ANÁLISE DE UM SETOR DO DIAGRAMA SPACE VECTOR DE 3 NÍVEIS.

TIARLES GUTERRES[1]

SUMÁRIO

LISTA DE FIGURAS

[1] *Grupo de Eletrônica de Potência e Controle (GEPOC), UFSM, Santa Maria, Brasil*

1 OBJETIVOS

O objetivo deste trabalho é a partir da escolha de um setor considerando o diagrama *Space Vector* três níveis e o conversor NPC trifásico:

(a) nomear cada um dos vetores pertencentes ao setor,

(b) identificar a associação destes vetores ao estado de comutação das chaves de cada fase do conversor,

(c) definir como cada um destes vetores contribui para a corrente no ponto neutro central, que é ligado ao divisor capacitivo do barramento CC que alimenta o conversor,

(d) escrever uma rotina que calcule os tempos chamados de *dwell times*,

(e) desenhar o diagrama de uma comutação,

(f) definir uma sequência de comutação e desenha-lá com os tempos para cada vetor e, por fim,

(g) descrever como seria a implementação destes vetores (nível ativo alto ou baixo e os valores em cada comparador)

2 DESENVOLVIMENTO

O setor escolhido foi chamado de *Setor 'S'* (Figura 1).

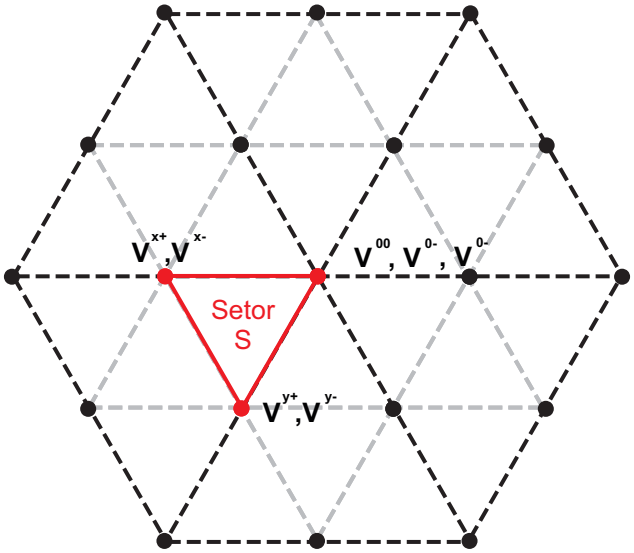


Figura 1: Diagrama SVM para conversores 3 níveis com detalhe para o *setor 'S'* e seus vetores.

VETORES DO *setor 's'*. Os seus vetores foram chamados de V^{0-}, V^{00}, V^{0+}, V^{x-}, V^{x+}, V^{y-} e V^{y+}. A equação (1) mostra a relação destes vetores com o estado de comutação das fases do conversor. A Tabela 1 mostra o que cada valor usado na representação dos vetores da equação (1) significa para o conjunto de cada fase do conversor trifásico.

$$V^{0-} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} ; V^{00} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} ; V^{0+} = \begin{bmatrix} 2 \\ 2 \\ 2 \end{bmatrix} ; V^{x-} = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} ; V^{x+} = \begin{bmatrix} 1 \\ 2 \\ 2 \end{bmatrix} ; V^{y-} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} ; V^{y+} = \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix}. \quad (1)$$

Tabela 1: Tabela que mostra o estado de condução e o estado real (ativo ou não) das chaves para cada fase do conversor.

Estado de condução	S1	S2	S3	S4
0	0	0	1	1
1	0	1	1	0
2	1	1	0	0

CORRENTE NO PONTO CENTRAL DO NPC. O estado de condução 1 faz com que a fase do conversor construa um caminho para a corrente que se conecte no ponto central do conversor, compartilhado pelas fases e onde se encontra o divisor capacito. Dependendo do estado das outras fases do conversor esta corrente ainda pode circular pelo conversor, como quando duas ou mais fases (pensando em conversores multifásicos) estão neste mesmo estado, entretanto quanto apenas uma apresenta-se neste estado de condução a corrente jogada ou extraída no ponto central irá desbalancear de forma mais agressiva os capacitores conectados.

Os vetores V^{0-}, V^{00}, V^{0+} se anulam e o único que apresenta este estado de condução, inclusive em todas as suas fases é o vetor V^{00}. Considerando que todas as fases estão compartilhando este mesmo ponto, o ponto neutro, as correntes podem circular livremente por todas das fases do conversor, o desbalanceamento no divisor capacitivo vai depender de como as correntes saem ou entram na carga em que o conversor está conectado. O mesmo vale para os vetores V^{x-} e V^{y+} que possuem este estado de condução em duas fases.

Os vetores V^{x+}, V^{y-} apresentam apenas uma das fases conectadas ao ponto central, com a sintetização do chaveamento utilizando estes vetores o desbalanceamento do divisor de capacito será bem mais significativo.

CÁLCULO DOS *dwell times*. A partir dos vetores em coordenadas alfa-beta podemos contruir as matrizes para o cálculos dos *dwell times*. A equação (2) mostra os vetores V^0, V^x e V^y em coordenadas alfa-beta.

$$V^0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix};\ V^x = \begin{bmatrix} -\frac{2}{3} \\ 0 \end{bmatrix};\ V^y = \begin{bmatrix} -\frac{1}{3} \\ -\frac{\sqrt{3}}{3} \end{bmatrix}. \tag{2}$$

Considerando que o nosso objetivo é V_{ref} a combinação linear para obtê-lô é:

$$V_{ref} = \frac{\Delta t_0 V^0 + \Delta t_x V^x + \Delta t_y V^y}{T_s}, \tag{3}$$

e isolando o período total T_s:

$$T_s = \Delta t_0 + \Delta t_x + \Delta t_y. \tag{4}$$

Com estas as equações (3) e (4) podemos descrever o seguinte sistema linear:

$$\begin{bmatrix} \Delta t_0 \\ \Delta t_x \\ \Delta t_y \end{bmatrix} \times \begin{bmatrix} V^0 & V^x & V^y \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} V_{ref} T_s \\ Ts \end{bmatrix} \text{ ou}$$

$$\begin{bmatrix} \Delta t_0 \\ \Delta t_x \\ \Delta t_y \end{bmatrix} = \begin{bmatrix} V^0 & V^x & V^y \\ 1 & 1 & 1 \end{bmatrix}^{-1} \times \begin{bmatrix} V_{ref} T_s \\ Ts \end{bmatrix},$$

tornando assim a matriz já invertida igual à

$$\begin{bmatrix} \Delta t_0 \\ \Delta t_x \\ \Delta t_y \end{bmatrix} = \begin{bmatrix} \frac{3}{2} & \frac{\sqrt{3}}{2} & 1 \\ -\frac{3}{2} & \frac{\sqrt{3}}{2} & 0 \\ 0 & -\sqrt{3} & 0 \end{bmatrix} \times \begin{bmatrix} V_{ref,\alpha} T_s \\ V_{ref,\beta} T_s \\ \Delta Ts \end{bmatrix}, \tag{5}$$

como mostra o código da Figura 2.

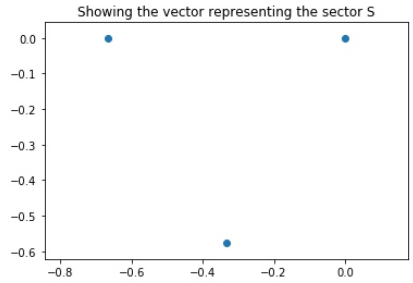
Dwell Times calculation to Sector S

In [2]:
```
V0 = np.array([0,0,0])
Vx = np.array([0,1,1])
Vy = np.array([0,0,1])

T = 2/3 * np.array([[1, -1/2, -1/2],
                    [0, np.sqrt(3)/2, -np.sqrt(3)/2]])
```

In [3]:
```
V0_ab = T @ V0
Vx_ab = T @ Vx
Vy_ab = T @ Vy

aux = np.array([V0_ab, Vx_ab, Vy_ab]); aux
```

Out[3]:
```
array([[ 0.        ,  0.        ],
       [-0.66666667,  0.        ],
       [-0.33333333, -0.57735027]])
```

In [4]:
```
plt.title('Showing the vector representing the sector S')
plt.scatter(aux[:, 0], aux[:, 1])
plt.axis('equal')
plt.show()
```



In [7]:
```
dwell_times_S_1 = np.array([V0_ab, Vx_ab, Vy_ab]).transpose(); print(dwell_times_S_1)
dwell_times_S_2 = np.ones(3); print(dwell_times_S_2)
```

```
[[ 0.         -0.66666667 -0.33333333]
 [ 0.          0.         -0.57735027]]
[1. 1. 1.]
```

In [10]:
```
dwell_times_S = np.vstack([dwell_times_S_1, dwell_times_S_2]); dwell_times_S
```

Out[10]:
```
array([[ 0.        , -0.66666667, -0.33333333],
       [ 0.        ,  0.        , -0.57735027],
       [ 1.        ,  1.        ,  1.        ]])
```

In [11]:
```
from numpy.linalg import inv
inv(dwell_times_S)
```

Out[11]:
```
array([[ 1.5       ,  0.8660254 ,  1.        ],
       [-1.5       ,  0.8660254 , -0.        ],
       [-0.        , -1.73205081, -0.        ]])
```

In [ ]:

Figura 2: Código em Python para o cálculo dos *dwell times*.

A Figura 2 mostra primeiro a seleção de três vetores para acomodá-los no referencial alfa-beta a partir da transformação com a *matriz 'T'* mostrada na figura e a representação no espaço a partir de um gráfico de pontos. Após, a matriz para o cálculo de *dwell times* é construída com outras duas, uma com a matriz de vetores, já chamados de V^0, V^x e V^y, transpostos e com um vetor de '1s' que foram 'empilhados' verticalmente para a criação da matriz *'dwell_times_S'*, a mesma mostrada na equação (5).

SEQUÊNCIA DE COMUTAÇÃO E IMPLEMENTAÇÃO DO SV MODULATION PARA O NPC. A Figura 3 mostra o projeto e implementação da sequência de comutação para o setor S.

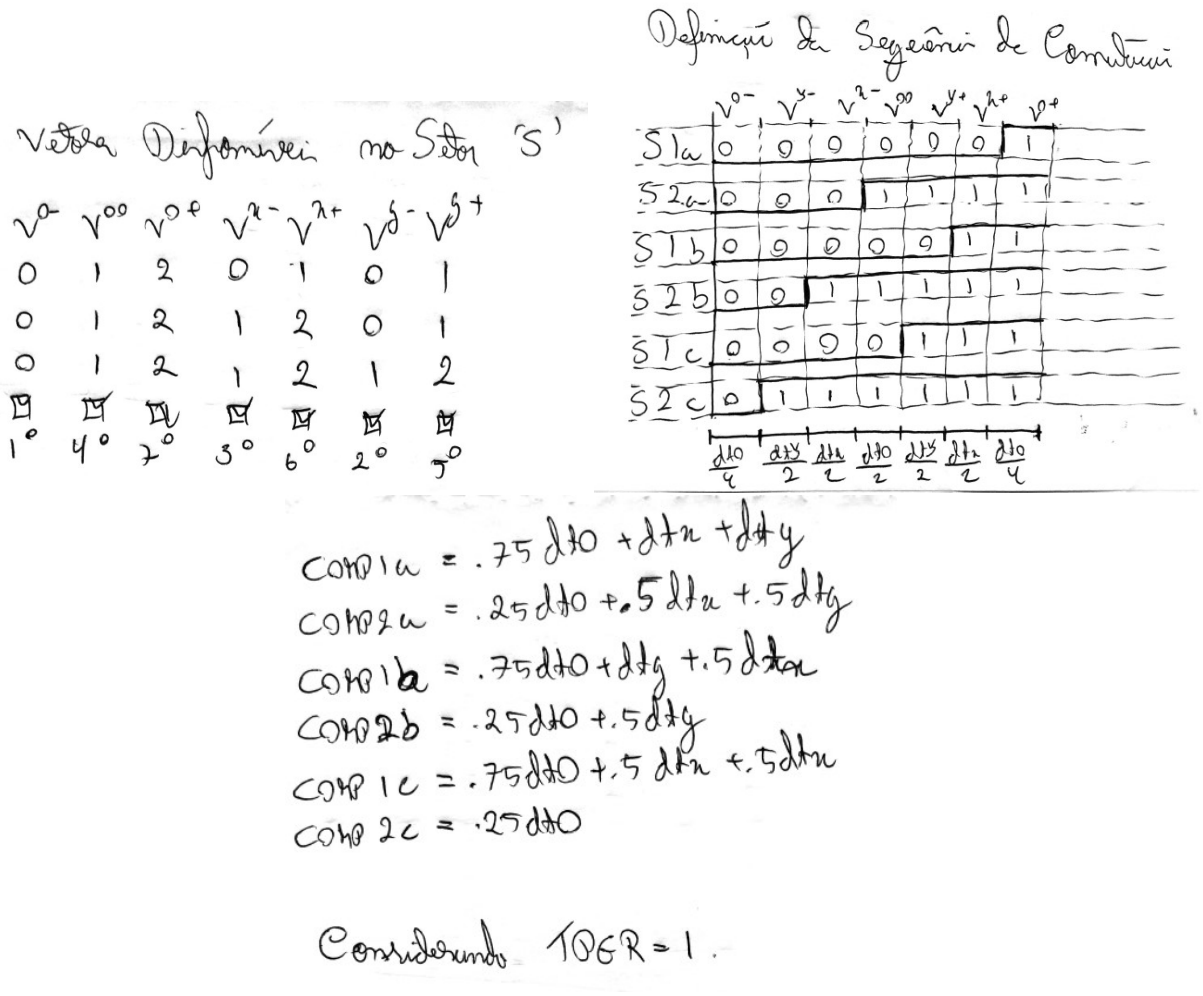


Figura 3: Sequência de comutação, tempos de cada vetor e implementação para o *setor 'S'*.

As figuras apresentam a finalização do projeto dos vetores para o Setor S, o tempo de cada para comutação, a sequência que eles se sucedem e a implementação com o conteúdo dos comparadores, nela chamados de COMP1a, COMP2a, COMP1b, COMP2b, COMP1c e COMP2c. Para o exemplo desenvolvido considera-se ainda que o nível PWM é ativo em nível baixo.