoi relevant vectors $\pm v$ are indicated.

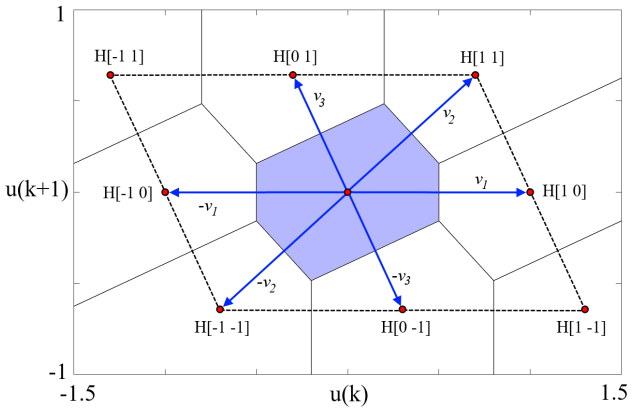

Fig. 2. Voronoi diagram with basic Voronoi cell and relevant vectors.

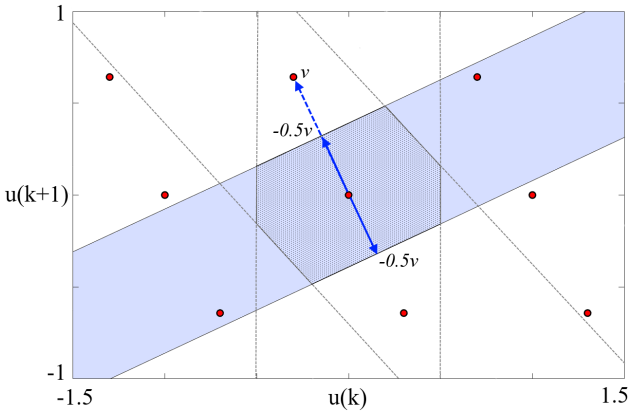

Fig. 3. Slice through the two dimensional H-space.

IV. Micciancio Voulgaris (*MV*) algorithm

A. Algorithm

The *MV* algorithm developed by [7] is based on the *iterative slicing algorithm* proposed by [12], also known as the *slicer* algorithm. The *slicer* finds the closest lattice point $s \in \Lambda$ to a target vector $t \in \mathbb{R}^d$ by iteratively determining which Voronoi cell of the lattice contains the given target. This is achieved by finding the lattice point s with the error vector $e = s - t$ that resides inside the basic Voronoi cell $V(\Lambda, \underline{0})$. Verification of e in $V(\Lambda, \underline{0})$ requires $(2^d - 1)$ *slicing* operations using the set $S(\Lambda)$ of Voronoi relevant vectors. A *slice* through $\mathbb{R}^d$ can be described as the intersection of half spaces constituted by $\pm\frac{1}{2}v$ and defined by $|\langle x \cdot v \rangle| \leq \frac{1}{2}\|v\|^2, x \in \mathbb{R}^d$. Fig 3 demonstrates the slice defined by the relevant Voronoi vector v_3. The intersection of all *slices* yields the basic Voronoi cell in terms of the dot product as

$$V(\Lambda, \underline{0}) = \left\{ x \in \mathbb{R}^d : |\langle x \cdot v \rangle| \leq \frac{1}{2}\|v\|^2, \forall v \in S(\Lambda), \; v \neq \underline{0}. \right\} \quad (11)$$

The slicing algorithm typically starts with the lattice point $s = \underline{0}$ at the origin and iteratively updates s until the resulting error e is found to be between all the slices defined by the set $v \in S(\Lambda)$. If e is found to not reside within a specific slice, then s is updated to $s = s - v$ until $e \in V$ identifies the closest lattice point to t as $s = t + e$.

Unfortunately, the slicing algorithm does not perform any better than previous CVP algorithms, except that it terminates after a finite number of iterations [7]. The main shortfall is the iterative manner in which s is updated with v until a solution is obtained.

A novel selection strategy proposed by [7] identifies the optimum relevant vector v to update s with. This is achieved by scaling the basic Voronoi cell to kV with $k > 1$ and $k \in \mathbb{R}$ so that t is located on a facet/border of V defined by the scaled Voronoi relevant vector kv. The optimal relevant Voronoi vector is selected as the vector with the maximum corresponding k determined from

$$k = max \left| \frac{\langle e \cdot v \rangle}{\|v\|^2} \right|, \forall v \in S(\Lambda), v \neq \underline{0}. \quad (12)$$

Algorithm 1 lists the *Micciancio Voulgaris* CVPP algorithm rewritten in terms of the *Slicer* variables to demonstrate the selection strategy applied. The algorithm requires as inputs

Algorithm 1 *Micciancio Voulgaris* CVPP algorithm

```
1:  function CVPP(t, S)
2:      s ← 0
3:      k ← 1
4:      while k ≠ 0 do
5:          e = s − t
6:          Find v* ∈ S that maximizes (12)
7:          k = rnd(⟨e · v*⟩/‖v*‖²)
8:          s = s − kv*
9:      end while
10:     return s = t + e
```

the target vector $t = HU_{unc}$ and the set of relevant Voronoi vectors S. It is initialized with $s = \underline{0}$ and $k \neq 0$. The error vector is determined by $e = s - t$. Then e is projected orthogonally onto all the relevant Voronoi vectors v and the relevant vector v^* with the maximum k is selected. The rnd function in line 8 slightly differs from the commonly used rounding operation in that the midpoint between consecutive integers is rounded downwards to the integer with smaller absolute value $rnd(0.5) = 0$. Rounding $(\langle e \cdot v_i \rangle / \|v_i\|^2)$ in this manner results in $k = 0$ if $e \in V$ and $k \geq 1$ if $e \notin V$ as per definition of the Voronoi cell in (11). If $k \geq 1$ then s is updated with an integer k of the vector v^*. The iteration is repeated until $k = 0$, in which case the algorithm returns the closest lattice vector to t as $s = t + e$.

B. Hull projection

Similar to the implementation of the *Slicer* algorithm in a single-phase inverter as described in [13], the *MV* algorithm is a CVP solver of a *lattice* structure. The solvers rely on

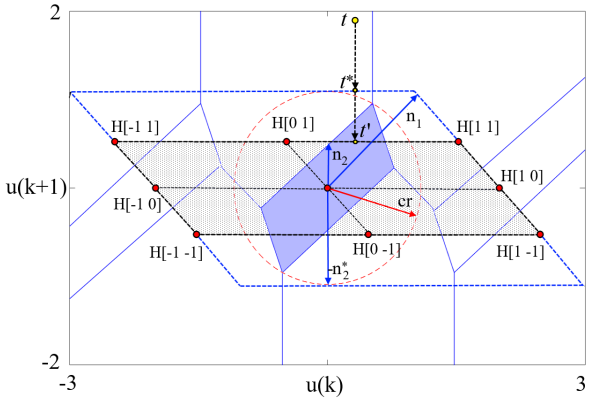

Fig. 4. Extension of the convex hull to a new projection hull that encloses the basic Voronoi cell. Illustrated for a single-phase inverter $N = 2$ and lattice basis with orthogonality defect $\gamma = 1.27$.

the recursive nature of the basic Voronoi cell for defining the Voronoi partition of the lattice. Due to the truncated nature of the lattice in our inverter application $U \in \mathcal{U}$, the only bounded Voronoi cell is the basic cell located at the origin with all other Voronoi cells being partial basic Voronoi cells, which are unbounded. Therefore, in solving the CVP problem, we have to consider two possible scenarios that might occur. One, in which t is inside the convex hull of the parallelotope and another one in which t is located outside the parallelotope. The first situation is the norm under steady-state operation of the inverter and the second usually occurs under transient conditions. In addressing the second scenario, it can be observed that all Voronoi borders exiting the convex parallelotope hull are orthogonal to it. In [7] it is proposed to project the target orthogonally onto the span of the lattice i.e. convex hull, resulting in a point with minimum distance from the target. This projected hull-point $t^{'}$ is then used by the CVP algorithm as an updated target to find the closest lattice point, i.e. the solution to the MPC problem. In [14] is a projection algorithm proposed to achieve a similar result, but in a sphere decoder application. In our case we opted to use an altered iterative projection algorithm.

Implementation of the projection principle in our inverter application resulted in inaccuracies for typical small values of the tuning factor λ. Reducing the switching weight by means of λ increases the *orthogonality defect*

$$\gamma(H) = \frac{\prod_{i=1}^{d} \|h_i\|}{\|det(H)\|}. \tag{13}$$

of the lattice basis. As a result may the basic Voronoi cell extend beyond the span of the truncated lattice. Fig. 4 illustrates such a scenario with $\gamma(H) = 1.27$ for a single-phase inverter with $N = 2$. By example in Fig. 4 it can be observed that the orthogonal projection of t onto the convex hull $P^{'}$ gives an updated point $t^{'}$ for use in finding the closest lattice point. This will result in an error since $t \in V(\Lambda, H\,[01])$ and after projection is $t^{'} \in V(\Lambda, H[00])$. To correct this possible error we introduce an enlarged *projection hull* P^{*} that will

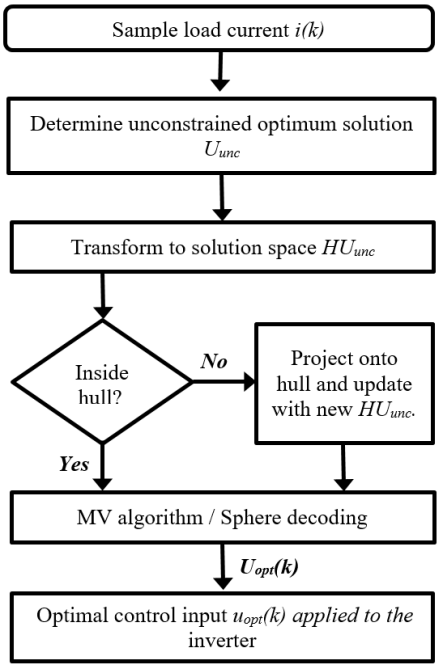

Fig. 5. Flow chart of the Online procedure.

ensure $V(\Lambda, \underline{0}) \subseteq P^{*}$ with all Voronoi borders exiting P^{*} in an orthogonal manner. The normals of P^{*} is determined by utilizing the covering radius cr of the basic Voronoi cell and adjusting the normals of $P^{'}$ with

$$n_i^{*} = \frac{cr}{\|n_i\|} n_i, \forall \|n_i\| \leq cr, i = 1 \ldots d. \tag{14}$$

In Fig. 4 the new *projection hull* is shown with the normal n_2 adjusted to n_2^{*}. Projection of t onto P^{*} gives $t^{*} \in V(\Lambda, H[01])$. Integrity is thus maintained as t^{*} remains in the same Voronoi cell as t.

V. Simulation and Results

Evaluation of the *MV* algorithm was done with a $MATLAB^{©}$ simulation model of the three-phase, three-level NPC inverter and an RL load with the following parameters: sampling interval of $T_S = 25\mu$s, load resistance of $R = 2\Omega$, and load inductance $L = 2m$H. The rated r.m.s. output voltage of the inverter is $V_{AC} = 3.3k$V and the dc-link voltage is $V_{DC} = 5.2k$V. Base quantities were used to establish a per unit system with the reference current amplitude assumed to be 0.8pu. The tuning factor λ was adjusted for every horizon to achieve an average switching frequency of proximately 500Hz to 600Hz. In an attempt to quantify the *MV* algorithm's performance in this specific application, the top-down search of the sphere decoder with the Babai estimate of the initial integer solution was used as reference.

A. Offline

The following pre-processing steps are required for the MV algorithm:

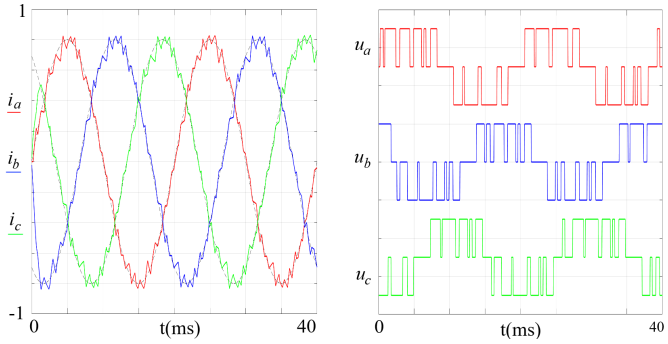

Fig. 6. Three-phase inverter output currents and inverter-leg switch positions.

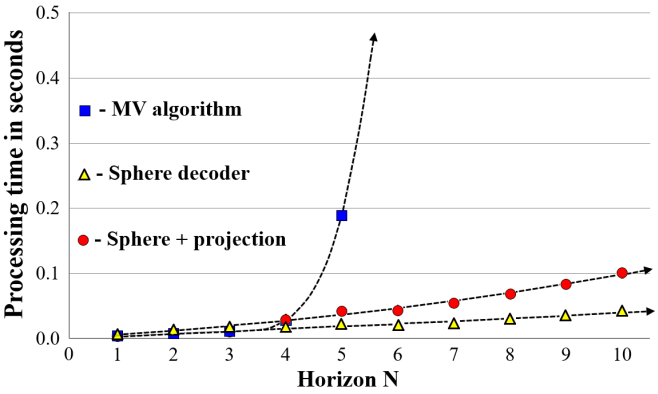

Fig. 7. Processing time required during steady-state operating conditions.

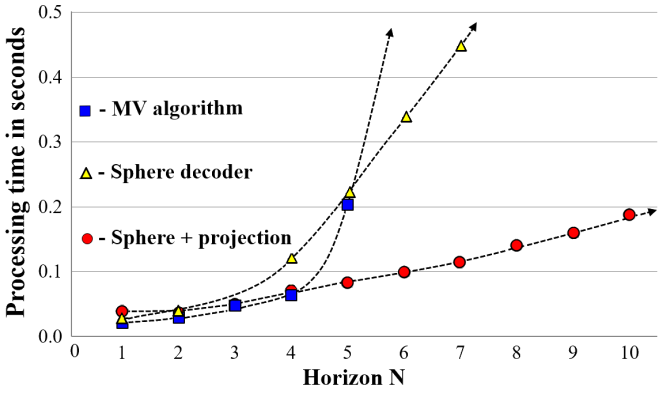

Fig. 8. Processing time required during transient conditions.

1) Calculate H-transformation matrix from the load model parameters.
2) Find the set of Voronoi relevant vectors.
3) From the H-matrix determine the normals of the convex hull $P^{'}$.
4) Determine the covering radius of the basic Voronoi cell.
5) If $V(\Lambda, \underline{0}) \subseteq P^{'}$ fails then adjust the normals with (14) to define the new projection hull P^{*}.

B. Online

The online procedure is illustrated by the flow chart in Fig. 5. Steady-state conditions as well as transient conditions were simulated. Fig. 6 shows typical phase and reference currents with the relevant inverter-leg switch positions. The simulated transient was a step from 0pu to 0.8pu in the reference current as it would occur at the start-up of the inverter. The average processing times required to solve the MPC problems for varying horizons N are depicted respectively in Fig. 7 and Fig. 8 for these conditions. Table I lists the maximum number of iterations required by the respective algorithms to solve the MPC problem during a sampling period. The MV algorithm was only tested to horizon $N = 7$ after which the processing time exceeded any practical consideration.

From Fig. 7 it can be observed that the MV algorithm performs comparatively well up to horizon $N = 4$ during steady-state operation but degrades for longer horizons. During the start up transient condition, the MV algorithm's performance exhibits the same characteristics as during steady-state operation but with an increased processing time constant, which can be attributed to the additional processing required for the projection. From Table I it can be observed that the number of iterations in the MV algorithm for both conditions remained constant at 4. Although this seems favorable, is the computational burden significant at longer horizons due to the $(2^d - 1)$ relevant Voronoi vectors that must be tested in d-dimensional space during each iteration.

Comparatively does the traditional sphere decoder perform fairly well under steady-state conditions but perform worse during transient conditions from horizon $N = 3$ on. In an attempt to provide a fair comparison of the decoding speed of the two algorithms, the projection algorithm was added to the sphere decoder resulting in a system similar to as proposed in [14]. The projected target on the convex hull improved the Babai estimate and reduced the initial sphere radius that facilitated more aggressive pruning by the sphere decoder. A significant reduction in the algorithm iterations can be observed in Table I. This addition to the traditional sphere decoder enabled it to operate efficiently in transient conditions with a limited loss in performance during steady-state operation.

TABLE I
MAXIMUM ITERATIONS REQUIRED BY THE ALGORITHMS TO SOLVE THE MPC PROBLEM DURING STEADY-STATE AND *TRANSIENT OPERATING CONDITIONS.

Horizon N	**3**	**5**	**7**	**9**	**10**
Sphere decoder	108	192	348	783	825
Sphere dec.+proj.	108	213	444	885	1071
MV algorithm	4	4	4		
*Sphere decoder	324	2745	19890	152397	329979
*Sphere dec.+proj.	108	213	444	579	768
*MV algorithm	4	4	4		

VI. Conclusion

The *MV* algorithm utilizes the basic Voronoi cell of a lattice that results from the preprocessing function to solve the CVP. Using a minimal set of Voronoi relevant vectors to describe the Voronoi cell equates to limited storage and offline processing complexities. The online use of the Voronoi relevant vectors facilitates solving the CVP in a short time for horizons $N = 1$ to 4 for a three-phase three-level NPC inverter. The similarity in the performance characteristics during steady-state and transient conditions can be attributed to the nature of the *MV* algorithm itself, which is bound by the $(2^d - 1)$ number of Voronoi relevant vectors during the online search. The improved performance compared to the traditional sphere decoder during transient conditions stems mainly from the hull projection algorithm. The projection onto the hull ensures a new target that resides within a translation of the basic Voronoi cell located around the nearest lattice point $s \in HU$, achieving a reduction of the $s-v$ iterations till $e \in V$. Although the *MV* algorithm seems to be unable to outperform the sphere decoder for long horizons, its computational burden is deterministic and upper bounded [7]. The altered sphere decoder with hull projection showed superior performance in both steady-state and transient conditions, confirming again its authority and reputation as the preferred practical tool for solving long horizon MPC problems.

References

[1] 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.

[2] 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. Orlowska-Kowalska, F. Blaabjerg, and J. Rodrguez, Eds., ed: Springer International Publishing, pp. 181-226, 2014.

[3] Karamanakos, P., Geyer, T., Mouton, T. and Kennel, R., "Computationally efficient sphere decoding for long-horizon direct model predictive control", *In Energy Conversion Congress and Exposition (ECCE)*, 2016 IEEE, pp. 1-8, 2016.

[4] M. Pohst, "On the computation of lattice vectors of minimal length, successive minima and reduced bases with applications", *ACM SIGSAM Bulletin*, vol. 15, pp. 37-44, Feb. 1981.

[5] Agrell, E. ; Eriksson, T. ; Vardy, A. et al., "Closest point search in lattices", *IEEE Transactions on Information Theory*, vol. 48,(8) pp. 2201-2214, 2002.

[6] R. Kannan, "Minkowskis convex body theorem and integer programming", *Mathematics of operation research*, 12(3), pp.415440, Aug. 1987.

[7] Micciancio, D. and Voulgaris, P., "A deterministic single exponential time algorithm for most lattice problems based on Voronoi cell computations", *SIAM Journal on Computing*, vol. 42,(3) pp.1364-1391, 2013.

[8] D. Micciancio, "The hardness of the closest vector problem with preprocessing", *IEEE Trans. Inform. Theory*, vol. 47, pp. 1212-1215, March 2001.

[9] 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.

[10] 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.

[11] Bonifas, N. and Dadush, D., "Short paths on the Voronoi graph and the closest vector problem with preprocessing", *arXiv preprint*, arXiv:1412.6168, Dec 2014.

[12] Sommer, N., Feder, M. and Shalvi, O., "Finding the closest lattice point by iterative slicing", *SIAM Journal on Discrete Mathematics*, vol. 23,(2) pp. 715-731, 2009.

[13] Raath, J., Mouton, H. and Geyer, T., "Iterative slicing as solution to the model predictive control problem of a three-level inverter", *In Proceedings of the 25th South African Universities Power Engineering conference (SAUPEC)*, Stellenbosch, pp. 75-80, February 2017.

[14] Baidya, R., Aguilera, R.P., Acuna, P., Delgado, R., Geyer, T., Quevedo, D. and Mouton, T., "Fast multistep finite control set model predictive control for transient operation of power converters.", *In Industrial Electronics Society, IECON 2016-42nd Annual Conference of the IEEE*, pp. 5039-5045, October 2016.