A

Project Report

on

Model Predictive Control for Power Converters and Drives

Submitted in fulfilment of the requirements for the award of degree of

Bachelor of Technology

in

Electric Drives and Control

Submitted by:-

Yashi Baldaniya (2015UEE1440)

Sunayana Gupta (2015UEE1754)

Supervised by:-

Dr. Rajesh Kumar

Associate Professor

Department of Electrical Engineering

Malaviya National Institute of Technology, Jaipur

May, 2018

CONTENTS

ACKNOWLEDGEMENTS

We take this opportunity to express our deep sense of gratitude to our supervisor **Dr. Rajesh Kumar**, Associate professor, Department of Electrical Engineering , Malaviya National Institute of Technology Jaipur, for his valuable guidance and cooperation throughout the project work. He provided constant encouragement and unceasing enthusiasm at every stage of the project work.

It gives us immense pleasure in acknowledging the encouragement showered on me by my PHD scholars. Their suggestions and moral support helped us to accomplish our work.

Date : May 4, 2018 **(Yashi)**
Place: MNIT Jaipur **(Sunayana)**

ABSTRACT

The aim of this project is to present implementation of MPC for drives. We have implemented a finite control set model predictive torque control of induction machine. The model consists of 3 main blocks: inverter, induction machine and MPC algorithm. MPC is used for fast settling of torque and speed and reduction of the ripples present in it. MPC for power converters and drives can be considered as well-established technology for prediction.

1.Introduction

Model predictive control (MPC) has been a topic of research and development for more than three decades. Originally, it was introduced in the process industry, but a very innovative and early paper proposed that predictive control be used in power electronics [1]. In the recent years, it has been proposed and studied as a promising alternative for the control of power converters and drives.It presents a fast dynamic response and allows for nonlinearities and constraints to be incorporated into the control law in a straightforward manner, and can incorporate nested control loops in only one loop.An analysis of MPC algorithms when applied t o power converters and drives reveals that the key elements for any MPC strategy are the prediction model, cost function, and optimization algorithm.

1.1 Literature Review

Recently, many research efforts have been focused on MPC, which reach good performance. MPC method in the field of electrical drives can be divided into two main categories:

1) Continuous MPC and
2) Finite-state MPC.

Generalized predictive control (GPC) can be considered as an example of continuous MPC method. They reach good performance but the implementation of the algorithm needs a pulse width modulator (PWM). Finite-state MPC considers the applied inverter model in the control algorithm. The controller directly calculates all switching states and finds the best state which minimizes the cost function. The switching signals are directly output to the inverter without a modulator, which leads to simplicity. Finite-state MPC has already been applied and researched widely.

In industrial situations such as traction and steel industry, torque control is very important. Direct torque control(DTC) method is known for its fast dynamics, but it has a main drawback: big torque ripples. Many strategies are researched to solve this problem. Among these methods, finite control set predictive torque control (PTC) has been approved to be a good alternative method. PTC is an important branch of finite-state MPC. PTC reduces the torque ripples while keeping the merits of DTC: fast dynamics and straightforward algorithm. Moreover, PTC can handle well with over-current protections.

1.2 MODEL PREDICTIVE CONTROL: OPERATING PRINCIPLE

In general, MPC defines the control action by minimizing a cost function that describes the desired system behavior. This cost function compares the predicted system output with a reference. The predicted outputs are computed from the system model. In general, for each sampling time, the MPC controller calculates a control action sequence that minimizes the cost function, but only the first element of this sequence is applied to the system. Although MPC controllers solve an open-loop optimal control problem, the MPC algorithm is repeated in a receding horizon fashion at every sampling time, thus, providing a feedback loop and potential robustness with respect to system uncertainties.

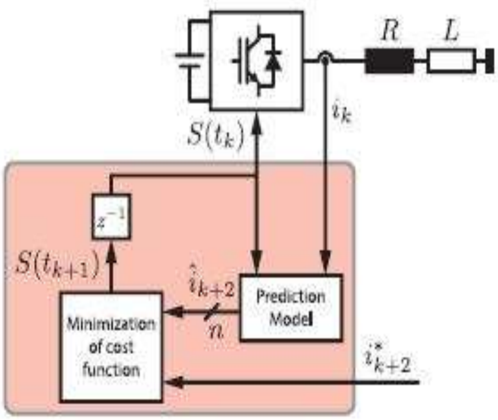
To illustrate the use of MPC for power electronics, a basic MPC strategy with a prediction horizon equal to one, applied to the current control of a voltage source inverter (VSI) with output RL load, is shown. The basic block diagram of this control strategy is presented in Fig. 1, where the reference and predicted currents at instant k + 2 are used in order to compensate for the digital implementation delay. The algorithm is repeated for each sampling time and performs the following steps:

1) Optimal control action S(tk) computed at instant k − 1 is applied to the converter.

2) Measurement of the current i(k), si(k) is taken at instant k. The reference current i(k+1), si(k+1) and T(k+1) is predicted.

3) Cost function is evaluated using T(k+1) and si(k+1). The optimal control action S(tk+1) to be applied at instant k + 1 is chosen as the one that minimizes the cost function's value.

S(tk) are the firing pulses for the power switches, these values are constant from instant k to k + 1.



1.3 Cost Function

The cost function in the MPC strategy defines the desired system behavior. For this purpose, it compares the predicted and reference values. The cost function can have any form, but in general, it can be written as

MPC strategies solve an optimization problem in order to
define the control signal to be applied to the system. The cost
function represents the desired behavior for the system. Therefore, MPC calculates the optimal actuation by minimizing it

It can be observed that current, voltage, torque, power, and other control objectives are considered for the cost function.

Some cost functions found in the literature with their application are given by:

Application		Cost function
CSC-AFE	[23]	$g = \|q\| + \lambda\|\hat{i}_L - i_L^*\|$
	[24]	$g = (q)^2 + \lambda\left(\hat{i}_L - i_L^*\right)^2$
	[17]	$g = \|\hat{i}_k - i_k^*\|$
VSC-AFE	[88]	$g = \|\hat{i}_k - i_k^*\| + \lambda_n n_c$
	[89]	$g = \left(\hat{i}_k - i_k^*\right)^2$
	[20]	$g = \left(\hat{P} - P^*\right)^2 + \left(\hat{Q} - Q^*\right)^2$
Motor drive	[36]	$g = \left(\hat{T} - T^*\right)^2 + \lambda\left(\hat{\psi} - \psi^*\right)^2$
VSC-UPS	[39]	$g = \left(\hat{v}_o - v_o^*\right)^2$
Statcom	[50]	$g = \left(\hat{i}_k - i_k^*\right)^2$
Matrix converter	[54]	$g = \|\hat{i}_L - i_L^*\| + \lambda\|\hat{Q} - Q^*\|$
	[65]	$g = \left(\hat{i}_L - i_L^*\right)^2 + \lambda\left(\hat{Q} - Q^*\right)^2$
HVDC	[69]	$g = g_1 + g_2 + g_3$ $g_1 = \|\hat{i}_{jk} - i_{jk}^*\|$ $g_2 = \lambda_{Ck}\sum_i \|\hat{V}_{cijk} - \frac{V_{dc}}{n}\|$ $g_3 = \lambda_{zk}\|\hat{i}_{zjk}\|$

The cost function used (for IM) in our project is given by:

$$g_j = \sum_{h=1}^{N}\Big\{\lambda_1 \cdot |T^* - \hat{T}(k+h)_j| + \lambda_2 \cdot |||\boldsymbol{\psi}_{\boldsymbol{s}}^*|| - ||\hat{\boldsymbol{\psi}}_s(k+h)_j||| + \lambda_3 \cdot |\boldsymbol{S}(k) - \boldsymbol{S}(k+h)_j|\Big\}$$

Where *j* denotes the index of applied voltage vector for the prediction, it is $j = 0, \ldots, 6$. The *h* represents the prediction horizon. In this work, only one prediction step is considered ($h = 1$). *T* denotes the electromagnetic torque. ***Ψs*** donates the stator flux. ***S*** denotes the switching vector and ***S*** = 000, 001, 010, . . . , 111; ***S***(*k*) and ***S***(*k* + 1) denote the current switching vector and the predicted switching vector, respectively. Parameters with the superscript " ˆ " refer to the estimated values. The torque reference $T*$ is generated by an external speed PI controller. The $\lambda 1,2,3$ denote the weighting factors, which weight the relative importance of each term. Here, $\lambda 1 = 1$, $\lambda 2 = 7.5$, $\lambda 3 = 0$.

1.3.1 Weighing Factor

MPC can handle several control objectives simultaneously. In order to do so, the variables to be controlled should be included in the cost function. As a result, the cost function can contain variables of differing natures.

In general, the differing natures of the variables hinder the selection of the weighting factors. This is because these variables usually have different orders of magnitude. Therefore, they do not equally contribute to the cost function's value.

The weighting factor values have a direct influence on the system's performance. It is not easy to define the suitable weighting factor values to achieve a desired system behavior. Usually, the procedure consists in a heuristic approach.

1.4 Prediction

In MPC algorithm, the stator flux $\hat{\psi}_s(k+1)$ and the electromagnetic torque $\hat{T}(k+1)$ must be predicted. In this work, the forward Euler discrete equation is considered. According to the IM model, the stator current, the stator flux, and the electromagnetic torque can be predicted as follows:

$$\hat{\boldsymbol{i}}_s(k+1) = \left(\frac{T_{\text{ptc}}R_r}{\sigma L_s L_r} - j\frac{T_{\text{ptc}}\hat{\omega}}{\sigma L_s}\right)\hat{\boldsymbol{\psi}}_s(k) + \left(1 - \frac{T_{\text{ptc}}R_s}{\sigma L_s} - \frac{T_{\text{ptc}}R_r}{\sigma L_r} + jT_{ptc}\hat{\omega}\right)\boldsymbol{i}_s(k) + \frac{T_{\text{ptc}}}{\sigma L_s}\boldsymbol{v}_s \quad (6)$$

$$\hat{\boldsymbol{\psi}}_s(k+1) = \hat{\boldsymbol{\psi}}_s(k) + T_{\text{ptc}}\boldsymbol{v}_s(k) - T_{\text{ptc}}R_s\boldsymbol{i}_s(k) \quad (7)$$

$$\hat{T}(k+1) = \frac{3}{2}\,p \cdot Im\{\hat{\boldsymbol{\psi}}_s(k+1)^*\ \hat{\boldsymbol{i}}_s(k+1)\} \quad (8)$$

where $\sigma = 1 - (L_m^2/L_sL_r)$. T_{ptc} is the sampling time, $\boldsymbol{v}_s$ represents the stator voltage vector, and $\boldsymbol{i}_s$ is the stator current. R_s and R_r are the stator and rotor resistances. ω is the electrical speed, and *Ls*, *Lr*, and *Lm* are the stator, rotor, and mutual inductance, correspondingly. *p* refers to the number of pole pairs.

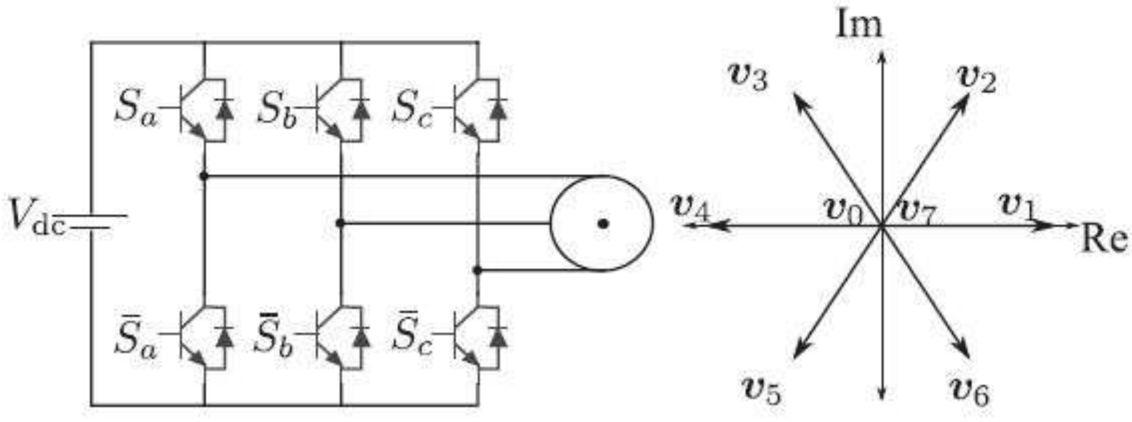
2. Major Components used in Model

2.1 Inverter

The multilevel inverters have always been as a preferred power converter topology for high voltage and high power applications. One approach uses predictive control to calculate the necessary states to optimize the torque and flux behavior. Later, the inverter is used to generate this desired voltage. This strategy has been used in current control for inverters as well as for rectifiers and active filters. One advantage of predictive control is the possibility to include nonlinearities of the system in the predictive model, and hence calculate the behavior of the variables for different conduction states. The model of the system is used to predict the behavior of the load current, torque and flux for each different switching state of the matrix converter. The switching state that minimizes a quality function is selected. This method demonstrates that the use of predictive control can avoid the use of complex modulation techniques.

In the proposed technique, only six active voltage vectors are used in the predictive model, and the vectors are selected based on the position of the future reference vector. In every sampling period, the position of the reference current is used to detect the voltage vectors surrounding the reference voltage vector. Besides the six active vectors, one of the zero vectors is also used.

As the FCS-MPC considers a finite number of valid switching states to predict the behavior of the system through a discrete model in every sampling period. The FCS-MPC uses a cost function to carry out the optimization in the prediction. The predefined cost function is used to compare each prediction with its respective reference, and the switching state that produces the minimum value of the cost function is applied to the inverter. This process is repeated in every sampling period as mentioned in, and thus, no modulation stage is required in this technique. The implementation of the FCS-MPC is very easy and simple, and it has fast a dynamic response in spite of its constraints and nonlinearity inclusion.



2.2 Induction Machine

All analysis and simulation in are based on the d-q or dynamic equivalent circuit of the induction motor.
The differential equations produced from analysis of the circuits in are as follows:

$$v_{ds} = R_s i_{ds} + \frac{d\lambda_{ds}}{dt} - \omega_e \lambda_{qs},$$

$$v_{qs} = R_s i_{qs} + \frac{d\lambda_{qs}}{dt} + \omega_e \lambda_{ds},$$

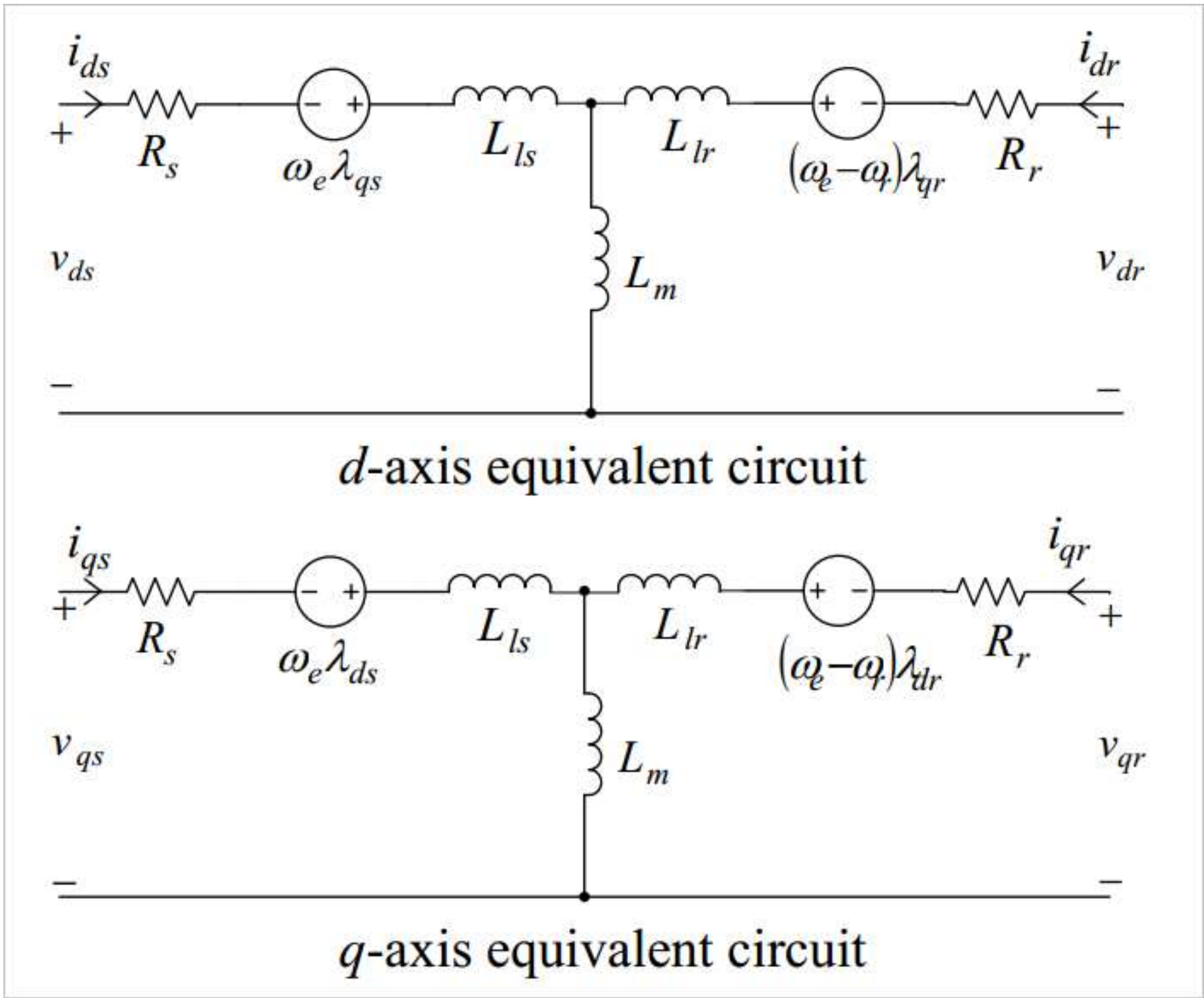
$$v_{dr} = 0 = R_r i_{dr} + \frac{d\lambda_{dr}}{dt} - (\omega_e - \omega_r)\lambda_{qr},$$

and

$$v_{qr} = 0 = R_r i_{qr} + \frac{d\lambda_{qr}}{dt} + (\omega_e - \omega_r)\lambda_{dr}$$

where *d* is the direct axis, *q* is the quadrature axis, *vds* is the *d*axis stator voltage, *vqs* is the q-axis stator voltage, *vdr* is *d*-axis rotor voltage, *vqr* is *q*-axis rotor voltage, *ids* is the *d*-axis stator current, *iqs* is the *q*-axis stator current, *idr* is *d*-axis rotor current, *iqr* is *q*-axis rotor current, *Rs* is the stator resistance, *Rr* is the rotor resistance, *ωe* is the angular velocity of the reference frame, *ωr* is the angular velocity of the rotor, and
λds , *λds*, *λds*, and *λds* are flux linkages.

D-q equivalent circuits of the Induction Machine can be given as:



The currents can be expressed in terms of flux linkages as:

$$i_{ds} = \frac{L_r}{L_r L_s - L_m^2}\lambda_{ds} - \frac{L_m}{L_r L_s - L_m^2}\lambda_{dr},$$

$$i_{qs} = \frac{L_r}{L_r L_s - L_m^2}\lambda_{qs} - \frac{L_m}{L_r L_s - L_m^2}\lambda_{qr},$$

$$i_{dr} = \frac{L_s}{L_r L_s - L_m^2}\lambda_{dr} - \frac{L_m}{L_r L_s - L_m^2}\lambda_{ds},$$

and

$$i_{qr} = \frac{L_s}{L_r L_s - L_m^2}\lambda_{qr} - \frac{L_m}{L_r L_s - L_m^2}\lambda_{qs}.$$

where *Lr* is the rotor self inductance, *Ls* is the stator self inductance, *Lm* is the magnetizing inductance.

The differential equations of flux linkages used in the model are as follows

$$\frac{d\lambda_{ds}}{dt} = v_{ds} - \frac{R_s L_r}{L_r L_s - L_m^2}\lambda_{ds} + \frac{R_s L_m}{L_r L_s - L_m^2}\lambda_{dr} + \omega_e \lambda_{qs} ,$$

$$\frac{d\lambda_{qs}}{dt} = v_{qs} - \frac{R_s L_r}{L_r L_s - L_m^2}\lambda_{qs} + \frac{R_s L_m}{L_r L_s - L_m^2}\lambda_{qr} - \omega_e \lambda_{ds} ,$$

$$\frac{d\lambda_{dr}}{dt} = \frac{-R_r L_s}{L_r L_s - L_m^2}\lambda_{dr} + \frac{R_r L_m}{L_r L_s - L_m^2}\lambda_{ds} + (\omega_e - \omega_r)\lambda_{qr} ,$$

and

$$\frac{d\lambda_{qr}}{dt} = \frac{-R_r L_s}{L_r L_s - L_m^2}\lambda_{qr} + \frac{R_r L_m}{L_r L_s - L_m^2}\lambda_{qs} - (\omega_e - \omega_r)\lambda_{dr} .$$

The electromagnetic torque of the IM is given by-

$$T_e = \frac{3}{2}\frac{P}{2} L_m \left[i_{qs} i_{dr} - i_{ds} i_{qr} \right]$$

where P is the number of pair of poles.

The torque and rotor speed are related by:

$$\frac{d\omega_r}{dt} = \frac{P}{2J}\left(T_e - T_L\right)$$

where *TL* is the load torque and *J* is the inertia of the rotor and connected load.

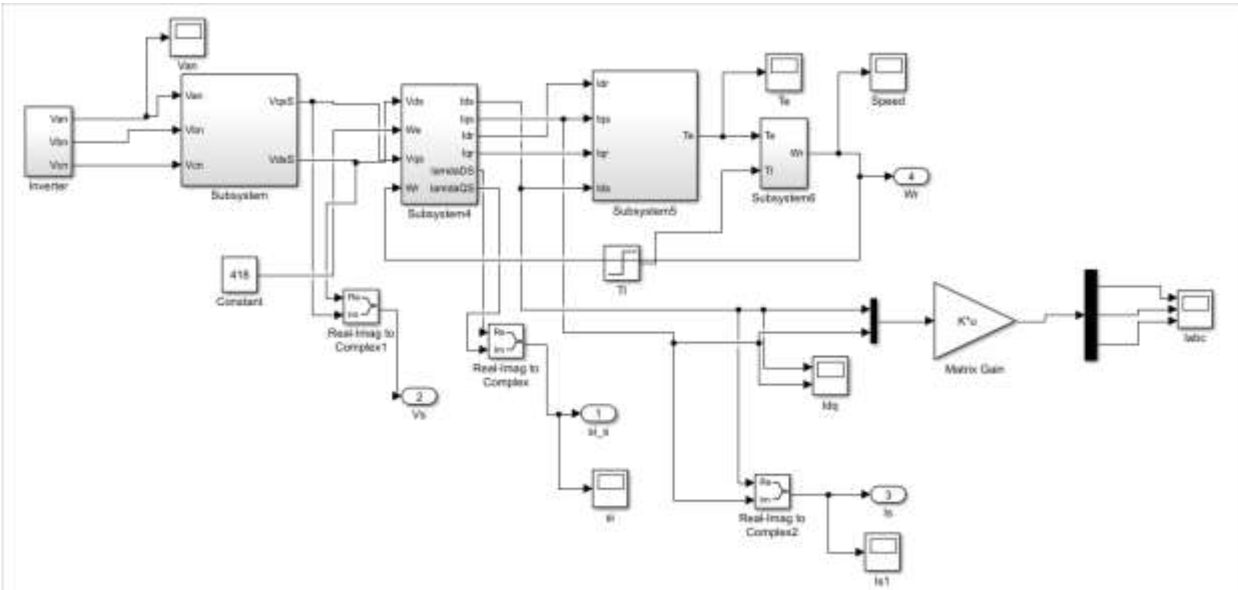
Three-phase voltages can be converted to the two-phase stationary frame using the following relationship:

$$\begin{bmatrix} v_{qs}^{s} \\ v_{ds}^{s} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{-1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \end{bmatrix} \begin{bmatrix} v_{an} \\ v_{bn} \\ v_{cn} \end{bmatrix}$$

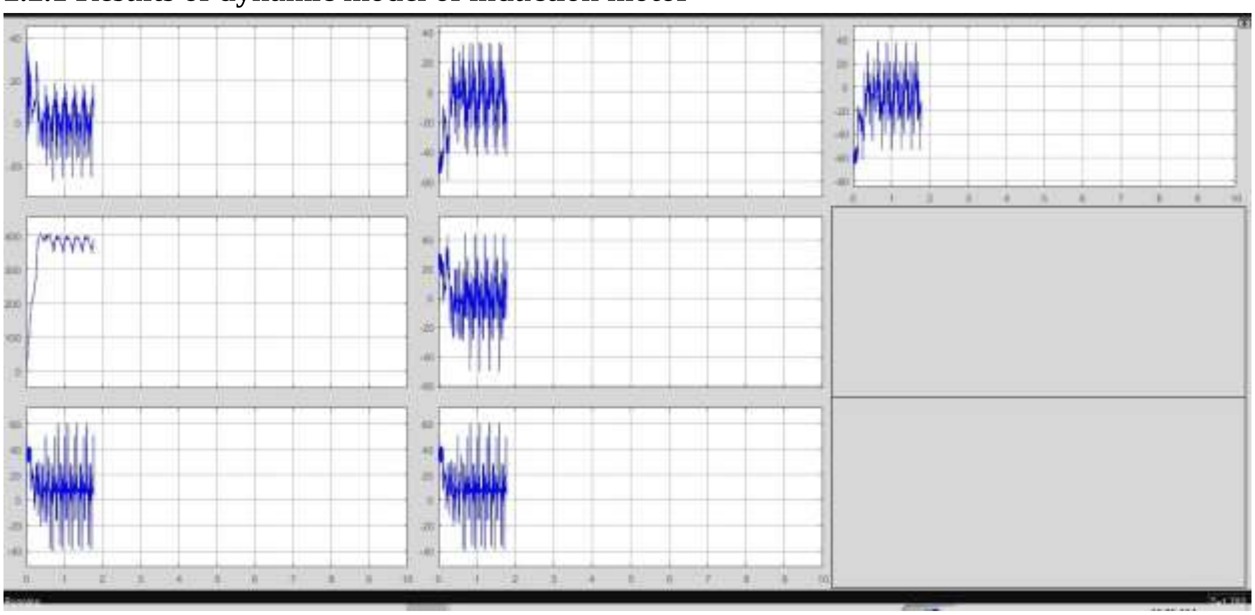
Also, the two phase current is converted to three phase current by using the following transformation:

$$\begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \frac{-1}{2} & \frac{-\sqrt{3}}{2} \\ \frac{-1}{2} & \frac{\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} i_{qs}^{s} \\ i_{ds}^{s} \end{bmatrix}.$$

The above equations are implemented in Simulink using the basic MATLAB blocks and subsystems are formed which are joined to make a dynamic model of IM. The Simulink model of IM+Inverter used in our project is as shown:



2.2.1 Results of dynamic model of induction motor



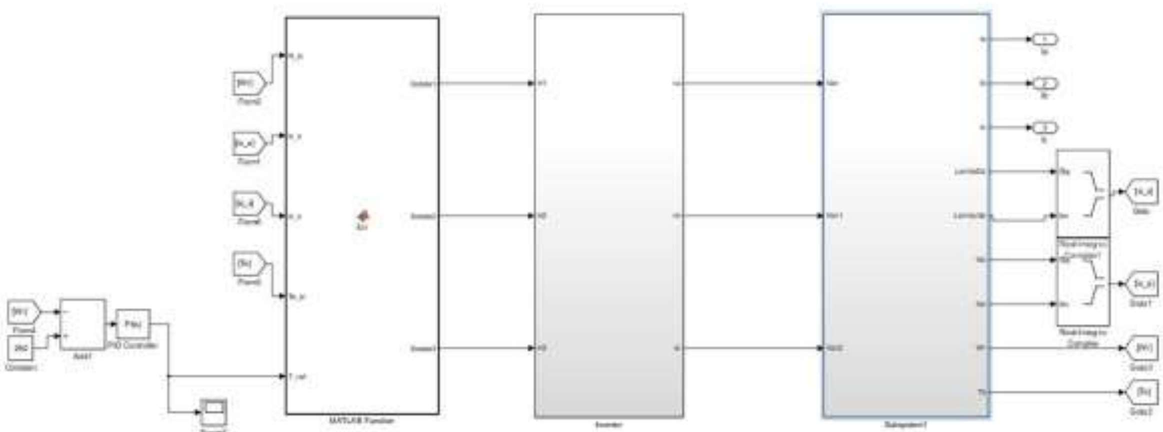
1st column consists of the results of torque, speed,Ids
2nd column consists of the results of Iqs, Ia,Ib
3rd column consists of the results of Ic

3.Model Predictive Control

We have implemented the algorithm of Model Predictive Control using two strategies which are explained below in detail with their results:

3.1.1 Strategy 1: Matlab function within the simulation

In this technique we have used a MATLAB function which firstly predicts the torque, stator current and flux for the next sampling instant and then uses these values to computes the cost function of all the 7 states of the inverter and the state which minimizes the cost function is selected and given to the inverter. We have taken the sampling time slightly above the run time of the matlab function for a single sampling instant to ensure that whole of the program runs at a time. And we have chosen the fixed step as a factor of this sampling time so that all the sampling instants are visible to the machine.



Matlab function:

```
function [Sstate1,    Sstate2,   Sstate3]     = fcn(N_pi, is_a, si_s, Te_pi,T_ref)
%#codegen

S=[0 0 0;0 0 1; 0 1 0;0 1 1;1 0 0;1 0 1;1 1 0;1 1 1];
```

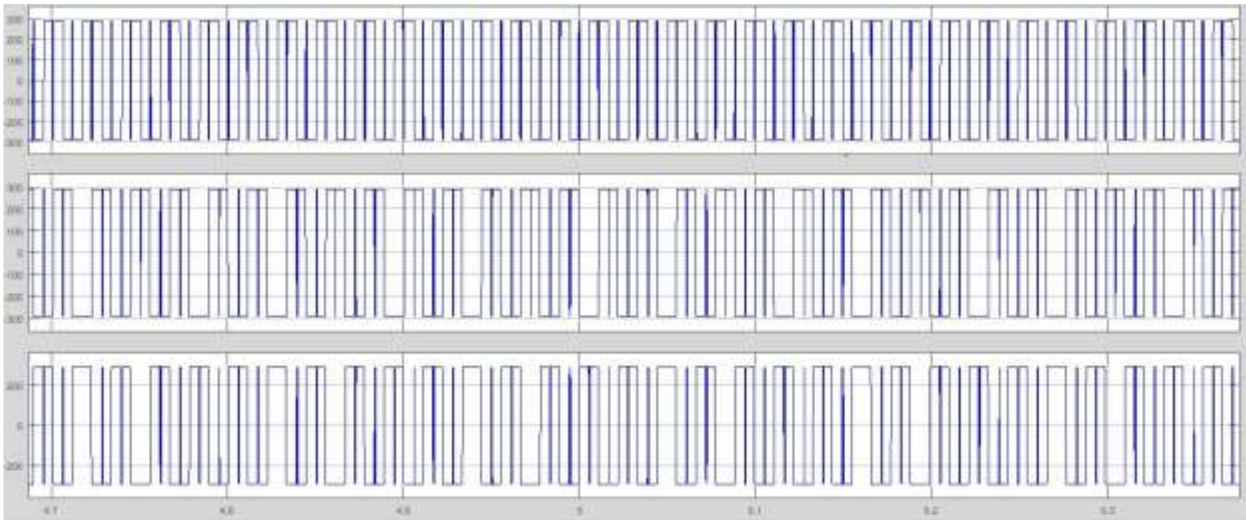
```
k=1;
gj=0;
Vs=[0+0j 0+0j 0+0j 0+0j 0+0j 0+0j 0+0j 0+0j];
is_a1_k1 = [0+0j 0+0j 0+0j 0+0j 0+0j 0+0j 0+0j 0+0j];
si_s1_k1 = [0+0j 0+0j 0+0j 0+0j 0+0j 0+0j 0+0j 0+0j];
Te_pi1_k1 =[0+0j 0+0j 0+0j 0+0j 0+0j 0+0j 0+0j 0+0j];
S_state=[0 0 0];

is_a1 = is_a;
N_pi1 = N_pi;
si_s1 = si_s;

Te_pi1=Te_pi;
Vdc=582;
Rs=2.68;
Rr=2.13;
Lm=0.275;
Ls=0.2834;
Lr=0.2834;
P=1;
N_pi_nom=290;
J=0.005;
ts=0.005;

a=-inf;

sigma=1-(Lm^2)/Ls*Lr;


for i = 1:8

    if S(i,1)==0
        Van=-Vdc/2;
    else
        Van=Vdc/2;
    end
    if S(i,2)==0
     Vbn=-Vdc/2;
    else
        Vbn=Vdc/2;
```

```
end
if S(i,3)==0
  Vcn=-Vdc/2;
else
   Vcn=Vdc/2;
end


Vs(i)= Van + j*(-Vbn/sqrt(3) +Vcn/sqrt(3));

is_a1_k1(i)=((ts*Rr)/(sigma*Ls*Lr)-(1*j)*(ts*N_pi1)/(sigma*Ls))*si_s1+...
        (1-(ts*Rs)/(sigma*Ls)-(ts*Rs)/(sigma*Lr)+(1*j)*ts*N_pi1)*is_a1+...
        (ts/(sigma*Ls))*Vs(i);

si_s1_k1(i)= si_s1+ts*Vs(i)-ts*Rs*is_a1;

Te_pi1_k1(i)= 3/2*P*imag(si_s1_k1(i)*is_a1_k1(i));

gj=abs((T_ref-Te_pi1_k1(i))) + abs(7.5*(0.9960- abs(si_s1_k1(i))));

    if i == 1
        a = gj;
    end
    if i==1
        v=Vs(i);
    end
    if i==1
        S_state=S(1,:);

    end

    if gj<a

        a=gj;
        v=Vs(i);
        S_state=S(i,:);


    end
i=i+1;
```

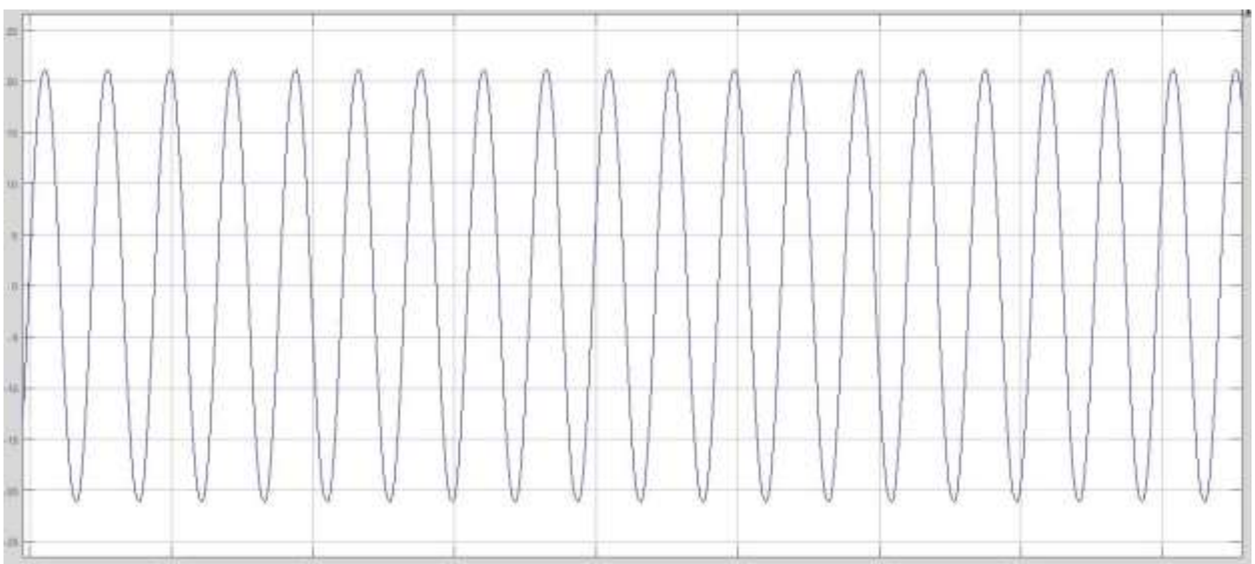
end

Sstate1=S_state(1);
Sstate2=S_state(2);
Sstate3=S_state(3);

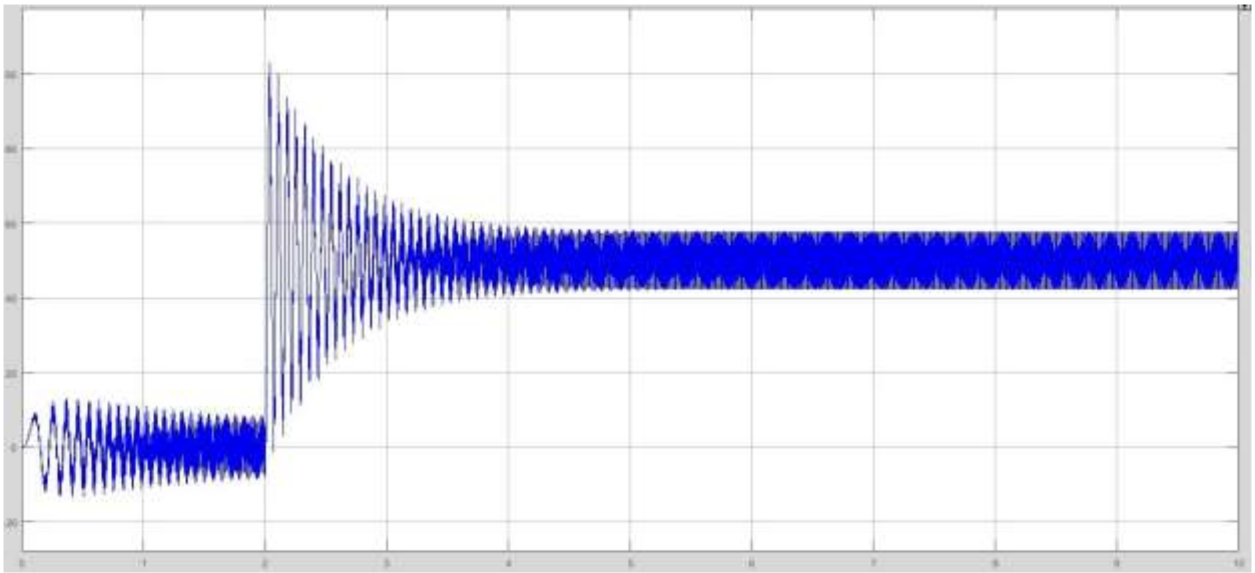
3.1.1 Results of first strategy:



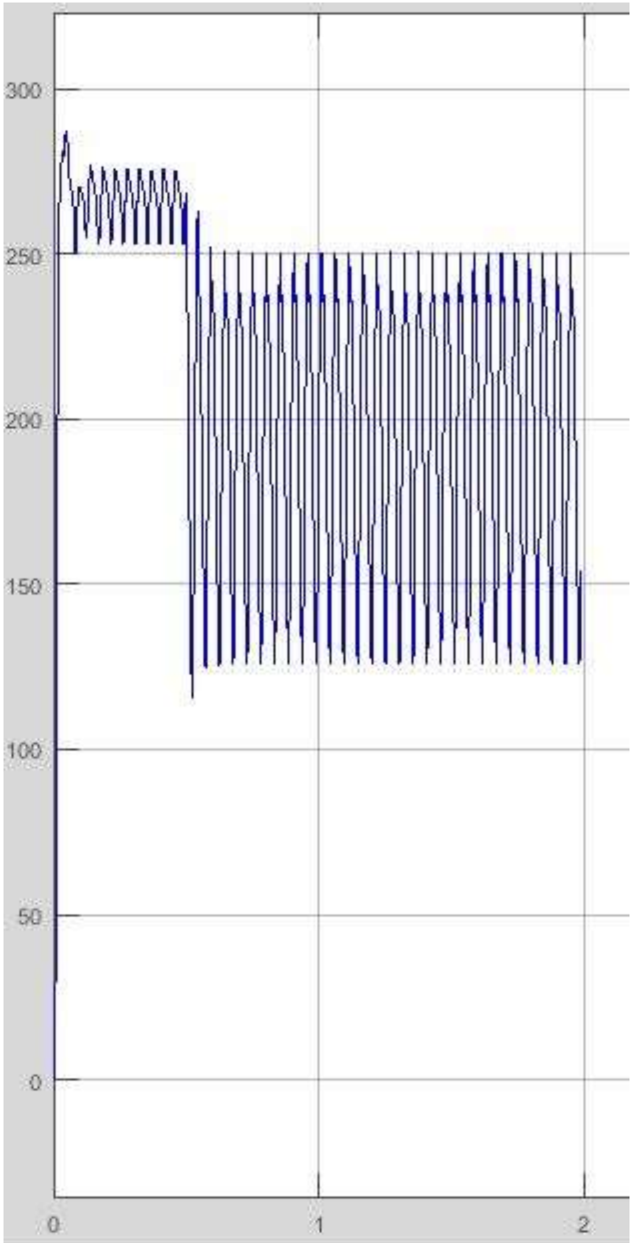
Voltages from the inverter.



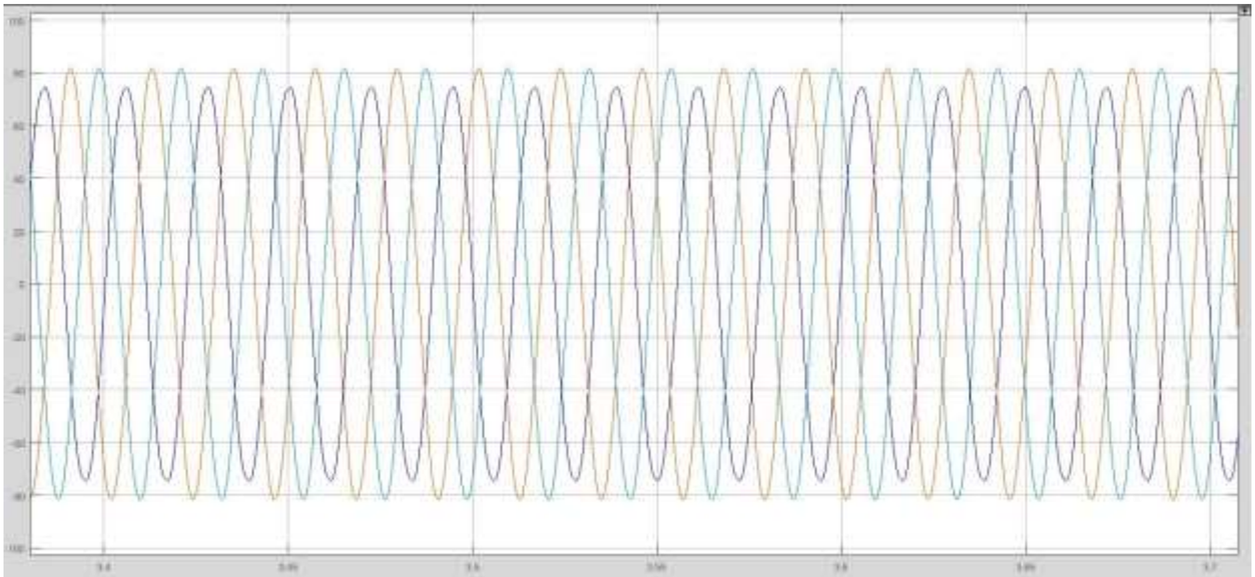
Flux.



Torque.



Speed



Currents

3.2.2 Strategy 2: Simution within the program

As explained above, inverter and IM are modeled in Simulink. The output voltage vectors, Current Is, Flux Ψs and Rotor speed Wr are used to calculate the predicted value of current, Flux and Torque for the next sampling instant. In this strategy, the values are predicted for each sampling time for the given voltage vectors and the most optimum state of the switching is taken and given to the inverter. With the new states, the simulation is run again and next sampling interval is taken which follows the same procedure. Thus, simulation is run with the new states again until all the sampling intervals for the given prediction horizon are run.

Program:

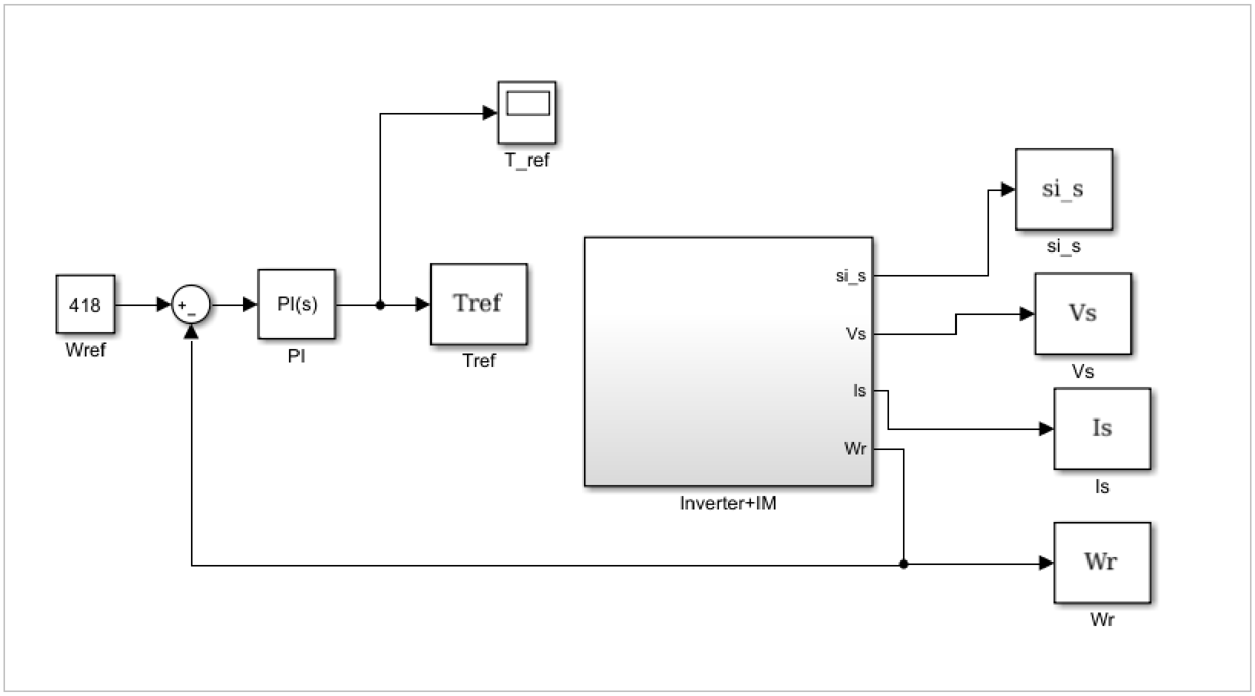
The algorithm used in this strategy is as follows:

```
clear
clc
Vdc=582;
Rs=2.68;
Rr=2.13;
Lm=0.275;
Ls=0.2834;
Lr=0.2834;
P=2;
J=0.005;
ts=0.0001;
```

```
si_s_ref=0.3558;
lamda1=1;
lamda2=7.5;
sigma=1-(Lm^2-(Ls*Lr));
a=2*520/3;
Vi=[0; a; a*(1/2); a*(-(1/2)); -a; a*(-(1/2)); a*((1/2)) ];
S=[0 0 0; 0 0 1; 0 1 0; 1 0 0; 1 0 1; 1 1 0;1 1 1];
S1=1;
S2=1;
S3=0;
S4=0;
S5=0;
S6=1;

simOut = sim('Trial','ReturnWorkspaceOutputs','on');
n=length(simOut.tout(:,1));
n1=round(n/1000);
for i=1:n1

    for m=1:7

        Is_pre(m)=((ts*Rr)/(sigma*Ls*Lr)-j*(ts*(simOut.Wr(i)))/(sigma*Ls))*(simOut.si_s(i))+...
            (1-(ts*Rs)/(sigma*Ls)-(ts*Rs)/(sigma*Lr)+j*ts*(simOut.Wr(i)))*(simOut.Is(i))+...
            (ts/(sigma*Ls))*Vi(m);

        si_s_pre(m)= (simOut.si_s(i))+ts*Vi(m)-ts*Rs*(simOut.Is(i));

        T_pre(m)= 3/2*P*imag(si_s_pre(m)*Is_pre(m));

        G(m)=lamda1*abs((simOut.Tref(i))-T_pre(m))+lamda2*abs(si_s_ref-si_s_pre(m));


    end
    Switch=S(1,:);
    cost=G(1);

    for k=2:7
        b=G(k);
        if b<cost
            cost=b;
```

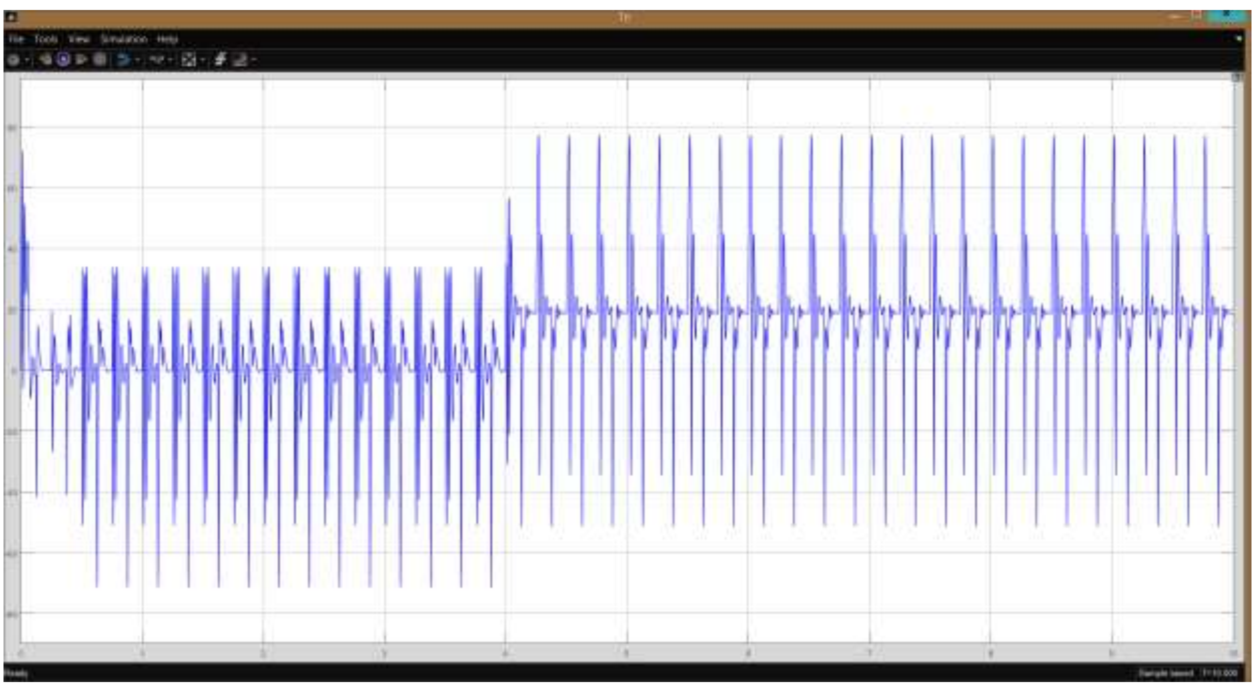
```
            Switch=S(k,:);
        end
    end


    S1=Switch(1);
    S2=Switch(2);
    S3=Switch(3);
    if S1==0
        S4=1;
    else
        S4=0;
    end
    if S2==1
        S5=0;
    else
        S5=1;
    end
    if S3==0
        S6=1;
    else
        S6=0;
    end

    simOut = sim('Trial','ReturnWorkspaceOutputs','on');

end
```

The Simulink Model is as shown:

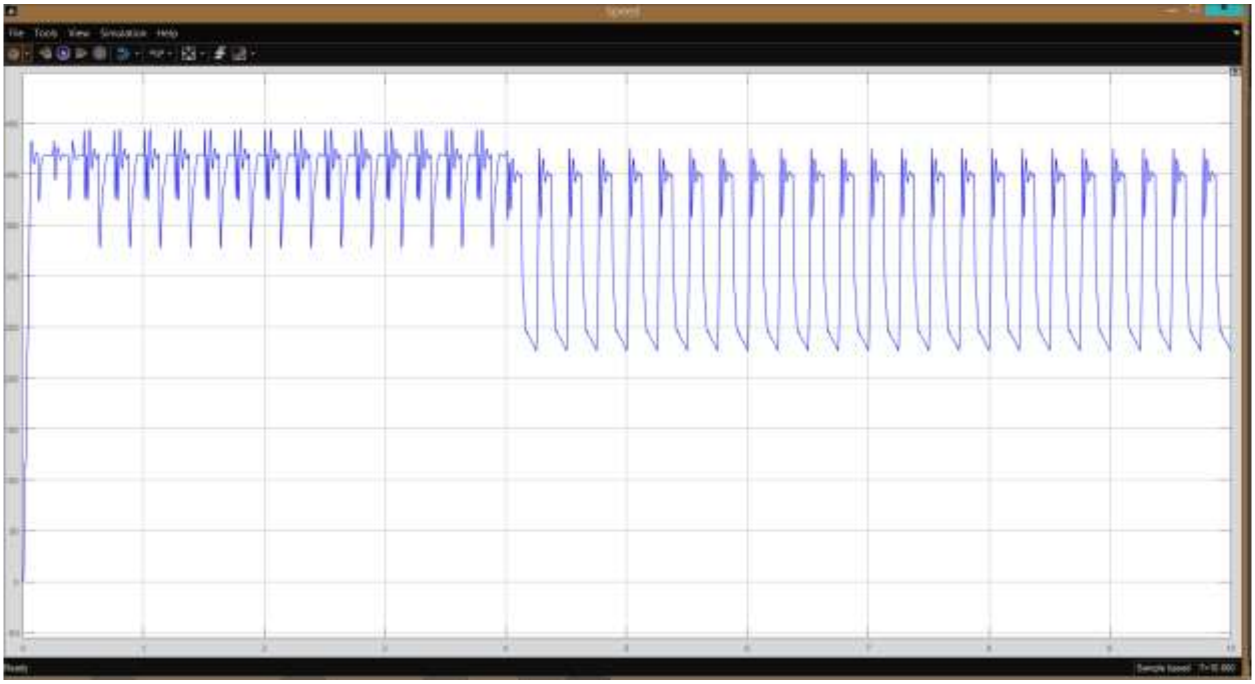


3.2.1 Results of second strategy

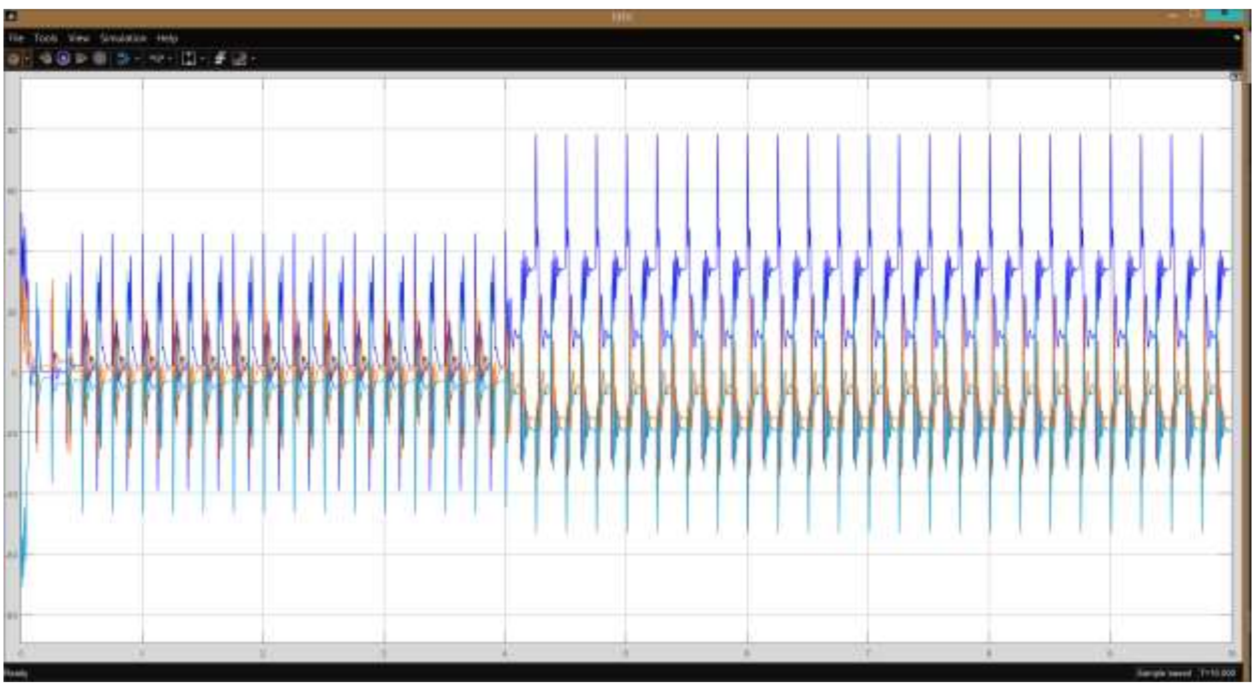
Torque:



Speed:



Current:



Conclusion

In this project we have implemented model predictive control of induction machine. And compared it with the results of a motor without any control. Thus, we found out that ripples of torque waveform were considerably reduced and settling time was also less. Hence this model can be used where there is a need of sudden increase of torque.

References

[1] S. Vazquez, J.Roderiguez, M.Rivera, L.Franquelo and M.Norambuena, "Model Predictive Control for Power Converters and Drives: Advances and Trendsl," *IEEE Trans. Ind. Electronics.*, vol. 64, no. 2, Feb. 2017.

[2] F. Wang, S. Davari, Zhe chen, Zhenbin Zhang, D. Khaburi, J. Rodriguez, Ralph Kennel, "Finite Control Set Modern Predictive Torque Control of Induction Machine with Robust Adaptive Observer", *IEEE Trans. Ind. Electronics.*, vol. 64, no. 4, April 2017.

[3] A.W Leedy, "MATLAB Dynamic Induction Motor Model for use in Undergraduate Electric Machines and Power Electronics Courses".