DSP Based Model Predictive Control for DC Motor

by

Md. Shakabul Islam Sourav

MASTER OF SCIENCE

IN

ELECTRICAL AND ELECTRONIC ENGINEERING

Department of Electrical and Electronic Engineering
Islamic University of Technology (IUT)
Board Bazar, Gazipur-1704, Bangladesh
May 16, 2023

DSP Based Model Predictive Control for DC Motor

by

Md. Shakabul Islam Sourav

MASTER OF SCIENCE

IN

ELECTRICAL AND ELECTRONIC ENGINEERING

Department of Electrical and Electronic Engineering
Islamic University of Technology (IUT)
Board Bazar, Gazipur-1704, Bangladesh
May 16, 2023

CERTIFICATE OF APPROVAL

The thesis titled "**DSP Based Model Predictive Control for DC Motor"** submitted by Md. Shakabul Islam Sourav, Student No. 171021017 of Academic Year 2017-2018 has been found as satisfactory and accepted as partial fulfillment of the requirement for the Degree of Master of Science in Electrical and Electronic Engineering on May 2023.

Board of Examiners:

1. ------------------------------

Dr. Golam Sarowar (Supervisor)
Professor,
Electrical and Electronic Engineering Department,
Islamic University of Technology (IUT), OIC, Gazipur

2. -------------------------------

Dr. Mohammad Rakibul Islam (Ex-Officio) Member
Professor and Head EEE Department,
Electrical and Electronic Engineering Department,
Islamic University of Technology (IUT), OIC, Gazipur

3. -------------------------------

Dr. Ashik Ahmed Member
Professor,
Electrical and Electronic Engineering Department,
Islamic University of Technology (IUT), OIC, Gazipur

.

4. ------------------------------

Dr. Md. Raju Ahmed Member (External)
Professor,
Electrical and Electronic Engineering Department,
Dhaka University of Engineering & Technology (DUET)

DECLARATION OF CANDIDATE

It is hereby declared that, this thesis report or any part of it has not been submitted elsewhere for the award of any Degree or Diploma.

Md. Shakabul Islam Sourav

Student No: 171021017

Academic Year: 2017-18

Date: May 16, 2023.

ACKOWLEDGEMENT

I would like to convey my heartiest gratitude to my supervisor Dr. Golam Sarowar, Professor, EEE Department, Islamic University of Technology (IUT), Gazipur, for the encouragement and valuable advice provided to me during completion of my thesis. I couldn't have completed my research without his continuous guidance and support.

I would like to thank all the faculty members and staffs of the Department of EEE, IUT for their inspiration and help. I would like to thank my parents for their blessings and my family and friends for having faith in me throughout the times of this thesis.

Finally, I would like to express my unparalleled gratitude to Almighty Allah for divine blessing without which it would not have been possible to complete this thesis successfully.

Md. Shakabul Islam Sourav May 16, 2023.

LIST OF ABBREVIATIONS

MPC	Model Predictive Control
DSP	Digital Signal Processing
MIMO	Multiple-Input Multiple-Output
PID	Proportional-Integral-Derivative
DC	Direct Current
AC	Alternating Current
TF	Transfer Function
OS	Over Shoot
PF	Power Factor
RPM	Revolutions per Minute
PWM	Pulse Width Modulation
LP	Linear Programming
QP	Quadratic Programming
GA	Genetic Algorithm
ADC	Analog to Digital Converter
DAC	Digital to Analog Converter
HIL	Hardware in the Loop
S	Second

ABSTRACT

This thesis presents a DSP based Model Predictive Control approach for controlling a DC motor. The MPC algorithm is designed to optimize the motor's performance by predicting its behavior over a finite time horizon and adjusting the control inputs accordingly. The proposed method provides improved performance in terms of faster response time, settling time, efficient tracking of reference trajectories and minimum steady-state errors. The system performance is evaluated under different operating conditions, including changes in sampling time, load torque, motor speed, and ability to handle constraints. The results show that the DSP based MPC approach provides better performance compared to traditional PID control methods. Further, the proposed method is implemented on a digital signal processor based hardware platform, and the results show that it is feasible for real-time control applications. The suggested approach illustrates how MPC can be a viable solution for the precise and efficient regulation of DC motor in real-world scenarios.

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction

1.1 Introduction

The implementation of control strategies mostly derived from mathematical modeling of the system is the main focus of control system engineering operations. Many industrial applications currently require advanced control systems, which are widely available. Model Predictive control method, hybrid predictive method, robust adaptive control method, fuzzy control (PID, FPGA), logic neural networks are some of the advanced control methods of control system engineering [1]. MPC is a real-time optimization technique that allows the control system to make decisions based on predictions of the future behavior of the system. The MPC algorithm is designed to minimize a cost function subject to constraint, while also taking in to account the dynamics of the system [2].

1.2 Problem statement

DC motor faces problems including speed fluctuations due to load variations, sensitivity to input voltage changes, parameter variations affecting response time, inadequate control system design, and sensor noise impacting accuracy. These issues can result in poor speed regulation, reduced efficiency, and compromised performance. Addressing these problems often involves control system optimization, parameter tuning, and maintenance practices to ensure reliable and accurate motor operation [3].

Model Predictive Control (MPC) can be an effective solution to address the speed regulation problems in DC motors. By utilizing a dynamic model of the motor, MPC can predict its future behavior and optimize control actions to minimize speed fluctuations and maintain the desired speed accurately.

Electrical motor's MPC applications are yet mostly unexplored. The computing hardware's performance has grown quickly. MPC can be used for rapid systems with lower time steps [4].

1.3 Research Objectives

The research is based on modeling of a DC motor. MPC will be used to control the speed of the motor using DSP. The objectives of this study with specific aims are:

- To achieve a stable system minimizing rise time, peak time and settling time
- To avoid overshoot for smooth reference tracking
- Be able to handle constraints
- Practical implementation to achieve the above criteria.

1.4 Literature Review

MPC is an advanced approach to process control that is widely employed in industry. For nonlinear systems, there has been much advancement, but there are still many challenges that need to be resolved for practical applications, such as the dependability and effectiveness of the online computation method. A complex dynamic programming problem needs to be 'rigorously' solved in order to deal with model uncertainty [5].

MPC's implementation was previously restricted to slowly changing systems because of the computational work involved in it, and these systems have time steps that are long enough to allow the control algorithm to run entirely. Now faster algorithms have been developed and computing hardware performance has improved rapidly, MPC can be implemented for quick systems with smaller time steps. Additionally, it provides sufficient tools for handling multivariable restricted control issues [6].

Despite of the benefits highlighted, few research labs are now involved in MPC applications to electrical drives, which are still largely unexplored. For example, MPC PWM control has been applied to the electronic drive system [7] and then for the speed control [8]. In other work [9] MPC has been used as a torque/flux controller of induction motor at a wide range.

Modern industrial applications depend mainly on digital control platforms like DSP. Digital control is suitable and efficient for industrial regulations and code requirements [10]. A fully DSP-based real-time data acquisition control system and MATLAB/Simulink environment are used during the design, analysis, and implementation phases. The Digital signal processor system has rapidity, control accuracy, adaptability, and robustness.

In the literature, the MPC technique is very often seen to be implemented at an earlier age when the digital processor was not as fast as nowadays.

Hence, a new MPC for DC motor control can be implemented in DSP for better performance and stability purposes.

1.5 Research Methodology

The research methodology of a DSP based MPC for DC motor involves problem statement, literature review, system modeling, MPC design, data collection, model identification, controller implementation, performance evaluation, and conclusion with future research suggestions.

The research problem must be precisely defined as the initial step or objective related to the DSP based MPC for DC motor model. This helps in providing a clear direction and focus for the research. The next step involves conducting a thorough literature review to gain insights into existing knowledge and research in the field of MPC for DC motor.

After the literature review, a mathematical model of a DC motor has to be developed. This model should accurately represent the electrical and mechanical dynamics of the motor.

The next step is to collect data to validate and tune the DC motor model. This data can be collected through simulations and experiments by measuring motor inputs and outputs under different operating conditions.

Once the model is identified, implementation of the MPC algorithm on a suitable platform or simulation environment has to be done. The implementation should consider the computational requirements and real-time constraints of the system. The output results (settling time, rise time, % of OS, speed and output voltage) should be collected and compared. Hardware implementation of the DC motor on DSP board has to be done. The obtained results should be analyzed and interpreted to draw conclusions.

Finally, the research concludes by summarizing the findings and suggesting future research directions.

1.6 Outline of the Thesis

The book is organized in the following chapters

Chapter 1: Represents the problem statement, research objectives and literature review.

Chapter 2: Represents the characteristics of DC Motor and MPC.

Chapter 3: Represents the DC motor model and MPC algorithm.

Chapter 4: Represents the simulation results of the MPC and PID applied on DC motor model.

Chapter 5: Represents the Hardware testing results of the MPC and PID on DC motor.

Chapter 6: Represents conclusion and future work

Chapter 2
DC Motor & MPC Characteristics

2.1 DC Motor

A DC motor is an electric motor that runs on direct current. Through electromagnetic contact between the armature and the magnetic field, it transforms electrical energy into mechanical energy. From little toys to massive industrial systems, DC motors are frequently utilized in a wide range of applications [11]. The motor's speed and torque can be adjusted by regulating the DC voltage applied to it or controlling the current flow.

2.1.1 Working Principle with Illustration

A DC motor's operation is defined by Faraday's Law of electromagnetic induction. It claims that when a current-carrying conductor is exposed to a magnetic field, a force is generated.

In a DC motor, the stator's fixed magnetic field creates rotation for the armature. The armature consists of a coil of wire that carries a current. This current interacts with the magnetic field, causing the armature to rotate. By altering the direction of the current flow, the rotation direction can be altered. The armature is attached to an axle that is supported by bearings, allowing it to rotate freely. The commutator, a split ring connected to the ends of the armature coil, ensures that the current always flows in the same direction, allowing the motor to maintain continuous rotation.

DC motors can come in brushed and brushless varieties. Brushes are used to deliver current to the armature in brushed DC motors, which have a spinning armature and a stationary magnetic field. On the other hand, brushless DC motors have a stationary armature and a spinning magnetic field, and the current is sent to the coils on the armature by an electronic commutation mechanism [12].

By regulating the amount of current passing through the armature, the motor's speed can be managed. This can be achieved by altering the voltage delivered to the motor or by regulating the current via a control circuit.

Fig 2.1 shows the simplified configuration of two brushes and a two-piece commutator that represents an example of the DC motor's operating principle [13]. The commutator is connected to the leads of the rotor coil, which is situated between magnetic fields.

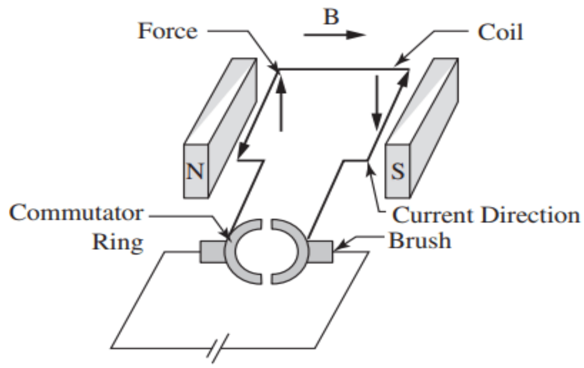

Fig 2.1: Working principle of DC motor.

When a current runs through the magnetic field produced by the stator in the rotor coils, a brush-type DC motor creates a torque (and subsequently rotates the rotor). Two equal but opposing forces acting on the sides of the coil produce the torque. According to Lorentz's law, a force acts on a conductor when it carries a current through a magnetic field. This force is represented mathematically as the vector cross-product of the current, the magnetic field and the conductor length:

$$F = ILB \sin\theta$$

Where, I is the current vector, B is the magnetic field vector, θ is the angle between the current and magnetic field direction and L is the length of the conductor [14].

Without a commutator to maintain the current's direction, the torque direction will change as soon as the coil crosses the vertical plane (also known as the commutation plane), which will result in no significant motion. The torque produced by a two-piece commutator will not be smooth and exhibit the ripple in Fig. 2.2(a).

Because the moment arm distance between the forces acting on the coil is maximum in this position, the torque is maximum when the coil is horizontal [15]. If the moment arm distance is zero while the coil is in the commutation plane, there will be zero torque.

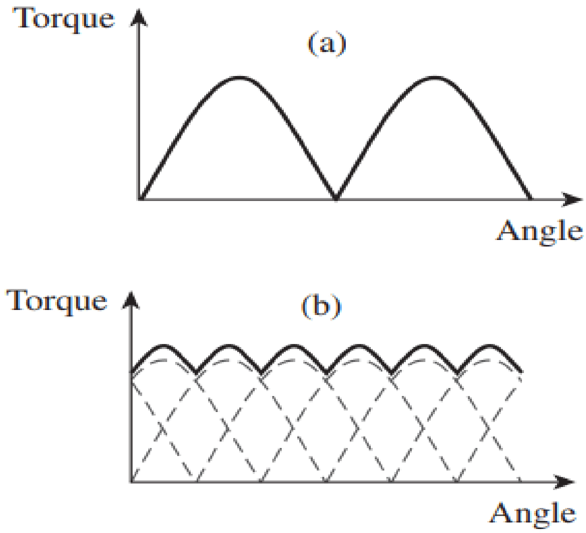

Fig 2.2: Torque Direction

The torque would be smoother (Fig. 2.2(b)) if we had used a six-piece commutator (and three coils, one for each commutator pair) as opposed to a two-piece commutator since the torque is the sum of all the torques in all the coils at any given time. The ripple would not exist since the angle would always be 90° if it were possible to make the stator magnetic field radial. Commercial motors have commutators with 50 or more segments to improve the torque characteristics.

2.1.2 Characteristics of DC Motor

A DC motor's performance may be evaluated based on its properties, making it simple to choose a motor for a certain application [16].The performance attributes of DC motor are:

- Torque versus armature current (T vs I_a)
- Speed versus armature current (N vs I_a)
- Torque versus speed (T vs N)

For DC motor characteristics, the following two relations are most important:

- Ta $\propto \Phi I_a$ and
- N $\propto E_b \Phi$

2.1.3 Speed control of DC motor

A DC motor's nameplate will state its base speed, which is an indication of how quickly the motor will operate at its rated armature voltage and current. By lowering the voltage applied to the armature and lowering the field current, a DC motor can be operated below base speed and above base speed.

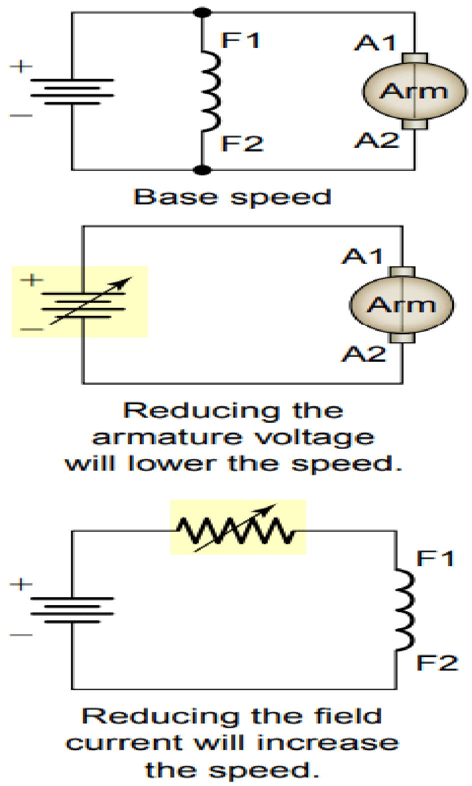

Fig 2.3: Speed control of DC motor [17].

Speed Control of DC Motor by Armature

The field is connected across a constant-voltage supply in armature-controlled adjustable-speed applications, where the armature is connected across an adjustable voltage source (Figure 2.4).

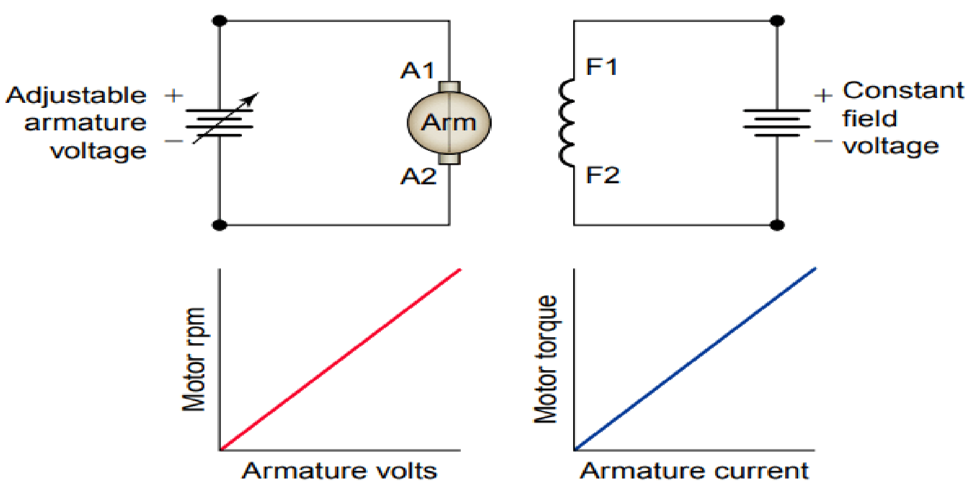

Fig 2.4: Armature voltage controlled DC motor [17].

The motor speed will change in response to changes in armature voltage in a proportional manner.

Speed Control of DC Motor by Field

By weakening the field, shunt motors can be made to run faster than their base speed. The motor is typically started with the highest field current to generate the highest flux and highest starting torque.

The flux weakens as the field current decreases, increasing the speed. A decrease in field current will also lead to a higher armature current flow for a given motor load and less counter EMF generation. A straightforward way to control a field is to connect a resistor in series with the source of the field voltage [18]. This could be helpful for fine-tuning to the motor speed that works best for the application. An alternative, more advanced technique makes use of a variable-voltage field source.

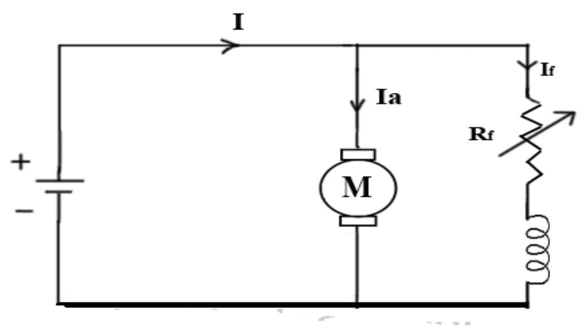

Fig 2.5: Field Flux controlled DC motor.

2.2 MPC

MPC is a powerful tool for controlling dynamic systems, and is increasingly being used to overcome the limitations of traditional control methods such as PID control. In MPC, a plant mathematical model is used for forecasting the system's behavior while considering the plant's present state and control inputs. This forecast is used to identify the optimum level of control that will minimize a cost function that demonstrates the system's desired performance [19]. The control action is then applied to the process, and the cycle repeats.

2.2.1 MPC Advantages

The capability of MPC to manage restrictions, such as process limits, equipment limitations, and safety regulations, is one of its main advantages. These constraints can be incorporated into the cost function, and the MPC algorithm will determine the control action that optimizes performance while satisfying the constraints. This is in contrast to traditional control strategies such as PID, which may struggle to handle constraints effectively, leading to suboptimal performance or even system failures [20].

The capability of MPC to handle complex and nonlinear systems is an additional advantage. The mathematical model used in MPC can be based on physical or empirical models of the process, and can be adjusted to account for nonlinearities and dynamic behavior. This allows MPC to handle processes that are difficult or impossible to control using traditional control strategies.

MPC can also be used to optimize multiple objectives, such as maximizing production, minimizing energy consumption, or reducing emissions. These objectives can be balanced using the cost function, allowing MPC to find the control action that optimizes performance in all aspects of the process.

2.2.2 MPC for DC Motor

MPC can be applied to DC motor, which are electric motors used to power a numerous applications including conveyors, pumps, fans, and other mechanical devices. MPC provides a powerful tool for controlling the speed, torque, and position of DC motor, allowing for improved performance and control.

In motor applications, MPC can be used to regulate the speed and to control the load torque. MPC can forecast system actions in the future and choose the best control inputs to achieve particular performance goals by modeling the motor and the load [21].

2.2.4 Basic Steps of MPC Algorithm

Implementation of MPC typically involves several stages, including model development, model validation, and controller design. The primary step is to develop a model of the motor and the load, which can be based on physical or empirical models. This model should be validated by comparing its predictions to real-world data from the motor drive. The controller design stage involves tuning the MPC algorithm to meet the specific requirements of the application, including setting the control horizon, choosing the cost function, and setting the constraints.

MPC can be implemented utilizing a variety of platforms, including programmable logic controllers (PLCs), and supervisory control and data acquisition (SCADA) systems, depending on the necessary hardware and software. MPC software is also available as standalone products, and can be integrated with existing control systems [22].

A mathematical model of the system is used by the control algorithm to forecast the behavior of the system in the future and identify the best control inputs to meet particular performance goals. MPC algorithms are designed to handle multiple objectives, constraints, and disturbance variables in real-time control applications [23].

The basic steps of the MPC algorithm are as follows:

i. **Model Development**: Determining the system’s mathematical model that to be controlled is the first stage. This model should be able to predict the system's future behavior and capture the main characteristics of the system.
ii. **Prediction Horizon**: The period of time into which the MPC algorithm predicts the system behavior is known as the prediction horizon. This horizon is typically chosen based on the control requirements of the application.
iii. **Cost Function**: The performance of the control inputs is determined using the cost function. The cost function should reflect the objectives of the control application, such as minimizing energy consumption, maximizing production.
iv. **Constraints**: Constraints are used to limit the control inputs and ensure that they remain within safe and feasible limits. These constraints can include speed limits, torque limits, and current limits.
v. **Optimization:** The MPC algorithm finds the control inputs that minimize the cost function while satisfying the constraints by using an optimization algorithm, such as linear programming or quadratic programming.
vi. **Control Input Update**: The control inputs are updated based on the optimization results, and the system is updated with the new control inputs from the optimizer.
vii. **Repeat**: The procedure is repeated, with the prediction horizon being moved ahead in time and the model is updated on the basis of new feedback from the system.

The algorithm is designed to handle multiple objectives, constraints, and disturbance variables in real-time, making it well-suited for a wide range of control applications [24].

2.3 Digital signal processing

Using mathematical techniques, digital signal processing is a way of extracting information from real-world signals (expressed as a list of integers). Math can decode the ones and zeros that make up a digital signal. Analog signals include signals from the physical world such as sound, light, temperature, pressure, and others. In a digital signal, the analog signal is numerically represented. These signals may be easier and more reasonably processed in the digital age. [25]. As shown in Fig. 2.6, in the real world, we can transform these signals into digital signals using our analog-to-digital conversion method, process the digital signals, and if necessary, return the digital signals to the analog world.

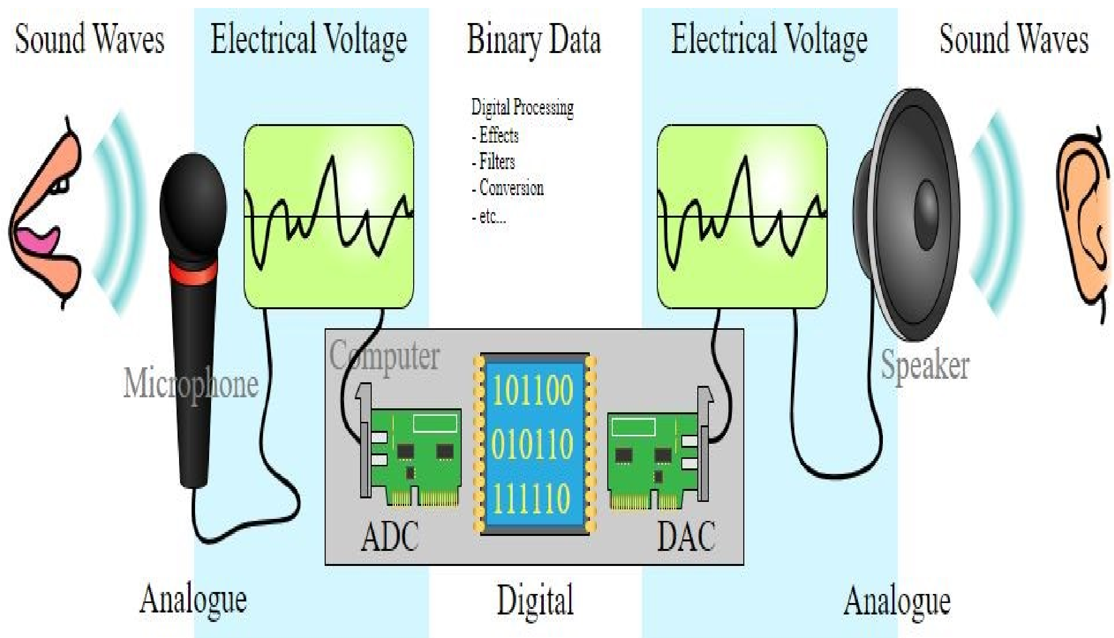

Fig 2.6: ADC and DAC [26]

2.3.1 DSP for Control System

DSP algorithms are used to process signals from sensors and to implement advanced control algorithms, such as MPC. The use of DSP in control systems has made it possible to implement real-time control systems that respond quickly to changes in the system being controlled.

Compared to conventional analog control systems, the use of DSP in control systems improves accuracy and robustness by enabling the application of advanced control algorithms like MPC.

In control systems, DSP algorithms are used to process signals from sensors, such as accelerometers, gyroscopes, and strain gauges, to obtain information about the system being controlled. The processed signals are then used by the control algorithm to determine the control inputs that will achieve the desired performance objectives [27].

DSP algorithms are used in control systems to perform various operations, such as filtering, Fourier transforms, and signal processing, to remove noise and improve the accuracy of the control inputs. In addition, DSP can be used to implement control algorithms.

DSP technology has also made it possible to implement real-time control systems, which can respond to changes in the system being controlled in real-time. This is important in many control applications, such as motor control systems, where it is essential to respond quickly to changes in the system.

Chapter 3
MPC for DC motor model

3.1 Importance of MPC for DC Motor

MPC is a type of advanced control strategy that can be applied to DC motor to improve performance and efficiency. MPC is particularly useful in applications where the control inputs (e.g., the voltage, torque or current applied to the motor) must be carefully regulated to achieve precise performance requirements.

One of the main benefits of MPC is its capability to optimize the control strategy in real time while taking into account the dynamics of the motor, the load it is driving, and any restrictions on the control inputs. This can lead to better performance and energy efficiency compared to traditional control methods [28].

3.2 DC Motor Dynamic Modeling

A DC motor can be modeled using various approaches, but one of the most common models is the electrical equivalent circuit model. This model represents the DC motor as an electrical circuit, where the motor's electrical and mechanical properties are represented by different components in the circuit. Fig 3.1 shows DC Motor electrical model [29].

The fundamental elements of the DC motor's electrical equivalent circuit model are:

1. **Armature resistance (Ra):** This component represents the resistance of the wire in the motor's armature.
2. **Armature inductance (La):** This component represents the inductance of the wire in the motor's armature.
3. **Back EMF (Eb):** This component represents the voltage generated by the motor as it rotates.
4. **Torque (T):** This component represents the force generated by the motor as it rotates.
5. **Mechanical inertia (J):** This component represents the motor's resistance to changes in speed.
6. **Damping coefficient (B):** This component represents the frictional forces that act on the motor.

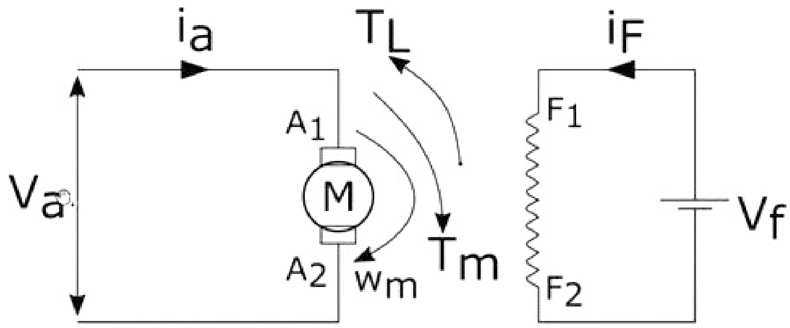

Fig 3.1: DC motor equivalent model.

The electrical equivalent circuit model of a DC motor can be represented by the following equations:

$V_a = i_aR_a + L_a \frac{dia}{dt} + E_b$... (1)

Here, V_a is the applied voltage and i_a is the armature current of the motor.

Back emf, $E_b = (K_e\varphi) * \omega_m$…… (2)

$K_e\varphi$ is constant if field remains constant and ω_m is the angular velocity of the motor shaft.

$T_m = (K_e\varphi) * I_a$ .... (3), where T_m is the mechanical torque developed by the shaft.

$= T_L + J \frac{d\omega m}{dt} + B * \omega_m$.... (4) Here, T_L is the load Torque.

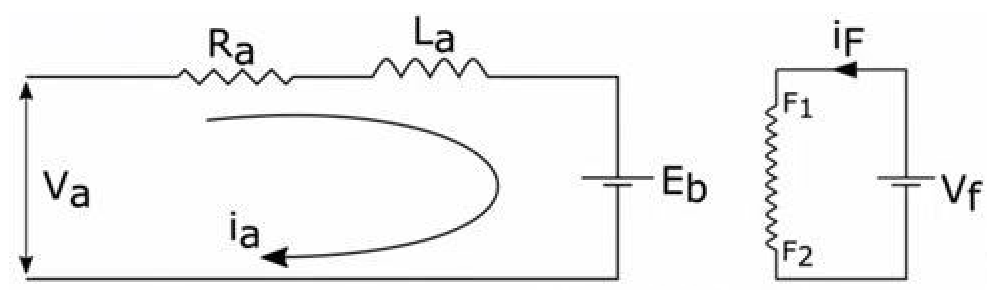

Fig 3.2: DC Motor electrical circuit model

Again, from equation 1 we get applied armature voltage,

$V_a = i_aR_a + L_a \frac{dia}{dt} + E_b$

The dynamic model can be further simplified by using Laplace transforms and assuming steady-state conditions.

Taking Laplace transformation

$$V_a(s)= I_a(s)*[R_a+ sL_a] + E_b (s)$$

Or, $V_a(s) - E_b (s) = I_a(s)*[R_a+ sL_a]$

$$\text{Or, } I_a (s) = \frac{Va(s) - Eb (s)}{Ra+ SLa}$$

So, the armature current I_a will be

$$I_a (s) = \frac{Va(s) - Eb (s)}{Ra(1 + S\frac{La}{Ra})}$$

$$\text{Or, } I_a (s) = \frac{Va(s)-(Ke\varphi)*\omega m(s)}{Ra(1+ s\tau a)} \quad \ldots (5)$$

Now, from equation (4) taking Laplace transformation we get,

$$T_m (s) - T_L (s)= \omega_m(s) * [B +sJ]$$

So, the angular velocity

$$\omega_m(s)= \frac{Tm (s) - TL (s)}{B(1 +s\frac{J}{B})}$$

The mechanical time constant, $\tau_m= \frac{J}{B}$

So,

$$\omega_m(s)= \frac{T_m (s) -T_L (s)}{B(1 +s\tau_m)} \ldots.. (6)$$

Hence, the angular speed from equation (3) and (6)

$$\omega_m(s)= \frac{(Ke\varphi)* Ia (s) -T_L (s)}{B(1 +s\tau_m)} \ldots\ldots (7)$$

Considering the angular velocity $\omega_m(s)$as the output variable and the applied armature voltage $V_a(s)$ as the input variable and from (7) and (5), the following open-loop transfer function is derived,

$$G_{w_m V_a}(s) = \frac{\frac{(Ke\varphi)}{B(1 + s\tau_m)}}{[Ra + sLa] + \frac{(Ke\varphi) * \omega m}{I_a}}$$

As $L_a << R_a$, by neglecting the L_a term,

$$G_{w_m V_a}(s) = \frac{(\mathbf{Ke\varphi})}{\mathrm{Ra}*\mathbf{B}(\mathbf{1} + \mathbf{s}\boldsymbol{\tau}_{\boldsymbol{m}}) + (\mathbf{Ke\varphi})^2} \quad \ldots\ldots (8)$$

Taking angular position and angular velocity as the states and applied armature voltage as the input, equation (8) can be rewritten in state space form as

$$x(t) = Ax(t) + Bu(t) \ldots\ldots (9)$$

$$y(t) = Cx(t) + Du(t) \ldots. (10)$$

Where, system matrices A, B, C and D are given as

$$A = \begin{bmatrix} 0 & 1 \\ 0 & \frac{-1}{\tau m} \end{bmatrix} \quad ; \quad B = \begin{bmatrix} 0 \\ \frac{Ke\varphi}{\tau m} \end{bmatrix} ,$$

$$C = \begin{bmatrix} 0 \\ 1 \end{bmatrix}^T \quad ; \quad D = 0.$$

3.3 MPC Model Structure

MPC is implemented for a variety of MIMO systems and it can handle constraints systematically without modification. In MPC, the current control signal is selected in a manner that will result in desirable output behavior in the future. Therefore, the minimum capacity is required to precisely forecast the system's future output behavior. The process's future behavior depends on both the previous inputs and the potential future inputs that are contemplating [30]. A feedback path is used in the MPC structure to compute the process measurements. Fig 3.3 shows the components of MPC structure. There are main components available in MPC structure are:

1. The plant model
2. The cost function
3. The optimizer

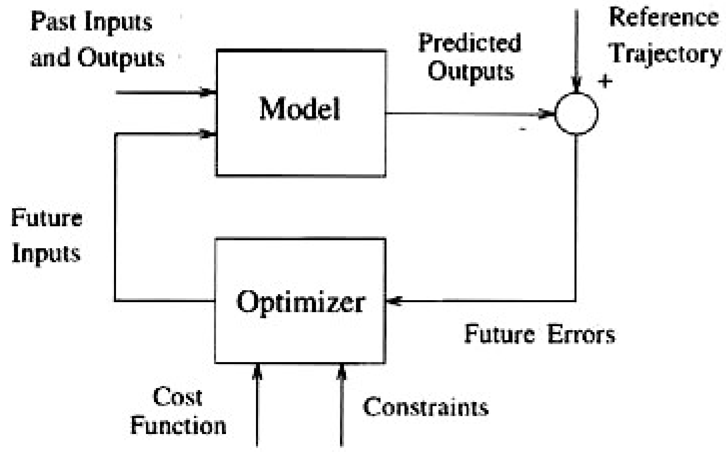

Fig 3.3: Components of MPC structure [31].

3.3.1 Plant Model

The plant is a DC motor in this research. From the previous discussion the armature current, torque and angular speed characteristic of a DC motor from equations (3), (5), (7) and (8) can be determined.

$$\text{Torque, } T_m = (K_e\varphi) * I_a$$

$$\text{Armature current, } I_a\,(s) = \frac{Va(s) - (Ke\varphi) * \omega m(s)}{Ra(1 + s\tau a)}$$

$$\text{Angular speed, } \omega_m(s) = \frac{(Ke\varphi) * Ia\,(s)\;\; - T_L\,(s)}{B(1 + s\tau_m\,)}$$

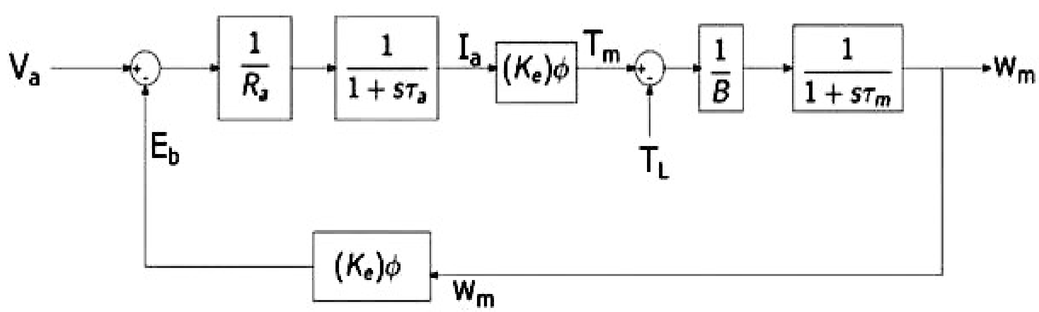

Fig 3.4: DC motor dynamic model block diagram.

Fig 3.4 shows the DC motor dynamic model block diagram with closed loop transfer function from equation (8),

$$G_{w_m V_a}(s) = \frac{(\mathrm{Ke}\varphi)}{\mathrm{Ra} * \mathrm{B}(1 + s\tau_m) + (\mathrm{Ke}\varphi)^2}$$

3.3.2 Cost function

Depending on the system type and performance requirements, the prediction horizon (N) is chosen, which affects how the cost function is evaluated for each sample interval. It is possible to design a cost function that takes future events, references, and actions into account. Either a LP or a QP can be used to formulate the linear MPC [32]. The goal of both formulations is minimization of the cost function.

3.3.3 Optimizer

Once the MPC problem is defined as a QP problem, obtaining optimal control inputs requires solving one QP for the specific initial conditions, which are typically comparable to state measurements. Interior Point Methods (IPM) and Active Set Methods (ASM) are the two widely used QP solution techniques used in MPC [33].

3.4 MPC Design for DC motor model

MPC also known as receding horizon control. The sequence of optimal control inputs is established for an anticipated evolution of the system model over a finite horizon. However, just the initial part of the control sequence is employed, and the succeeding sampling time is used to reevaluate the system's condition [34]. Through the introduction of input into the system, the so-called Receding Horizon Strategy (RHC) enables the system to be compensated for any potential modeling errors or disturbances.

Linear MPC problem is formulated as QP. Consider a discrete time LTI state space model of the system

$$x_{k+1} = Ax_k + Bu_k \ldots\ldots (9)$$

$$y_k = Cx_k \ldots\ldots\ldots\ldots\ldots (10)$$

Where $x \in R^n$ is the state vector, $u \in R^l$ is the control input, and $y \in R^m$ is the output vector.

$A \in R^{n\times n}$, $B \in R^{n\times l}$ and $C \in R^{m\times n}$ are the system matrices. Full state measurement and no disturbances or model uncertainty are assumed, unless explicitly specified.

1) Unconstrained MPC: The objective of MPC is to minimize the difference between output (y_k) and reference (r_k) . To do this, a least square problem can be used. This can be done using a least squares problem. Minimum cost function J_y can be written as

$$\min \ J_y = \frac{1}{2}\sum_{k=0}^{N_p} ||y_k - r_k||^2 Q$$

Since y_0 can't be influenced, the term $\frac{1}{2}\,||y_0 - r_0||^2 Q$ is discarded.

$$\min J_y = \frac{1}{2}\sum_{k=1}^{N_p} ||y_k - r_k||^2 Q \text{...... (11)}$$

The output equation can be written as

$$y_k = CA^k x_0 + C\sum_{j=0}^{k-1} A^{k-j-1} u_j \text{........ (12)}$$

$$y_k = yx_0 + yu_j \text{............. (13)}$$

This can be written as; $y_0 = Cx_0$

$$Y = \Phi x_0 + \Gamma U \text{......... (14)}$$

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ . \\ y_{Np} \end{bmatrix} = \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ . \\ CA^{Np} \end{bmatrix} x_0 + \begin{bmatrix} CA & 0 & . \; . \; . & 0 \\ CAB & CB & \vdots \; . \; \; . & 0 \\ \vdots & \ddots & \vdots & . \\ CA^{Np-1}B & CA^{Np-2}B & . \; . \; . & CB \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ . \\ u_{Np-1} \end{bmatrix}$$

$Here, y_r$ is introduced as a vector representing the required reference.

$$y_r = [r1 \quad r2 \ldots \ \ldots \ldots . \quad rNp]^T \ldots . . (15)$$

$$\text{So, min } J_y = \frac{1}{2}\sum_{k=1}^{N_p} ||Y - y_r||^2 Q \text{.... (16)}$$

Where the output weight matrix Q is given by

$$Q = \begin{bmatrix} q & 0 & 0 \\ 0 & q & 0 \\ 0 & 0 & q \end{bmatrix}$$

$$J_y = \frac{1}{2}\sum_{k=1}^{N_p} ||Y - y_r||^2 Q$$

$$=\frac{1}{2}||\Phi x_0 \ + \ \Gamma \mathrm{U} - y_r||^2 Q$$

$$=\frac{1}{2}|| \ \Gamma \mathrm{U} - (y_r - \Phi x_0)||^2 Q$$

$$=\frac{1}{2}|| \ \Gamma \mathrm{U} - \mathrm{b}||^2 Q, \text{ here } \mathrm{b}=(y_r - \Phi x_0) \ldots\ldots (17)$$

It is convenient to express this problem as a QP problem to make it more straightforward to solve;

$$J_y \ = \frac{1}{2}|| \ \Gamma \mathrm{U} - \mathrm{b}||^2 Q$$

$$= \frac{1}{2} \ (\Gamma \mathrm{U} - \mathrm{b})^{\mathrm{T}} \ Q(\Gamma \mathrm{U} - \mathrm{b})$$

$$= \frac{1}{2} U^T H_Q U + F_Q^T U + \rho \ldots.. (18)$$

Where, H, F, ρ are given by

$$H_Q = \Gamma^T \mathrm{QT}$$

$$F_Q = -\Gamma^T \mathrm{Qb} \quad \text{and} \quad \rho = \frac{1}{2} b^T Q_b$$

Since, ρ doesn't affect the solution to the problem, it can be ignored. This is the QP problem equivalent to problem (11). This QP formulation of equation (11) resulted in to

$$\text{Hence, min } J_y = \frac{1}{2} U^T H_Q U + F_Q^T U \ldots\ldots (19)$$

2) Regularization: The following step is to formulate the input, which can be accomplished by adding a new term., JΔu, in the objective function, where $\Delta u_k = u_k - u_{k-1}$. Control problem is then

$$min \ J_y = \frac{1}{2}\sum_{k=1}^{N_p} ||Y - y_r||^2 Q + \frac{1}{2}\sum_{k=1}^{N_p - 1} ||\Delta u_k||^2 R \ldots\ldots (20)$$

This new term requires making sure that steps in u are either continuously dropping or growing, which produces more "smooth" input, by minimizing the difference between two consecutive steps in u. Once more, this needs to be framed as a QP problem. Input term can be rewritten as

$$J_{\Delta u} = \frac{1}{2}\sum_{k=1}^{N_p-1} ||\Delta u_k||^2 R$$

$$=\frac{1}{2}\sum_{k=1}^{N_p-1} ||u_k - u_{k-1}||^2 R$$

$$= \frac{1}{2}\sum_{k=1}^{N_p-1} (u_k - u_{k-1})^T R\,(u_k - u_{k-1}) \ldots\ldots(21)$$

$$J_{\Delta u} = \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ . \\ u_{Np-1} \end{bmatrix} \begin{bmatrix} 2R & -R & . & . & 0 \\ -R & 2R & . & . & 0 \\ . & . & . & . & . \\ 0 & 0 & . & 2R & -R \\ 0 & 0 & . & -R & R \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ . \\ u_{Np-1} \end{bmatrix} + \begin{bmatrix} -R \\ 0 \\ \vdots \\ . \\ 0 \end{bmatrix} u_{-1}^T \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ . \\ u_{Np-1} \end{bmatrix} + \frac{1}{2} u_{-1} R u_{-1} \ldots .(22)$$

$$J_{\Delta u} = \frac{1}{2} U^T H_R U + (M_{u-1} u_{-1})^T U + \frac{1}{2} u_{-1} R_{u-1} \ldots\ldots (23)$$

This shows, that introducing $J_{\Delta u}$ extends the QP problem by following terms;

$$H_{\Delta u} = H_R \;;\; F_{\Delta u} = M_{u-1} u{-1}$$

Like with ρ, the term $\frac{1}{2} u_{-1} R_{u-1}$ is discarded, because it has no impact on how the is resolved. The new QP problem is;

$$\min J = \frac{1}{2} U^T H U + F^T U \ldots.. (24)$$

Where the terms H and F are given as

$$H = H_y + H_{\Delta u} = \Gamma^T Q T \;\; + H_R$$

$$F = F_Q + F_{\Delta u} = M_{X0} x_0 \; + M_R R + \; M_{u-1} u_{-1} \ldots\ldots.. (25)$$

3) Input Constraints: There are constraints set for the quantity of input, the movement rate of the input, and the output. It is assumed that the limits are the same for every time step k for all constraints. The maximum and minimum inputs are constrained by the input constraints. MPC with an input restriction;

$$min\ \ J_y = \frac{1}{2}\sum_{k=1}^{N_p} \left|\left|Y - y_r\right|\right|^2 Q + \frac{1}{2}\sum_{k=1}^{N_p-1} \left|\left|\Delta u_k\right|\right|^2 R \ldots\ldots (26)$$

$$\text{s.t.} \quad x_{k+1} = Ax_k + Bu_k, \text{ k=0,1,2, } \ldots\ldots\ldots\ldots\ldots, N_p - 1$$

$$y_k = Cx_k, \quad \text{k=0,1,2,} \ldots\ldots\ldots\ldots\ldots, N_p$$

$$u_{min} \le u_k \le u_{max}, \ \text{k=0,1,2,} \ldots\ldots\ldots\ldots\ldots, N_p - 1 \ldots\ldots (27)$$

This yields a constrained QP problem; QP formulation of problem

$$\text{min J} = \frac{1}{2} U^T H U + F^T U$$

$$\text{s.t. } U_{min} \le U_k \le U_{max}$$

Chapter 4
Simulation Results

4.1 Comparative Analysis

A comparative analysis between the MPC and PID for proposed DC motor model, along with sampling time variation have been done. Matlab Simulink based simulation data is observed for the comparison. Table 4.1 shows the motor parameters for the DC motor model. Both MPC and PID models have the same attributes. The proposed DC motor model is also compared with Simulink built-in DC motor model. MPC for DC motor model has been simulated with different sampling time. MPC DC motor model speed has also been analyzed with imposing constraint on output [35].

4.2 Proposed DC motor model Parameters

Table 4.1 shows the DC motor parameters. From experimental analysis, the following parameters were derived. For simulation, the following values are used using Matlab Simulink for DC motor model.

Table 4.1: DC motor model parameter

Parameter name	Symbol with unit	Value (MPC)	Value (PID)
Armature resistance	R_a (ohms)	1.82	1.82
Armature inductance	L_a(Henry)	0.015	0.015
Load Torque	T(Nm)	10	10
Mechanical inertia	J(kg.m^2)	0.001	0.001
Damping coefficient	B(N.m.s)	0.01	0.01
Constant	$K_e\varphi$	1.64	

4.3 MPC for proposed DC motor model Simulation

Fig 4.1 shows the MPC for proposed DC motor model Simulink simulation. It shows the transfer functions after inserting the motor parameters.

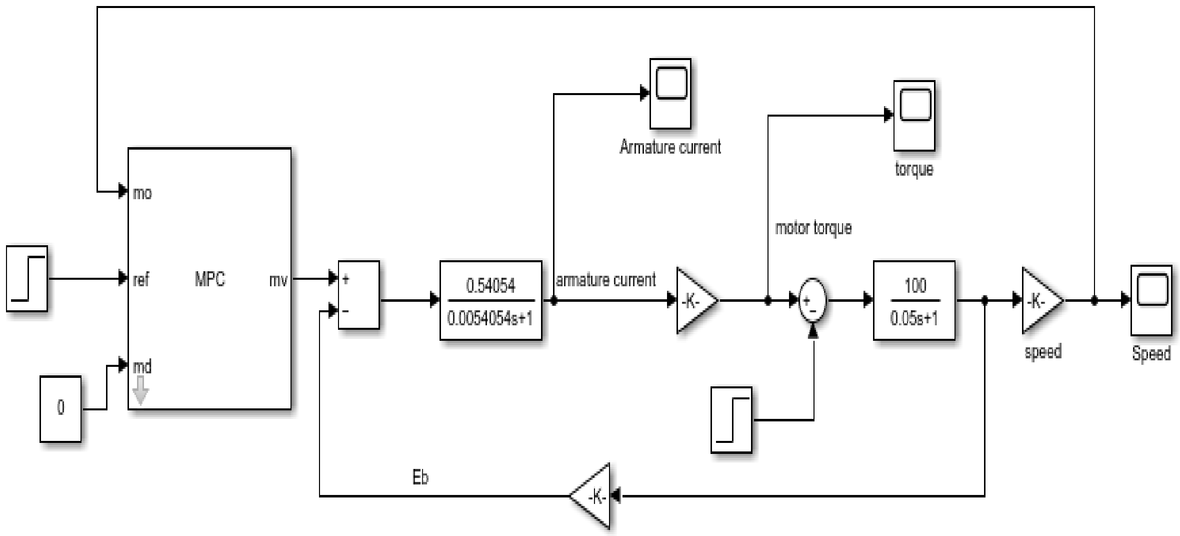

Fig 4.1: MPC for proposed DC motor model.

MPC design parameters are shown in table 4.2. Sampling time, prediction horizon, control horizon etc. parameters control the MPC controller's output with constraints.

Table 4.2: MPC design parameters

MPC Parameter	Value
Sampling time	0.005s
Prediction horizon	10
Control Horizon	02
Input constraints	–infinity to + infinity
Output constraints	–infinity to + infinity

4.3.1 Output Speed of MPC for proposed DC motor model

Fig. 4.2 shows the output speed of MPC for the proposed DC motor model at 0.01s sampling time**.** The reference speed was initially 800 RPM but changed to 1200 RPM at the interval of 5 seconds **.**

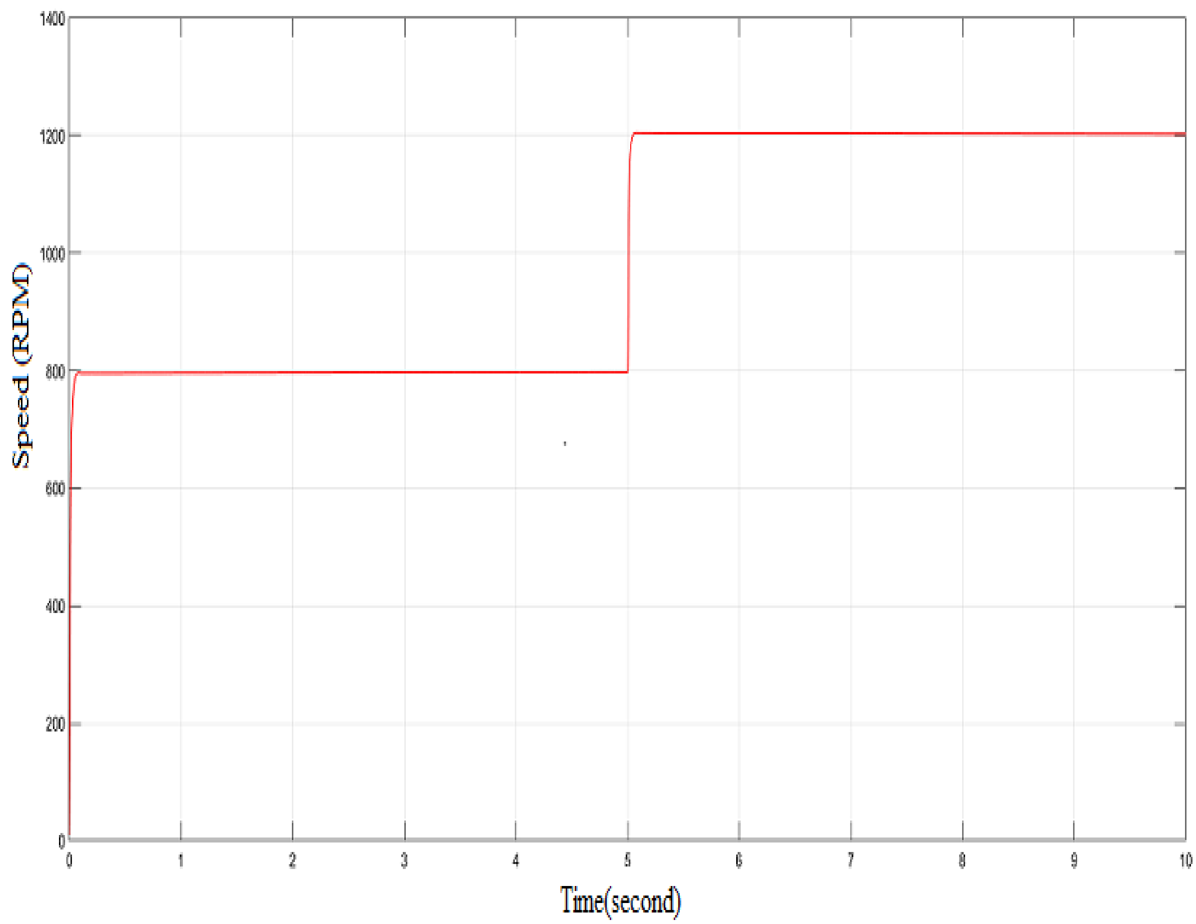

Fig 4.2: Output speed of MPC for proposed DC motor model.

4.3.2 Output Torque of MPC for proposed DC motor model

Fig. 4.3 shows the output torque of the MPC for proposed DC motor model at 0.01s sampling time**.** Initialy the starting torque was very high as there was no back emf. The reference speed was initially 800 RPM but changed to 1200 RPM at the interval of 5 seconds. As the speed changed over a 5s interval, the torque became high instantaneously in a short period of time and soon stabilized.

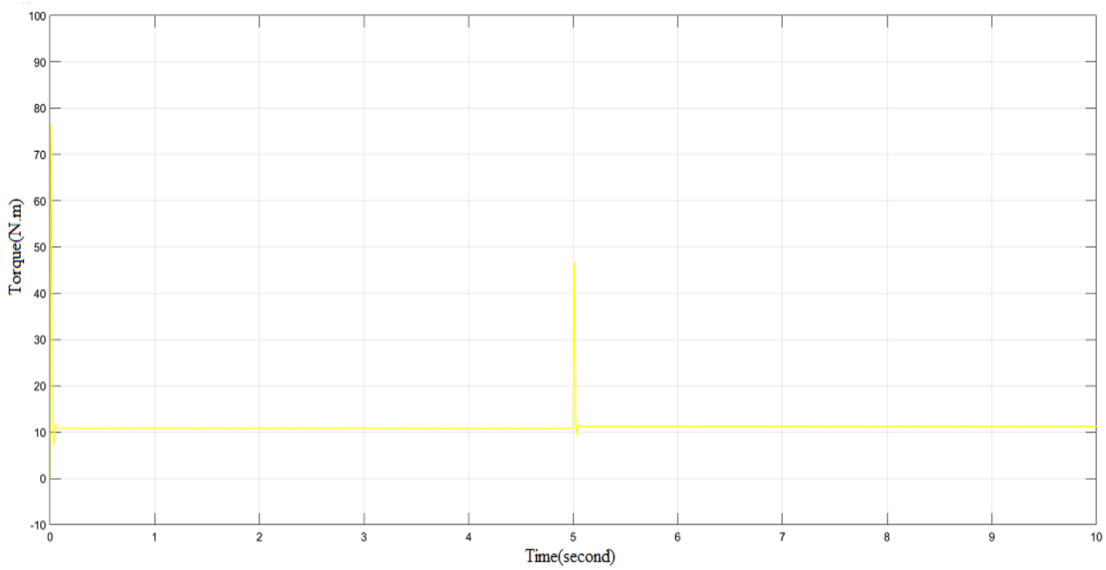

Fig 4.3: Output torque of MPC for proposed DC motor model.

4.3.3 Input Armature current of MPC for proposed DC motor model

Fig. 4.4 shows the input armature current of MPC for the proposed DC motor model. Initially the starting current was very high as there was no back emf, but it settled down rapidly with the generation of back emf. The reference speed was initially 800 RPM but changed to 1200 RPM at the interval of 5s. As the speed changed over 5s interval, the armature current became high instantaneously for a short period of time and soon stabilized.

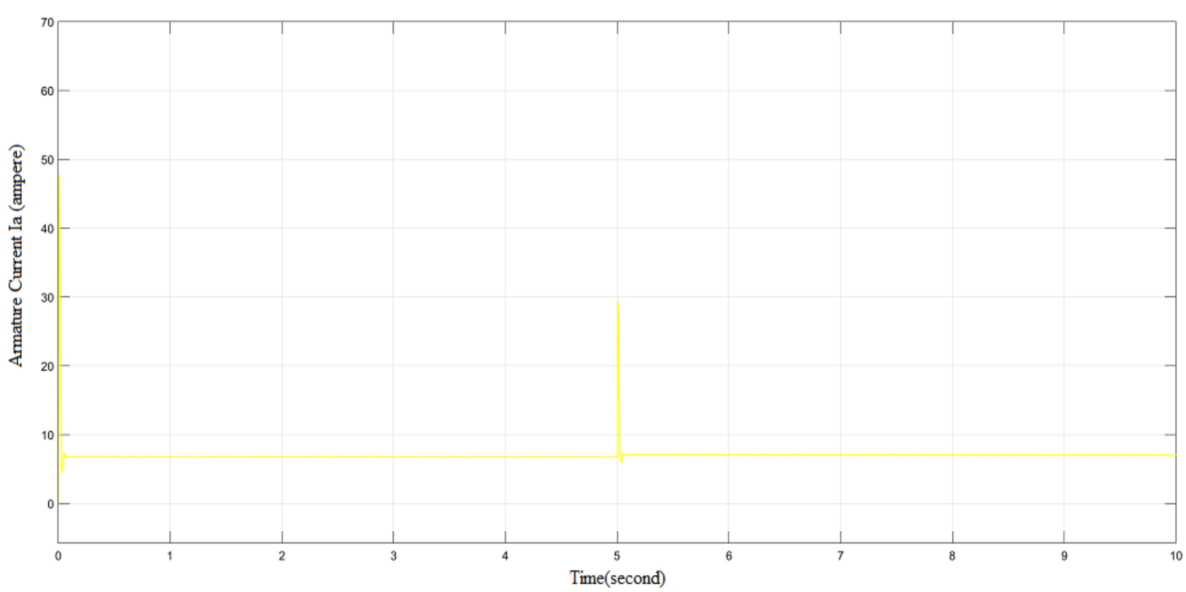

Fig 4.4: Armature current of MPC for proposed DC motor model.

4.4 DC motor model PID Simulation results

Fig 4.5 shows PID controlled Simulink simulation for DC motor model. It shows the transfer functions after inserting the motor parameters.

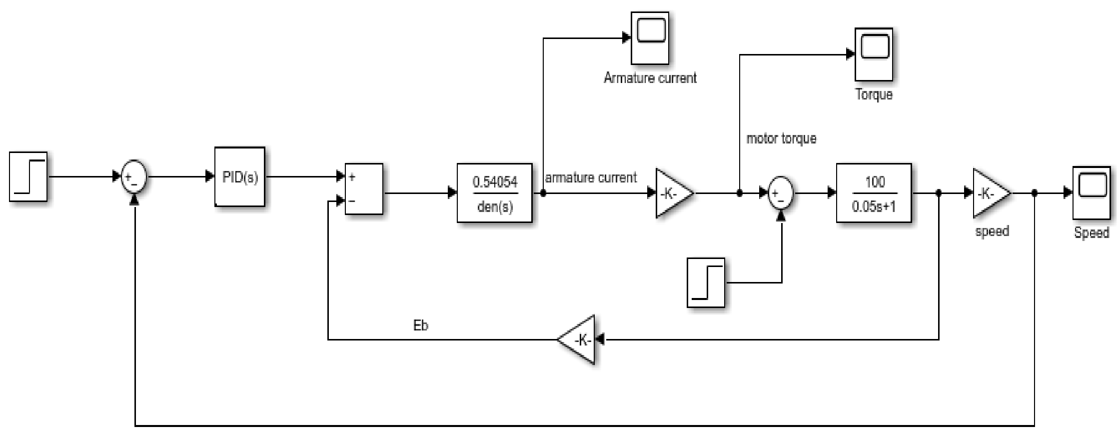

Fig 4.5: PID controller for DC motor model.

4.4.1 PID Tuning with GA

Fig 4.6 shows GA based PID controller for DC motor model. Population size is taken 50 and generation is 25. It was tuned for a better speed response with lower settling time, rise time and % of OS.

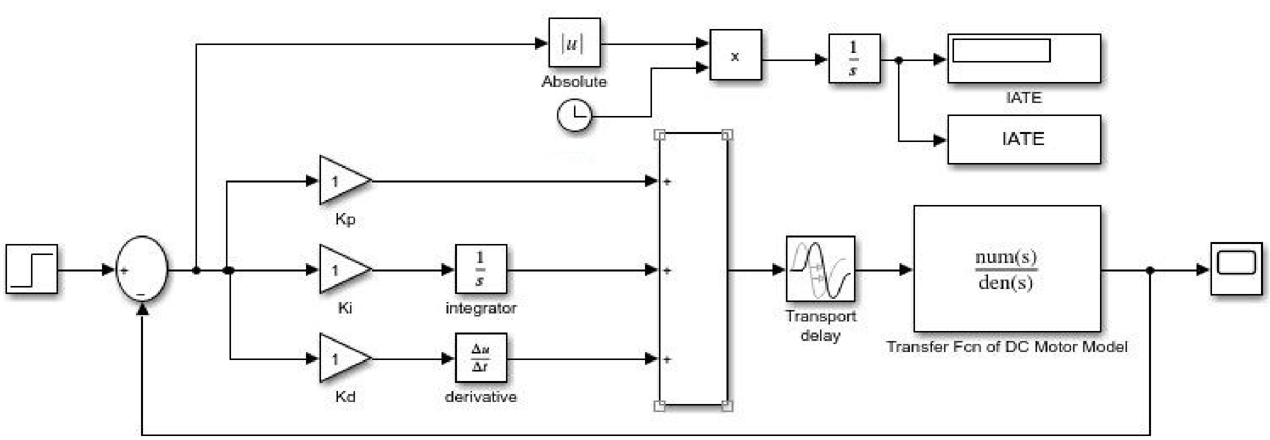

Fig 4.6: GA based PID controller for DC motor model.

Table 4.3: PID Tuning values

Controller Parameters	Tuned with GA	Auto Tuned
Proportional, P	0.275	0.14942
Integral, I	1.963	2.4837
Derivative, D	0.040	0.00403

4.4.2 Speed of PID controlled DC motor model

Fig. 4.7 shows the output speed of the PID controlled DC motor model. The reference speed was initially 800 RPM but changed to 1200 RPM at the interval of 5s.

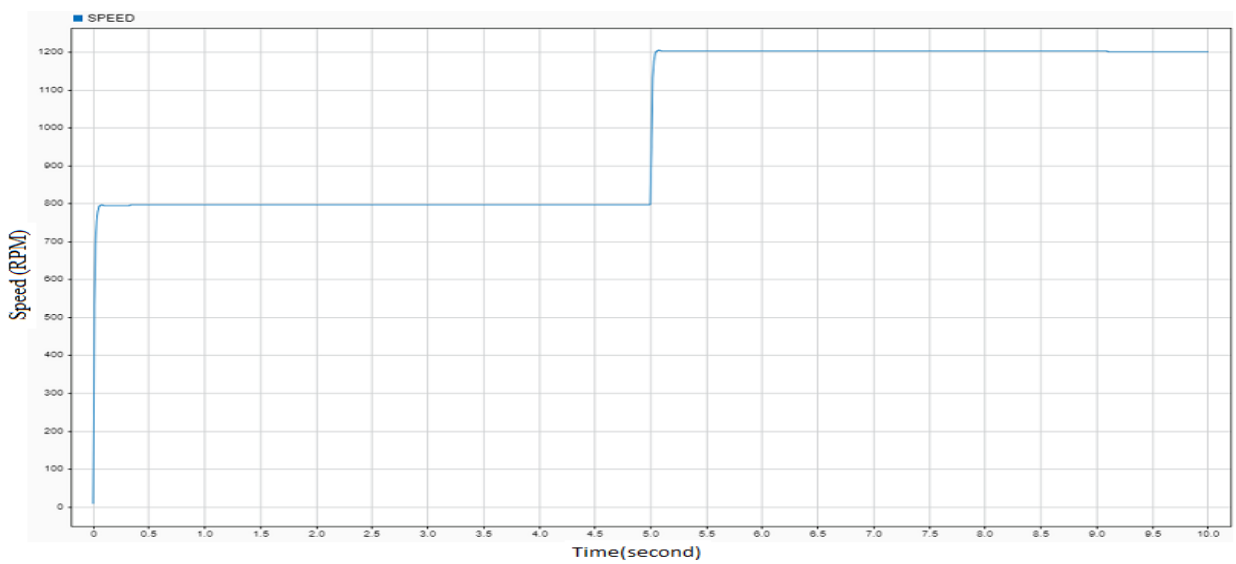

Fig 4.7: Output speed of PID controlled DC motor model.

4.4.3 Torque of PID controlled DC motor model

Fig. 4.8 shows the output torque of PID controlled DC motor model. Initially starting torque was very high, as there was no back emf but it settled down too fast. Reference speed was initially 800 RPM, but at 5s interval it was changed to 1200 RPM.

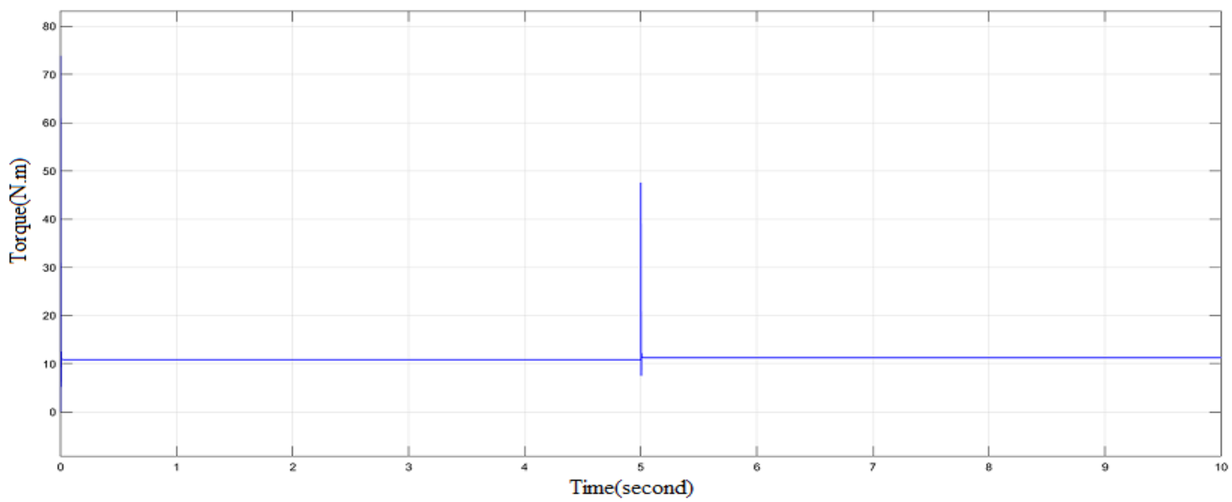

Fig 4.8: Output torque of PID controlled DC motor model.

4.4.4 Armature current of PID controlled DC motor model

Fig. 4.9 shows the input armature current of PID controlled DC motor model. Initially starting current was very high, but it settled down very fast. As the speed changed over a 5s interval, the armature current raised very high instantaneously for a short period of time and soon stabilized.

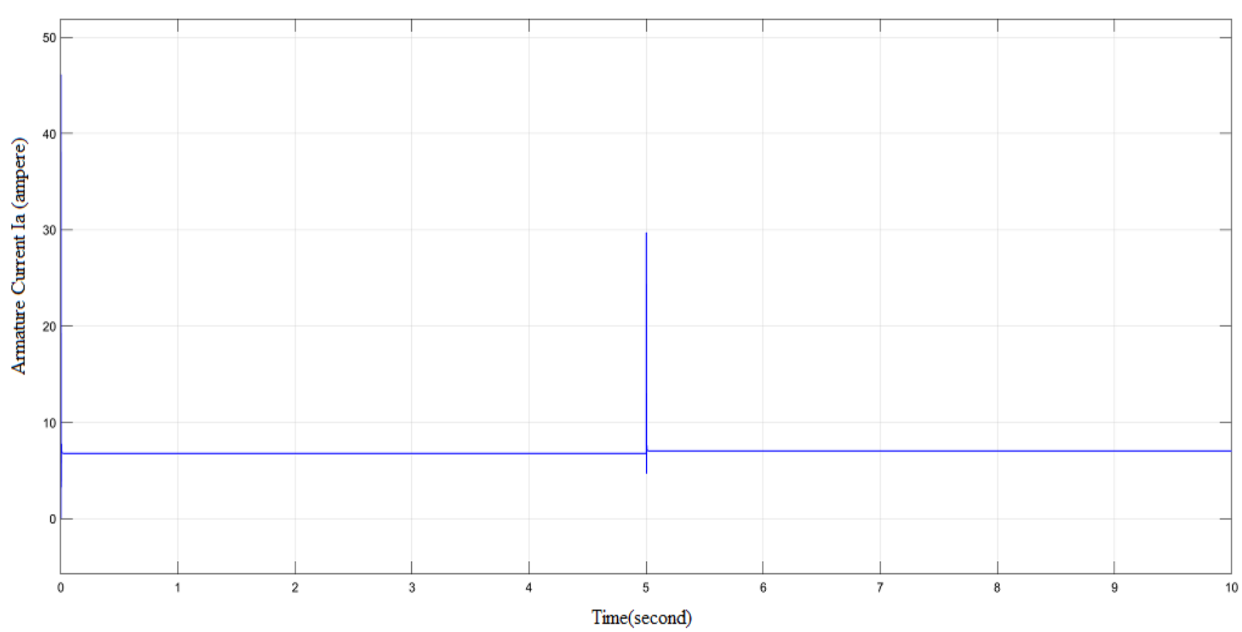

Fig 4.9: Armature current of PID controlled DC motor model.

4.5 Proposed DC motor model MPC and PID Simulation comparison

DC motor model speed, torque and armature current are determined using a MPC controller. Also, PID controller is used further to determine the motor speed, torque and armature current.

Here, a comparison will be done between the two types of controller. The reference speed was initially 800 RPM but changed to 1200 RPM at the interval of 5s. Speed, torque and armature current will be compared between these two controllers.

4.5.1 Speed comparison

Fig. 4.10 shows the speed comparison of MPC and PID for DC motor model. The reference speed was initially 800 RPM, but changed to 1200 RPM at the interval of 5s.

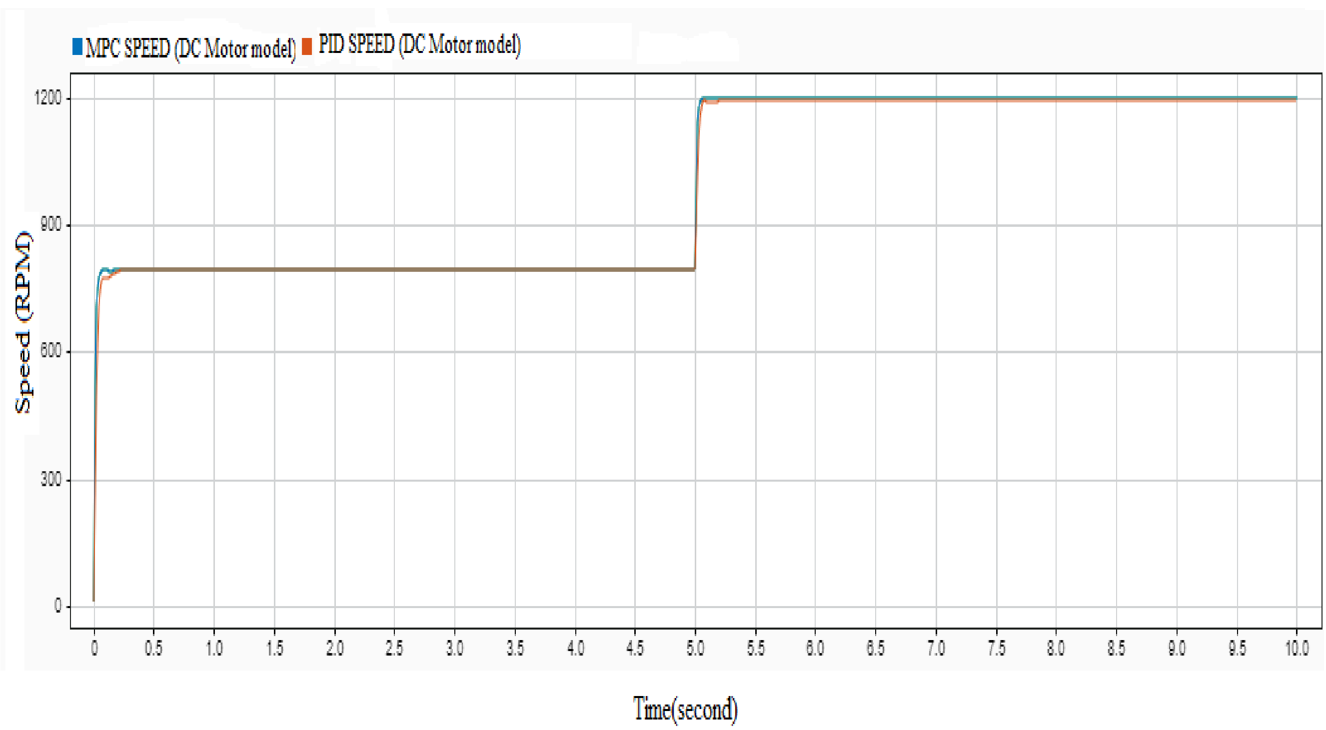

Fig 4.10: Output speed comparison of MPC and PID for DC motor model.

Fig 4.11 shows initial speed changing comparison of MPC and PID for proposed DC motor model. Here, MPC gives fast and better starting speed compare to PID. So, MPC generates faster response compared to PID.

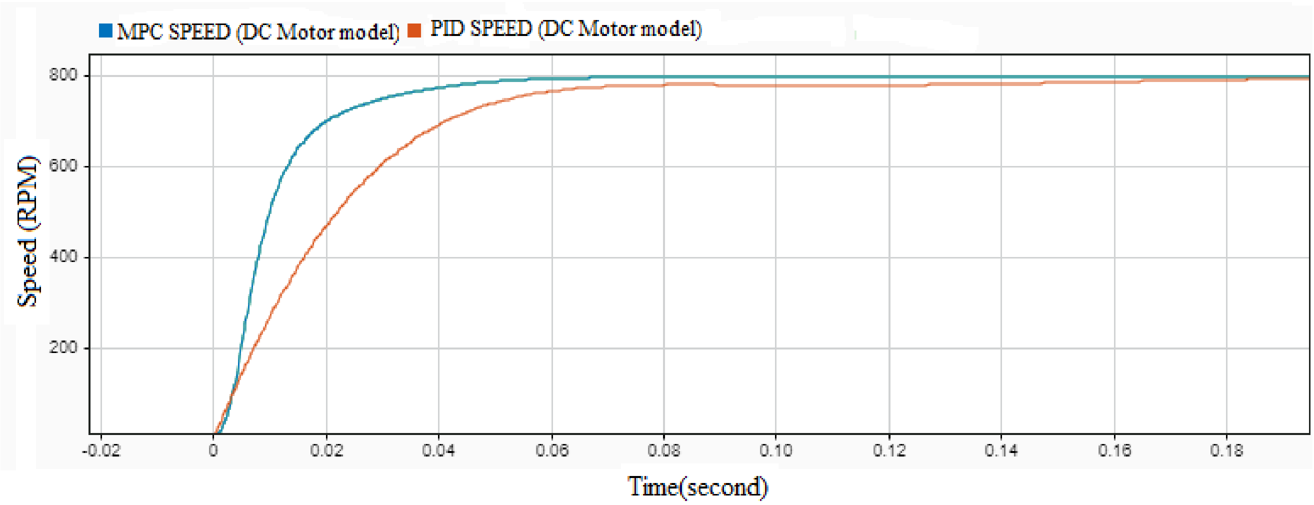

Fig 4.11: Output initial speed changing comparison of MPC and PID for proposed DC motor model.

At time interval of 5s the reference speed was increased to 1200 RPM from 800 RPM. Fig 4.12 shows the intermediate speed changing comparison of MPC and PID for the proposed DC motor model. Here, settling time and % of OS of PID is more than MPC.

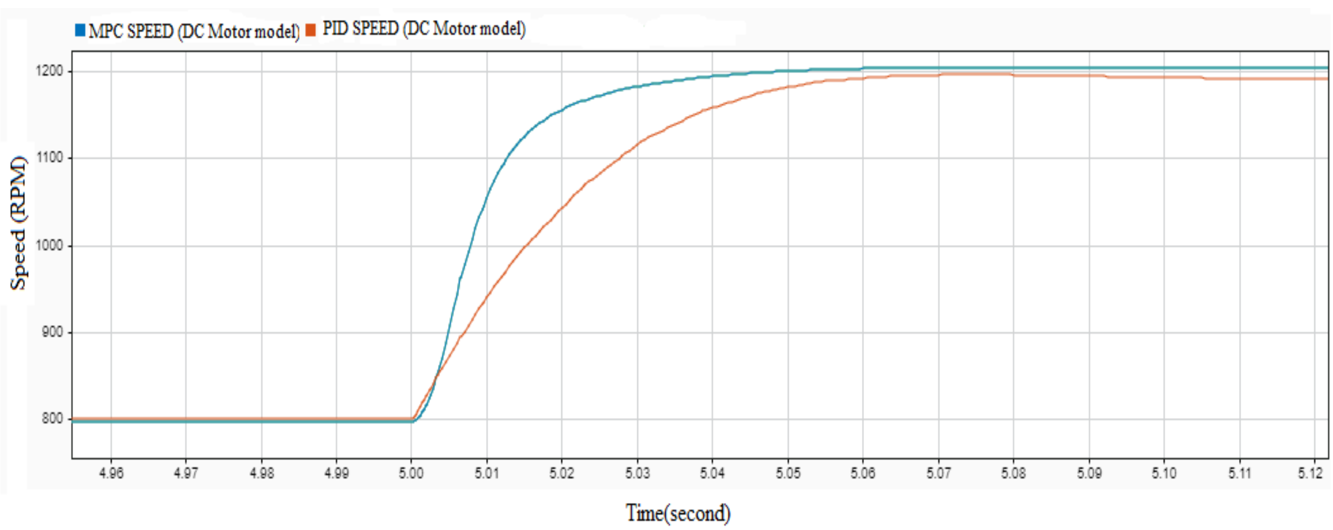

Fig 4.12: Output intermediate speed changing comparison of MPC and PID for the proposed DC motor model.

4.5.2 Armature current comparison

Fig. 4.13 shows the input armature current comparison of MPC and PID for the proposed DC motor model. The reference speed was initially 800 RPM but changed to 1200 RPM at the interval of 5s. The armature current increased to accommodate with the increased reference speed.

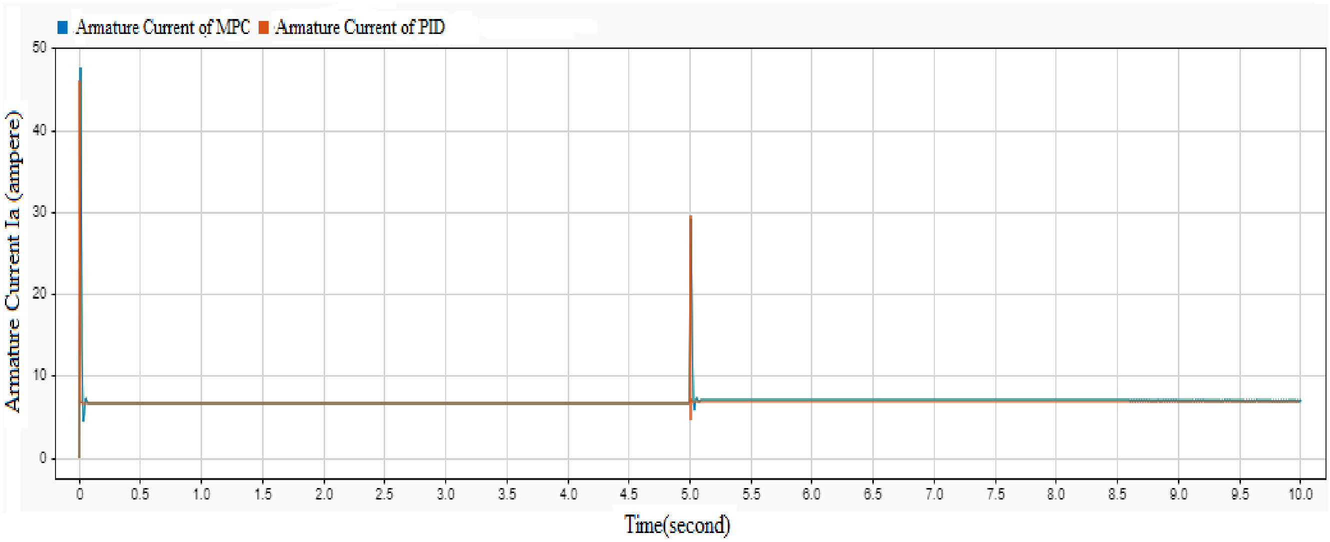

Fig 4.13: Input armature current comparison of MPC and PID for proposed DC motor model.

Initially MPC current was higher than PID controller. Fig 4.14 shows the input initial armature current changing comparison of MPC and PID for the proposed DC motor model. As MPC works faster than PID, the initial armature current of MPC is higher than PID control method.

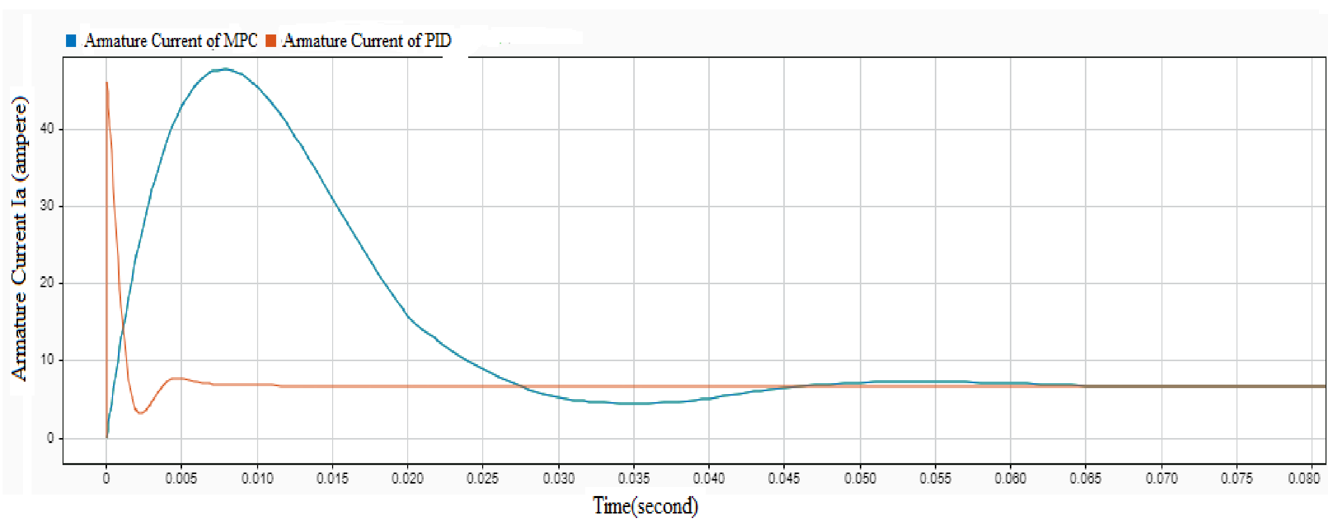

Fig 4.14: Input initial armature current changing comparison of MPC and PID for the DC motor model.

At time interval of 5s the reference speed was increased to 1200 RPM from 800 RPM. Fig 4.15 shows the intermediate armature current changing comparison of MPC and PID for the proposed DC motor model. PID took shorter time than MPC to settle down the armature current.

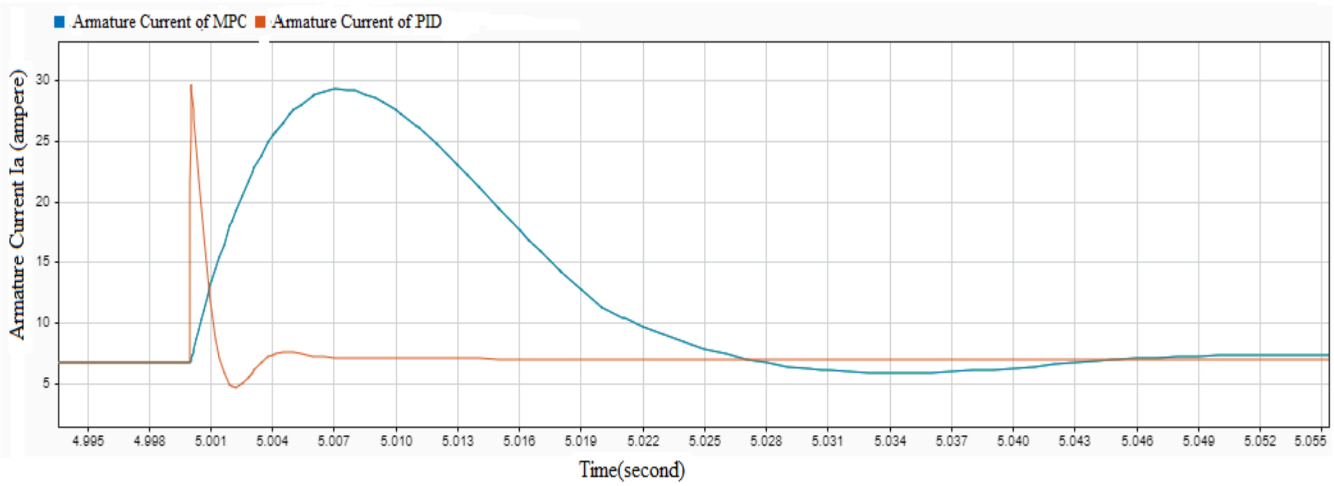

Fig 4.15: Input intermediate armature current changing comparison of MPC and PID for the DC motor model.

4.5.3 Torque comparison

Fig. 4.16 shows the output torque comparison of MPC and PID for the proposed DC motor model. The reference speed was initially 800 RPM but changed to 1200 RPM at the interval of 5s.

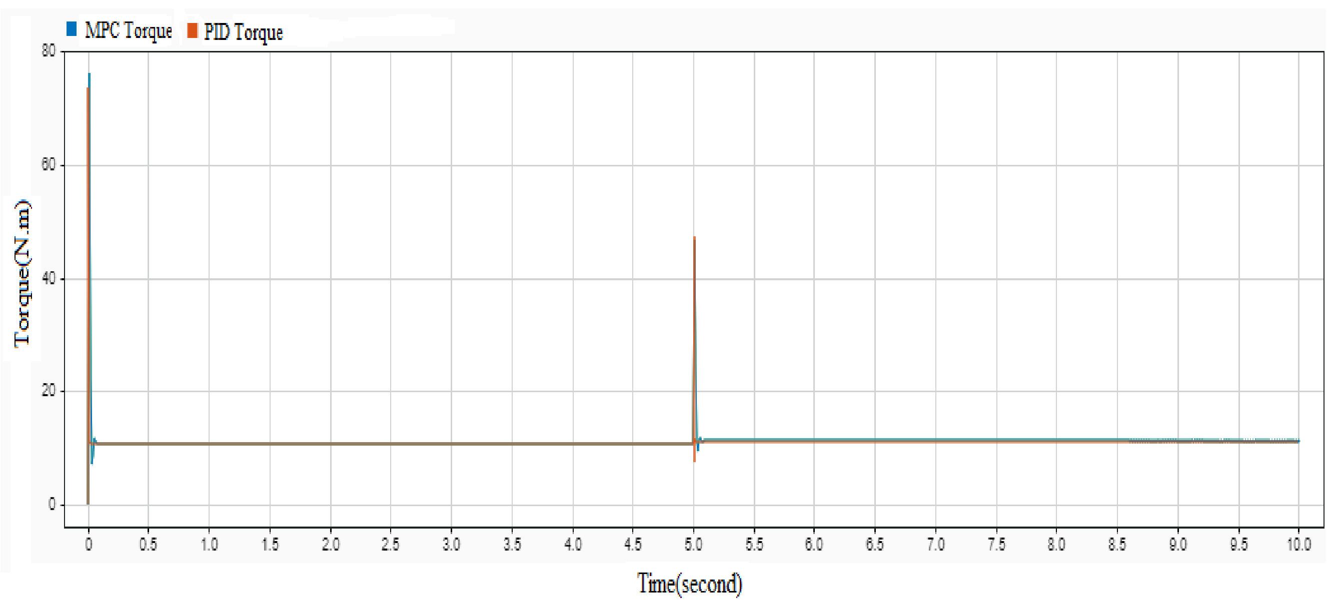

Fig 4.16: Output torque comparison of MPC and PID for proposed DC motor model.

Initially the MPC motor model torque was higher than PID controller. Fig 4.17 shows the output initial torque changing comparison of MPC and PID. As MPC takes initial armature current higher than PID, the initial torque of MPC is higher than PID control method. Torque is proportional to the armature current.

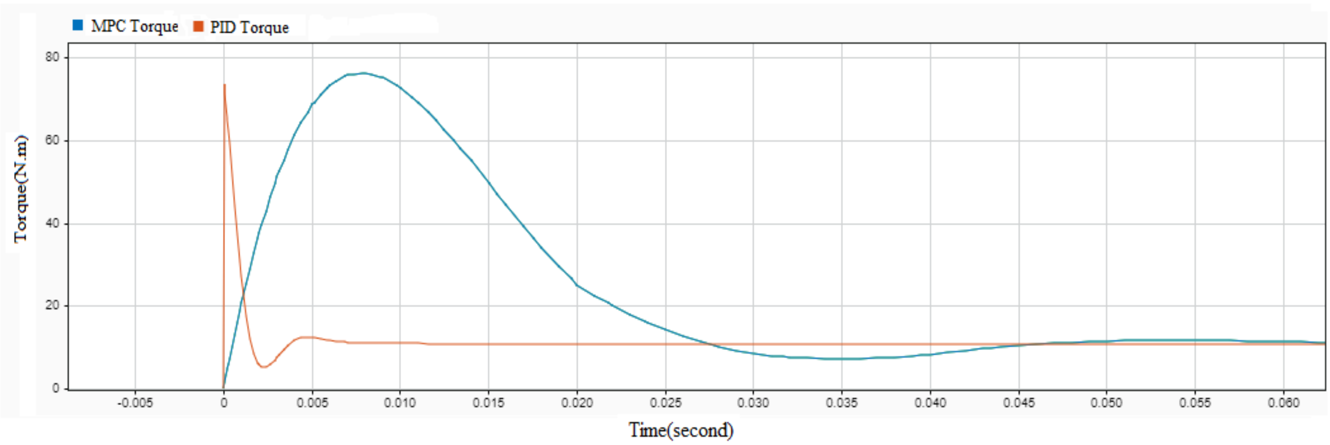

Fig 4.17: Output initial torque changing comparison of MPC and PID for the DC motor model.

At time 5s interval the reference speed was increased to 1200 RPM from 800 RPM. Fig 4.18 shows the intermediate torque changing comparison of MPC and PID for the proposed DC motor model.

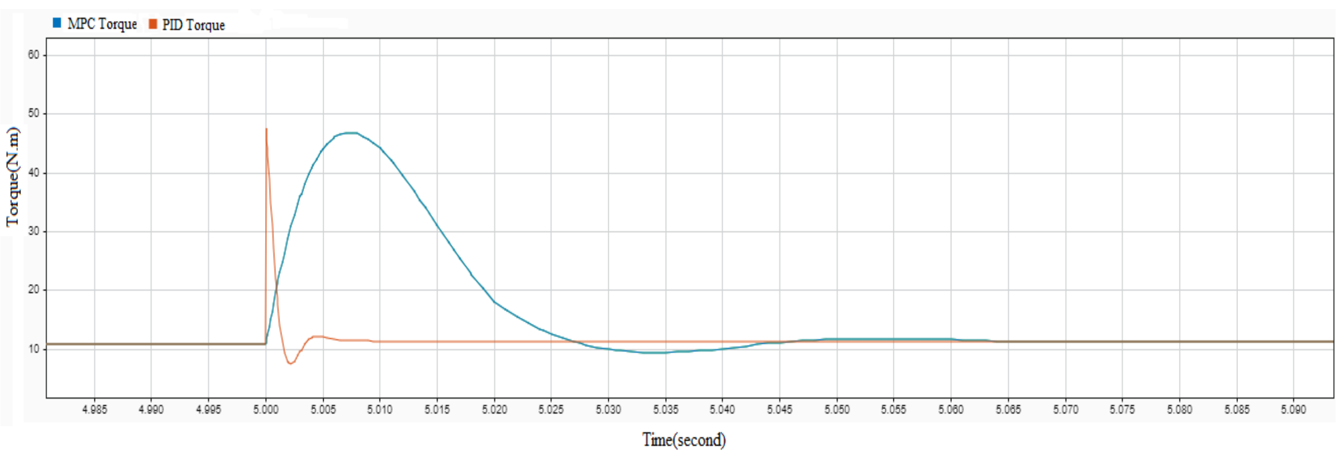

Fig 4.18: Output intermediate torque changing comparison of MPC and PID for the DC motor model.

4.5.4 Performance comparison

Rise time, settling time, and overshoot are the important performance parameters for control systems.

The rise time measures how long it takes for a step change in the input to cause the output of the system to increase from 10% to 90% of its steady-state value. A faster rise time generally indicates a more responsive system, which is desirable in many control applications [36].

The amount of time it takes for a step change in the input to cause the system's output to stabilize within a predetermined range of its steady state value is known as the settling time. A shorter settling time typically denotes an improved transient performance and a quicker system response.

The amount by which a step change in the input causes the output of the system to vary from its steady-state value before settling is known as overshoot. In some applications, a certain degree of OS may be acceptable, but too much overshoot might cause instability and oscillations in the system.

A control system with a fast rise time, short settling time, low steady-state-error and minimal overshoot is desirable, as it indicates a more responsive and stable system [37].

Table 4.4: Performance comparison of MPC and PID

Comparison Horizon	PID	Research paper PID [8]	MPC
Rise time (Second)	0.046	0.0571	0.042
Peak time (second)	0.064	0.0657	0.054
Settling time(second)	0.127	0.0974	0.055
% of OS	5.22	6.88	NA

Chart 4.1 : Performance comparison of MPC and PID

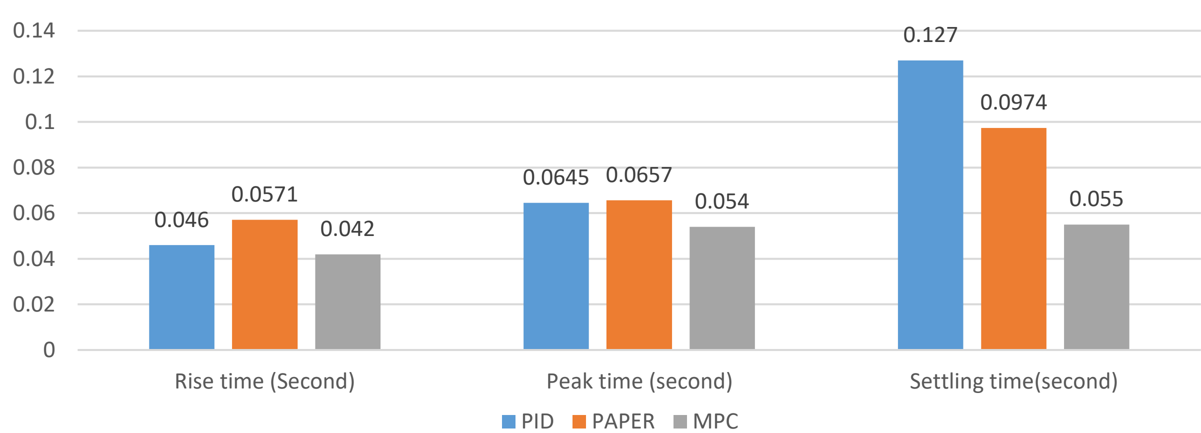

Table 4.4 shows the performance comparison of MPC and PID controller of the proposed DC motor model. MPC gives a faster rise time, shorter settling time and no overshoot than PID controller. Peak speed, armature current and torque comparison between PID and MPC is shown in the Table 4.5. Peak speed of MPC is less than PID as it has less overshoot. But, armature current and torque generated by MPC controller are higher than PID to compensate with fast response.

Table 4.5: PID vs MPC peak speed, armature current and torque comparison.

Comparison Horizon	PID	MPC
Peak Speed (RPM)	841	799
Peak Armature current(A)	45	46.5
Peak Torque(N. m)	74.5	77

4.6 MPC for built-in DC motor model Simulation

Matlab Simulink software has a built-in DC machine model with field control method. Fig 4.19 shows the MPC for built-in DC motor Simulink simulation model.

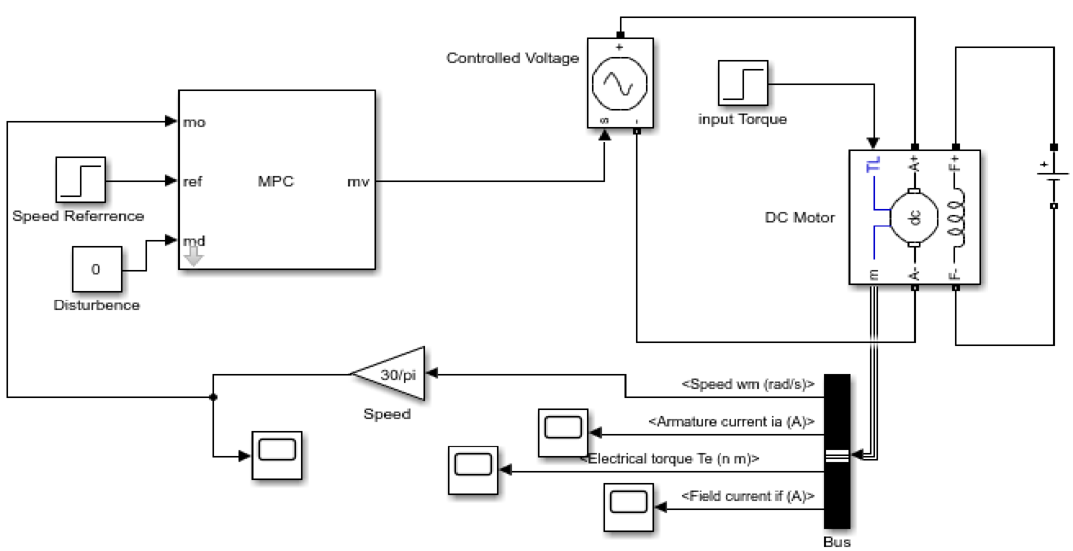

Fig 4.19: MPC for built-in DC motor model.

4.6.1 Built-in DC motor model parameter

Table 4.6 shows the motor parameters. The derived parameters from experimental analysis are utilized in the simulation using Matlab Simulink for the built-in DC motor model.

Table 4.6: Simulink built-in DC motor parameter

Parameter name	Symbol with unit	Value
Armature resistance	R_a (ohms)	1.82
Armature inductance	L_a(Henry)	0.015
Torque	T(Nm)	10
Mechanical inertia	J(kg.m^2)	0.001
Damping coefficient	B(N.m.s)	0.01

4.6.2 Speed comparison of MPC for proposed DC motor model and built-in DC motor model

Fig. 4.20 shows the speed comparison of MPC for proposed DC motor model and built-in DC motor model. The reference speed was initially 800 RPM but changed to 1200 RPM at the interval of 5s.

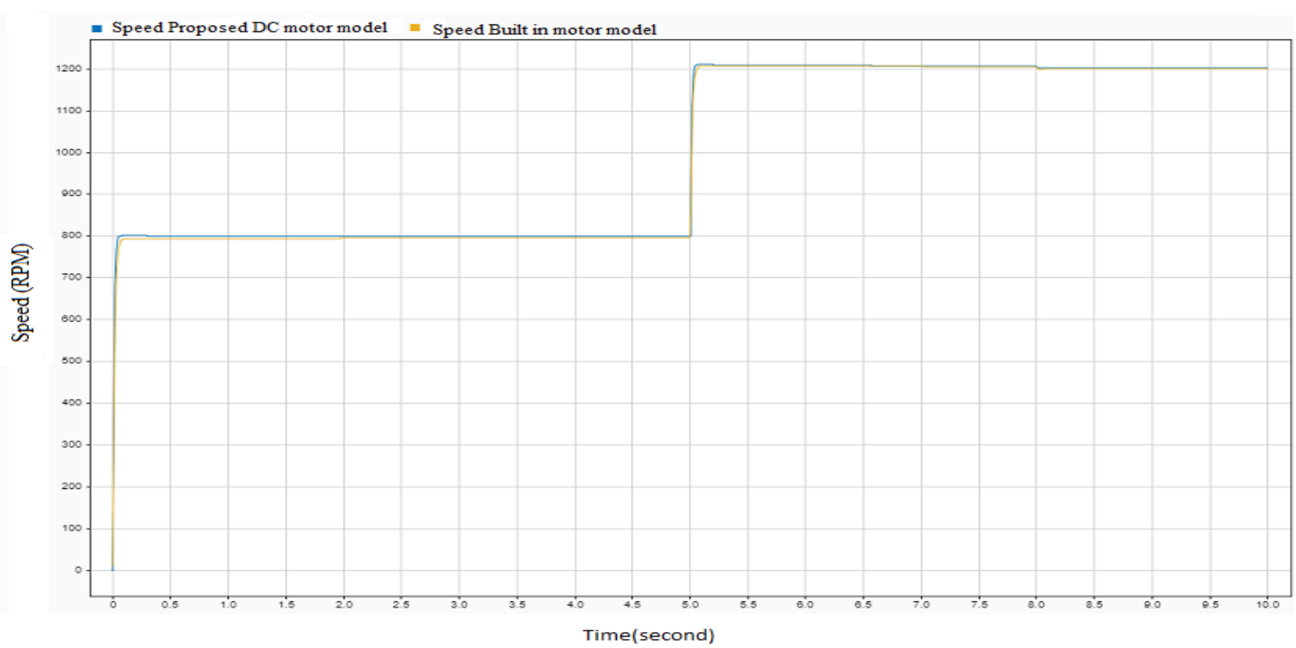

Fig 4.20: Output speed comparison of MPC for proposed DC motor model and built-in DC motor model.

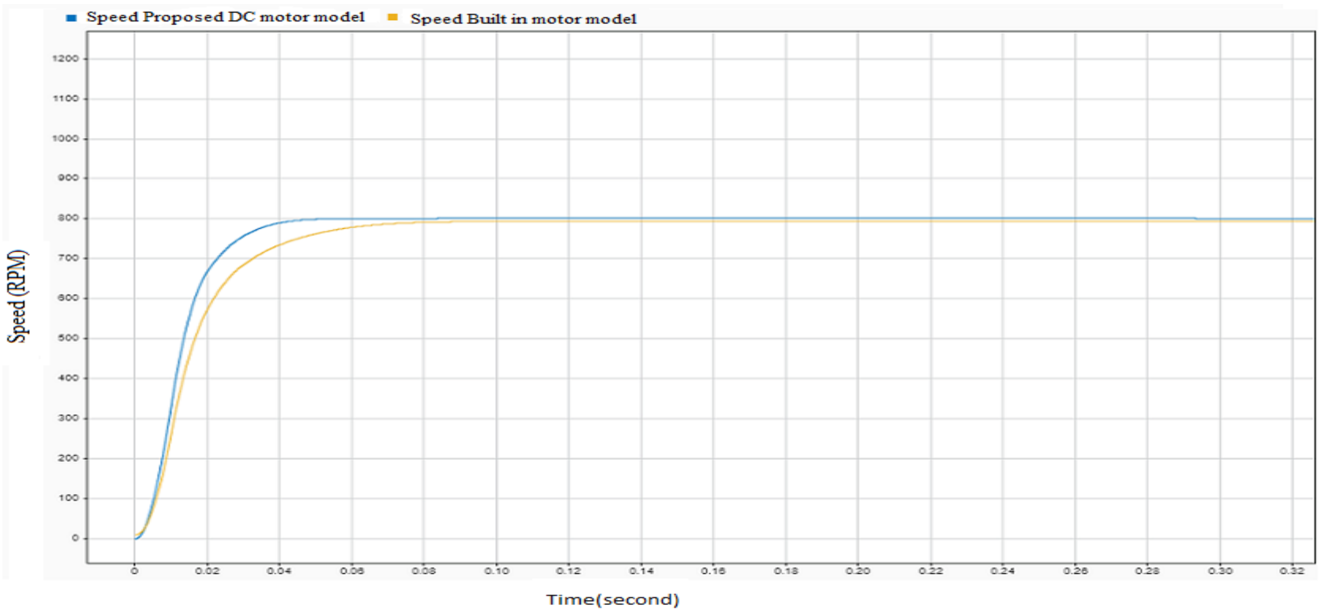

Fig 4.21: Output initial speed changing comparison of MPC for proposed model and built-in model.

Fig 4.21 shows initial speed changing comparison of MPC for proposed DC motor model and built-in DC motor model. At time interval of 5s the reference speed was increased to 1200 RPM from 800 RPM. Fig 4.22 shows the intermediate speed changing comparison of proposed of MPC for DC motor model and built-in model.

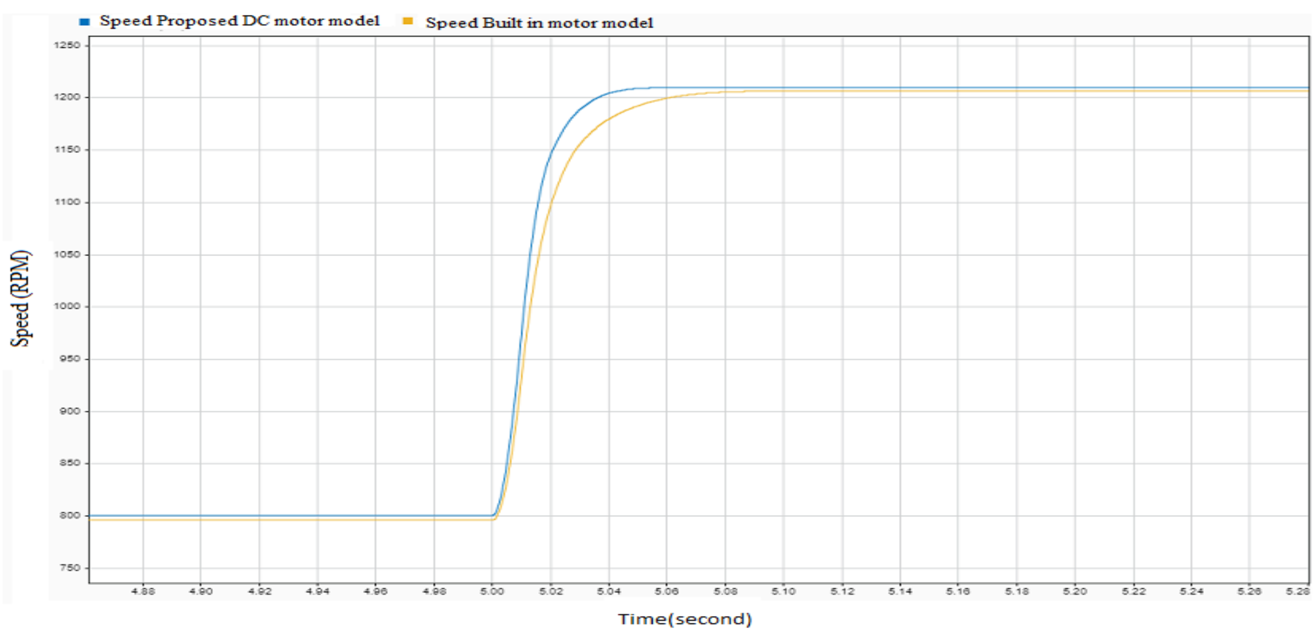

Fig 4.22: Output intermediate speed changing comparison of MPC for proposed DC model and built-in model.

4.6.3 Armature current comparison of MPC for proposed DC motor model and Built-in DC motor model

Fig. 4.23 shows the input armature current comparison of MPC for proposed DC motor model and built-in DC motor model. Initially the MPC built-in DC motor model current is slightly higher than proposed DC motor model. Fig 4.24 shows the input initial armature current changing comparison of MPC for proposed DC motor model and built-in DC motor model.

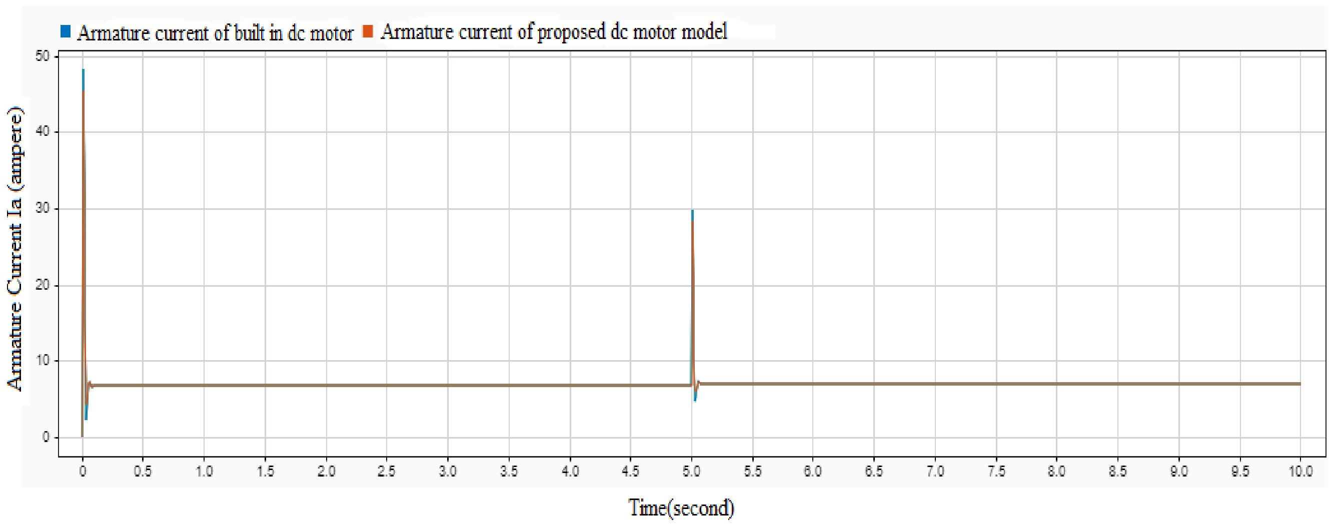

Fig 4.23: Input armature current comparison of MPC for proposed DC motor model and built-in DC motor.

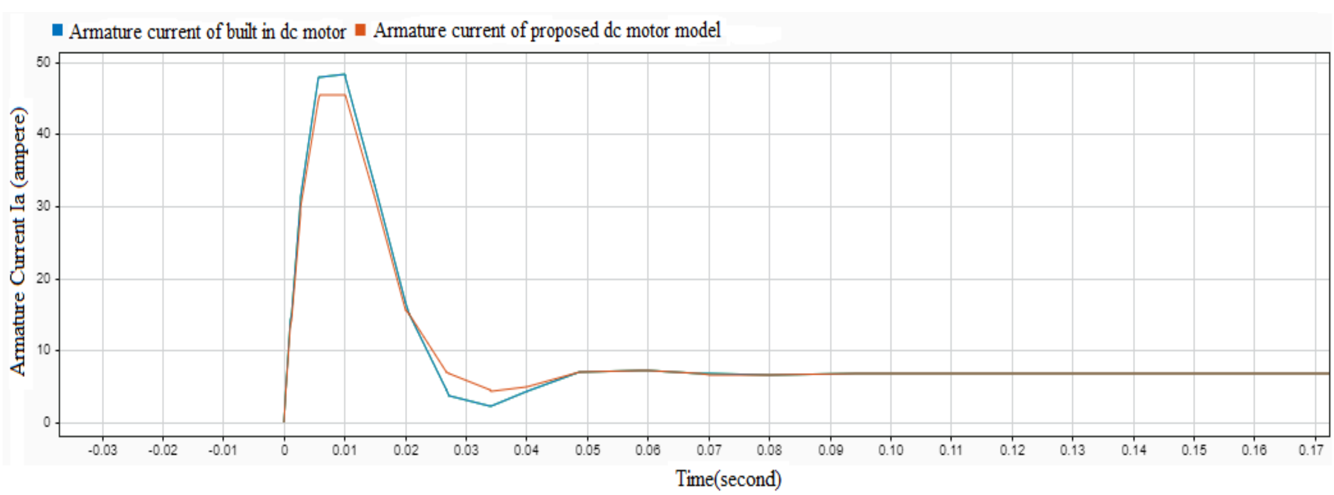

Fig 4.24: Input initial armature current changing comparison of MPC for proposed DC motor model and built-in DC motor model.

At time interval of 5s the reference speed was increased to 1200 RPM from 800 RPM. Fig 4.25 shows the intermediate armature current changing comparison of MPC for proposed DC motor model and built-in model.

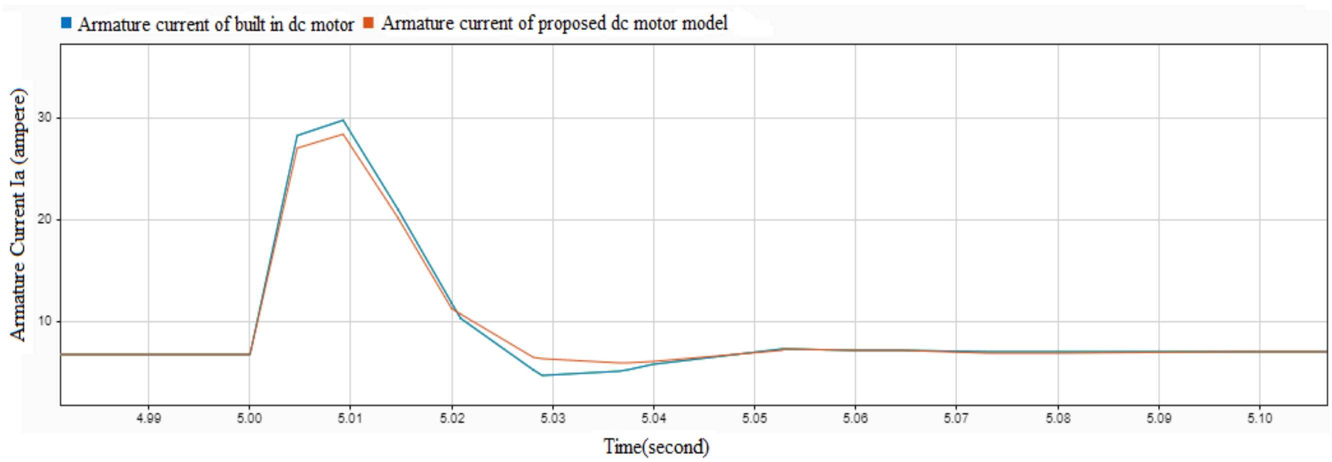

Fig 4.25: Output intermediate armature current changing comparison of MPC for proposed DC motor model and built-in DC motor model.

4.6.4 Torque comparison of MPC for proposed DC motor model and Built-in DC motor model

Fig. 4.26 shows the output torque comparison of MPC for proposed DC motor model and built-in DC motor model. The reference speed was initially 800 RPM but changed to 1200 RPM at the interval of 5s.

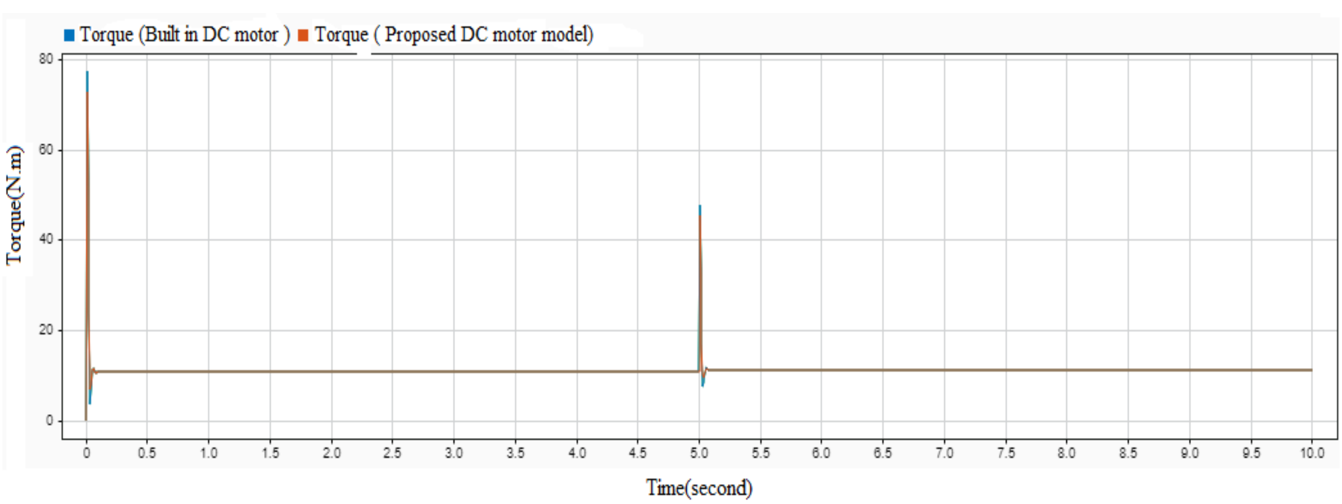

Fig 4.26: Output torque comparison of MPC for proposed DC model and built-in DC motor model.

Fig 4.27 shows the initial torque changing comparison of MPC for the proposed DC motor model and built-in DC motor model. The intermediate torque of the MPC for proposed DC motor model is slightly higher than built-in DC model. Fig 4.28 shows the intermediate torque changing comparison.

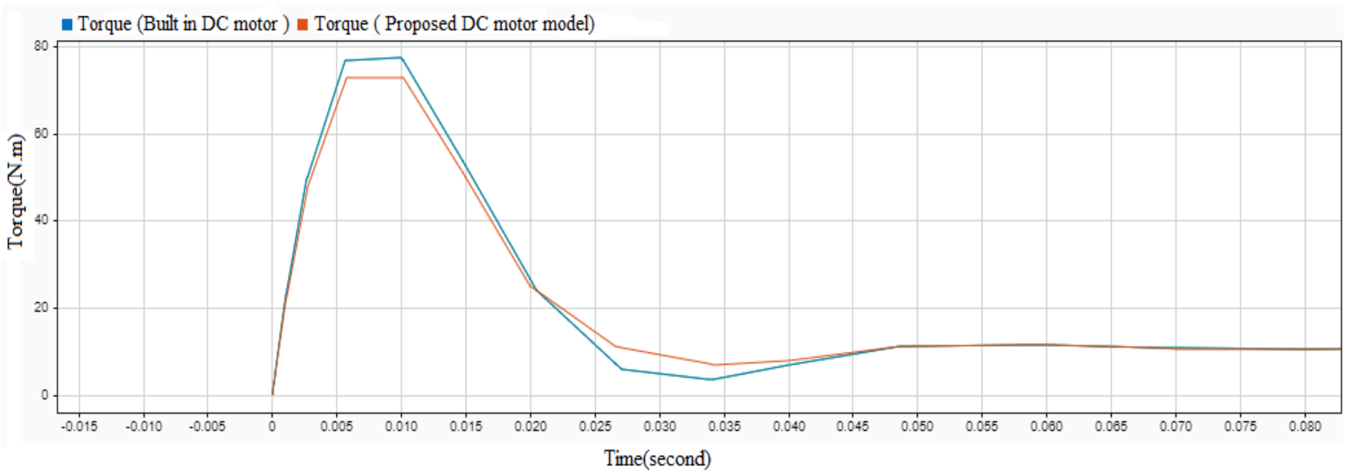

Fig 4.27: Initial torque changing comparison of MPC for proposed DC motor model and built-in DC model.

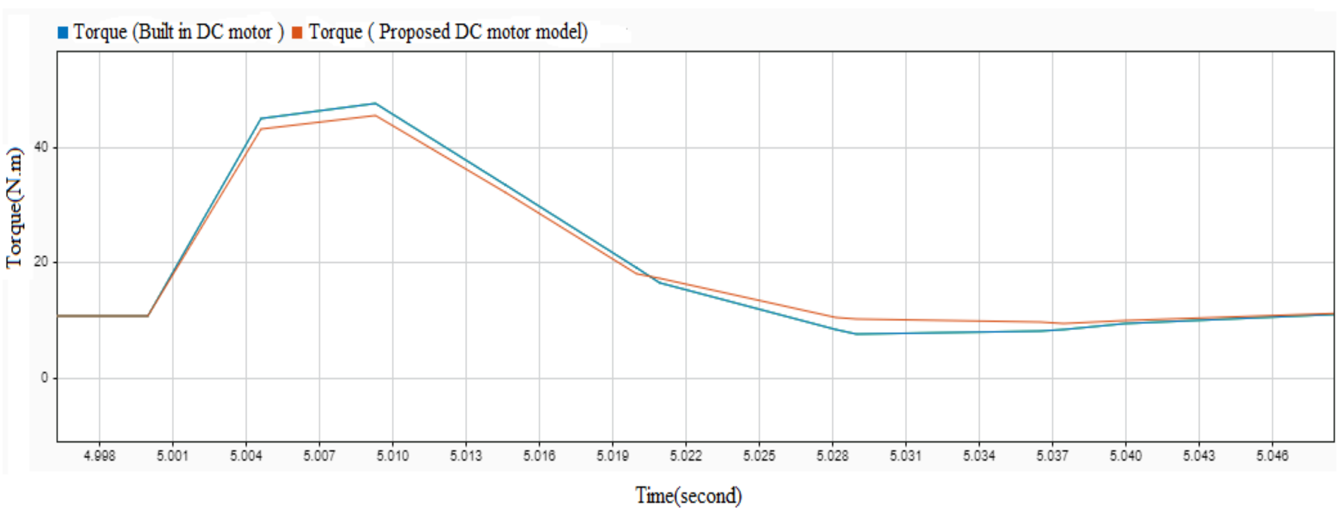

Fig 4.28: Output torque changing comparison of MPC for proposed DC motor and built-in DC motor model.

4.6.5 Overall comparison of MPC for proposed DC motor model and Built-in DC motor model

A control system with a fast rise time, short settling time, low steady-state-error, and minimal overshoot is desirable, as it indicates a more responsive and stable system. The fundamental idea behind MPC is to forecast the DC motor's behavior over a predetermined time frame utilizing a mathematical model of motor. The model can take into account the effects of torque, speed, voltage, and current, as well as any other relevant factors. The predictions are then used to optimize the control inputs to achieve a desired output [38].

Table 4.7 shows the rise time, settling time, peak armature current and torque comparison between MPC for proposed DC motor model and built-in DC motor model.

Table 4.7 :Comparison between MPC for proposed DC motor and built-in DC motor model.

Comparison Horizon	MPC for proposed DC Motor Model	MPC for Built-in Motor Model
Rise time (second)	0.042	0.048
Settling time(second)	0.055	0.071
Peak Armature current (A)	46.5	48
Peak Torque(N.m)	77	79.5

Proposed model gives rise time, settling time, peak armature current and torque close to the built-in model. From the table, proposed DC motor model can be verified as the difference is minimal.

4.7 Sampling rate Comparison

When using MPC, the sampling time is a crucial parameter that controls how frequently the control inputs are updated depending on the system's current state. The choice of sampling time depends on the system's dynamics, accuracy of the model, and computational resources available. In general, the sample time should be large enough to prevent excessive processing overhead while still being small enough to handle the system's dynamics. The dominant time constant in this case is often the electrical circuit's time constant, which is normally in the range of milliseconds. Therefore, a few milliseconds of sample time would be suitable for MPC control of a DC motor.

If the sampling period is too small, the computational load may be too high, leading to slow execution or instability. Poor performance could come from the control inputs if the sample period is too long.

The choice of sampling time for MPC depends on a balance between the accuracy of the model, the computational resources available, and the desired performance of the control system. Clock speed of the processor should be large enough to do the computation.

4.7.1 Speed Comparison of MPC for the proposed DC motor model at sampling time 0.005s and 0.01s

Fig. 4.29 shows the speed comparison of MPC for the proposed DC motor model at sampling time 0.005s and 0.01s. The reference speed was initially 800 RPM but changed to 1200 RPM at the interval of 5s. The computional time increased with sampling time.

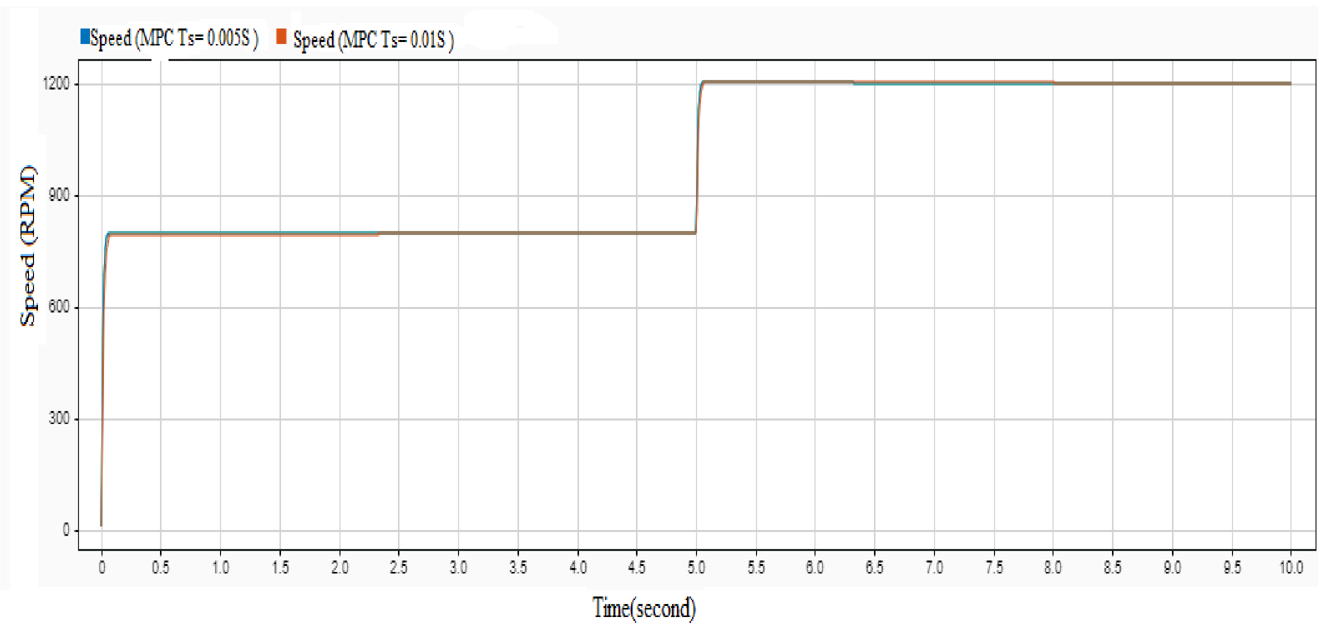

Fig 4.29: Output speed comparison of MPC for proposed DC motor model at 0.005s and 0.01s sampling time.

Fig 4.30 shows initial speed changing comparison of MPC for proposed DC motor model at 0.005s and 0.01s sampling time. Here, MPC at 0.005s sampling time gives fast and better starting speed compare to MPC at 0.01s sampling time. So, sampling time reduction generates faster response.

At time interval of 5s the reference speed was increased to 1200 RPM from 800 RPM. Fig 4.31 shows the intermediate speed changing comparison of MPC for the proposed DC motor model at 0.005s and 0.01s sampling time. Sampling time reduction gives faster settling time, rise time of the system model.

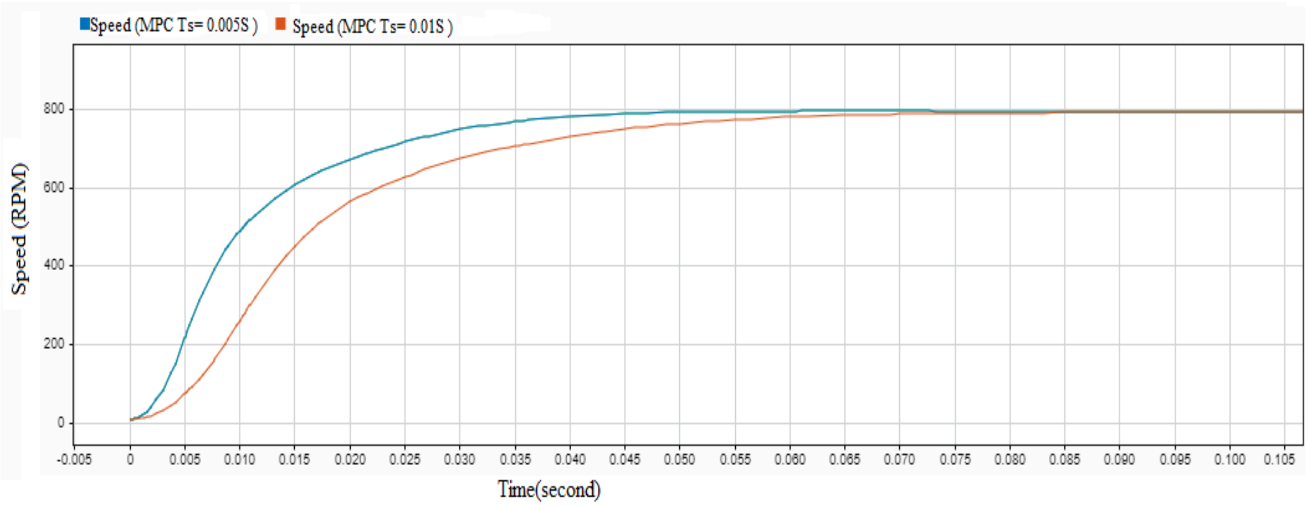

Fig 4.30: Output initial speed comparison of proposed model at 0.005s and 0.01s sampling time.

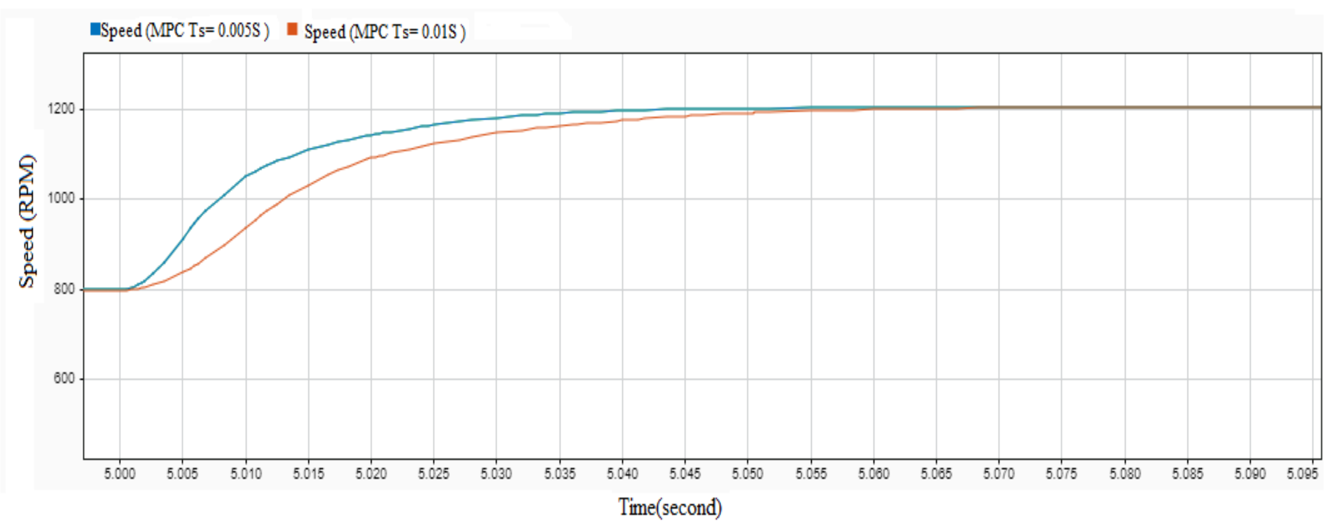

Fig 4.31: Output intermediate speed changing comparison of proposed model at 0.005s and 0.01s sampling time.

4.7.2 Speed Comparison of MPC for the proposed DC motor model at sampling time 0.02s and 0.01s

Fig. 4.32 shows the speed comparison of MPC for the proposed DC motor model at sampling time 0.01s and 0.02s. The reference speed was initially 800 RPM but changed to 1200 RPM at the interval of 5s.

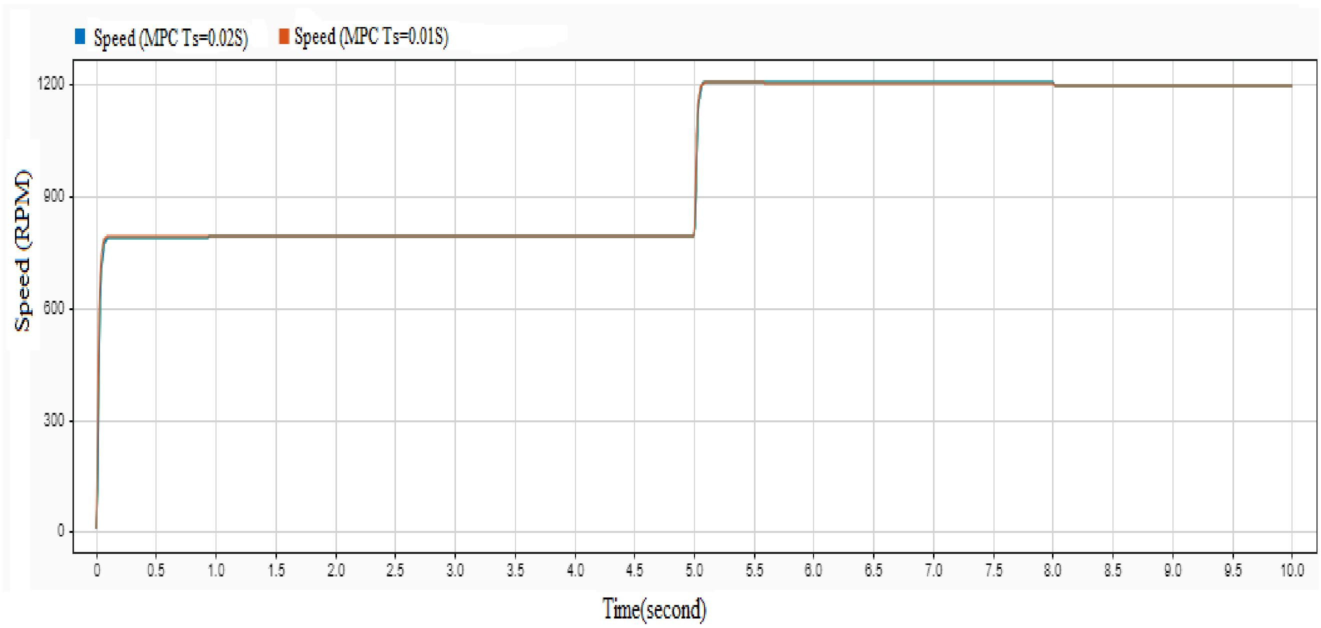

Fig 4.32: Output speed comparison of MPC for proposed DC motor model at 0.02s and 0.01s sampling time.

The computional time increases with sampling time. Fig 4.33 shows initial speed changing comparison of MPC for the proposed DC motor model at 0.01s and 0.02s sampling time. Here, MPC at 0.01s sampling time has given fast and better starting speed compared to MPC at 0.02s sampling time.

At time the interval 5s the reference speed was increased to 1200 RPM from 800 RPM. Fig 4.34 shows the intermediate speed changing comparison of MPC for the proposed DC motor model at 0.01s and 0.02s sampling time. Hence, sampling time reduction gives faster settling time and rise time of the system.

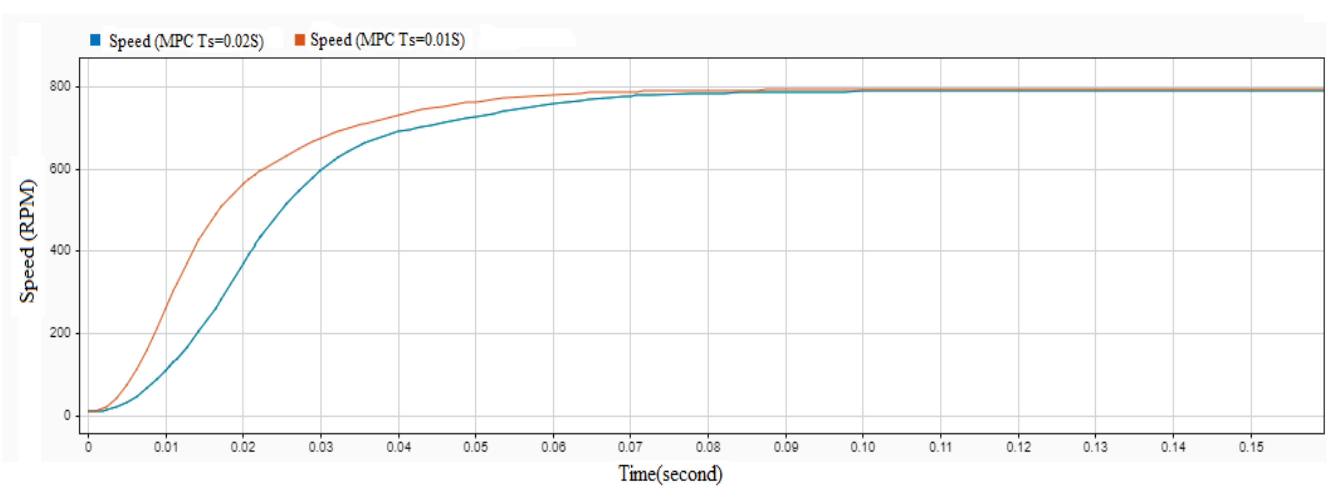

Fig 4.33: Output initial speed comparison of MPC for proposed DC motor at 0.02s and 0.01s sampling time.

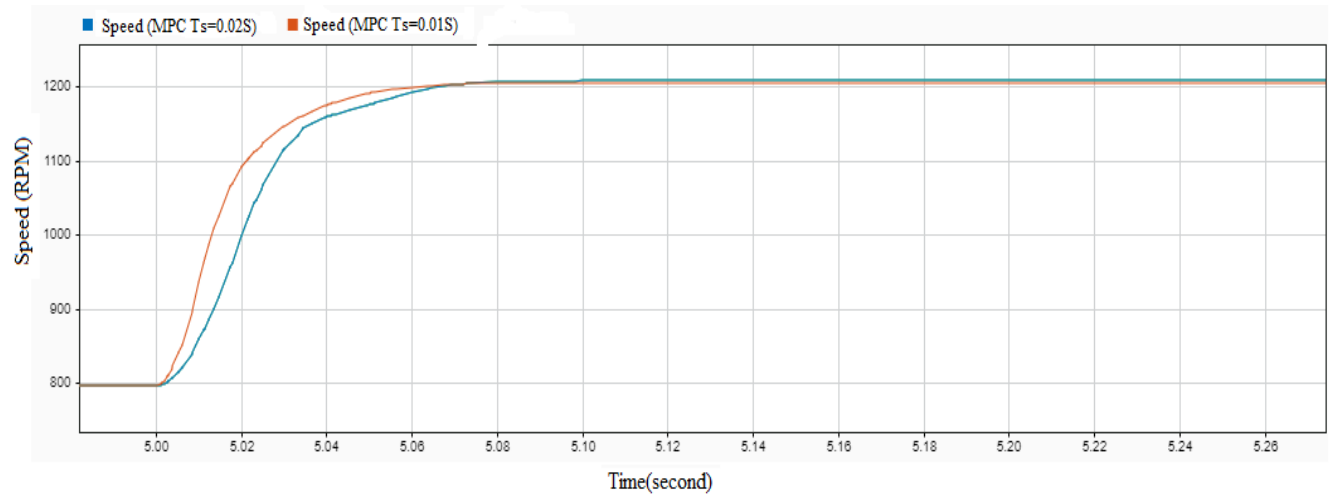

Fig 4.34: Output speed changing comparison of MPC for proposed DC motor model at 0.02s and 0.01s sampling time.

4.7.3 Armature current comparison of MPC for the proposed DC motor model at sampling time 0.005s and 0.02s

Fig. 4.35 shows the input armature current comparison of MPC for the proposed DC motor model at 0.005s and 0.02s sampling time. The reference speed was initially 800 RPM but changed to 1200 RPM at the interval of 5 seconds. Fig 4.36 shows initial armature current changing comparison of MPC for proposed DC motor model at 0.005s and 0.02s sampling time.

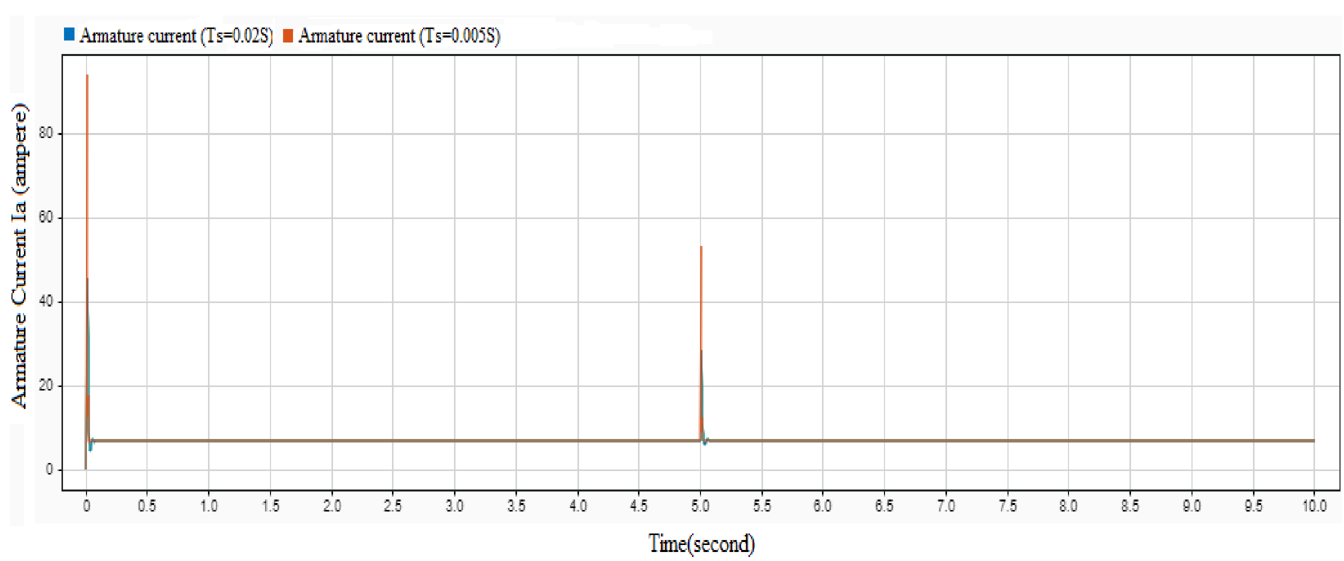

Fig. 4.35: Input armature current comparison of MPC for DC motor at 0.005s and 0.02s sampling time.

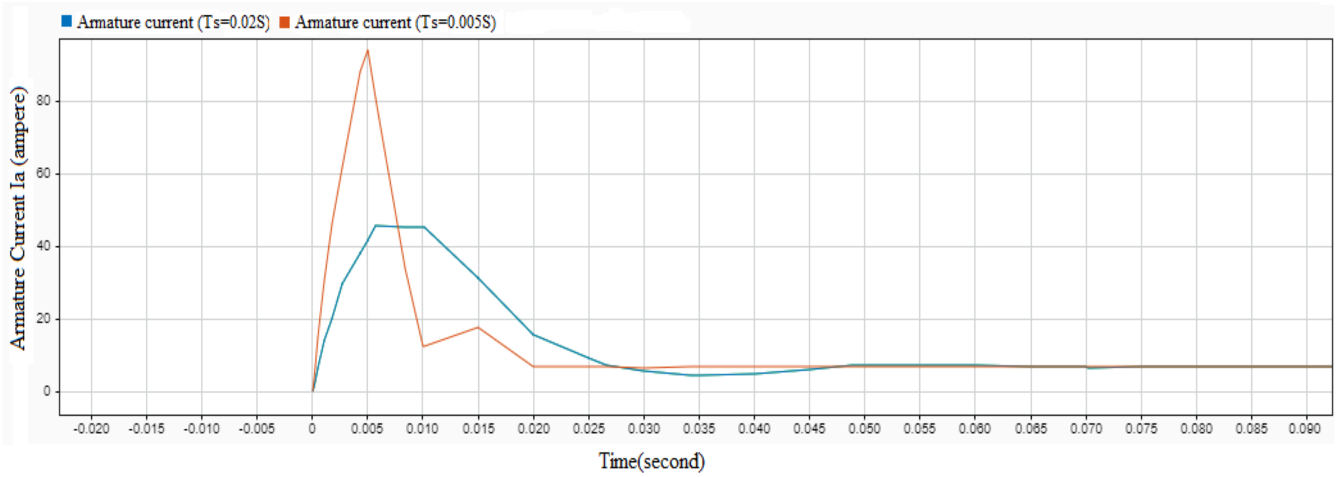

Fig 4.36: Initial armature current changing comparison at sample time 0.005s and 0.02s.

Fig 4.37 shows intermediate armature current changing comparison of MPC for proposed model at 0.005s and 0.02s sampling time. As sampling time increases, the peak value of the armature current decreases.

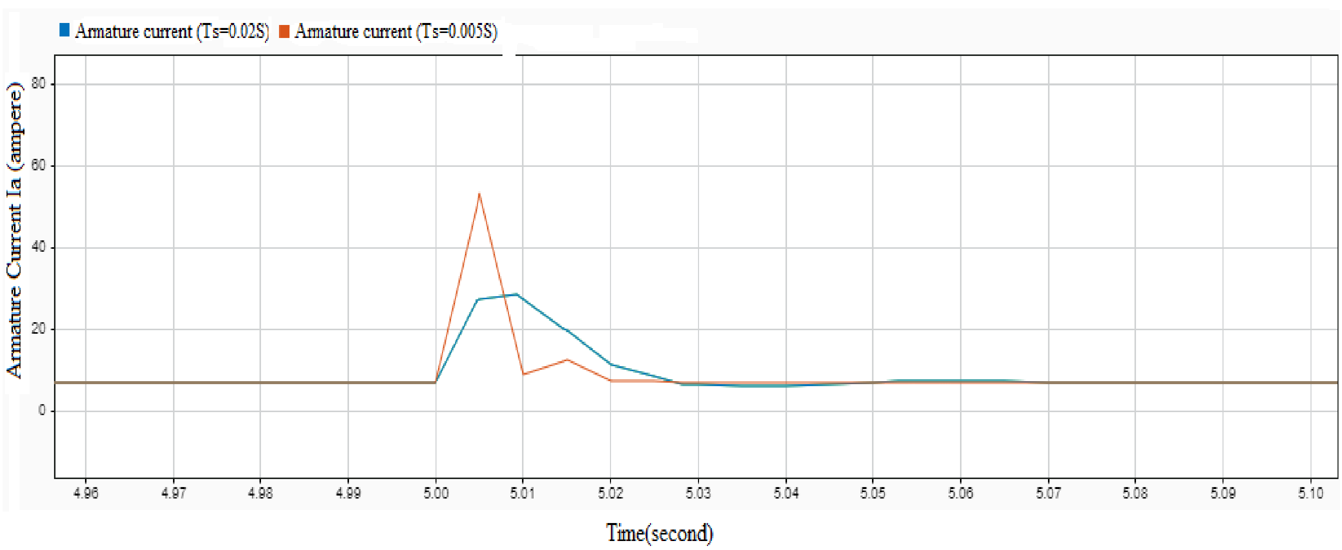

Fig 4.37: Intermediate armature current changing comparison at 0.005s and 0.02s sampling time.

4.7.4 Armature current comparison of MPC for proposed DC motor model at sampling time 0.01s and 0.02s

Fig. 4.38 shows the input armature current comparison of MPC for the proposed DC motor model at 0.01s and 0.02s sampling time. The reference speed was initially 800 RPM but changed to 1200 RPM at the interval of 5s.

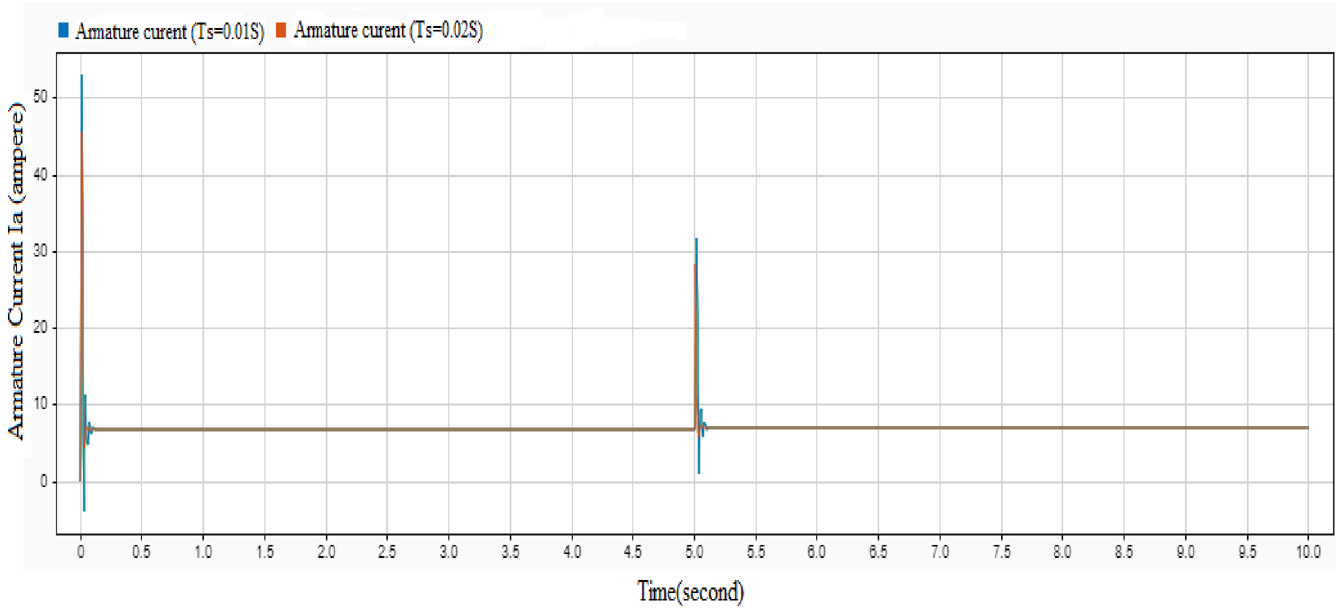

Fig. 4.38: Input armature current comparison at 0.01s and 0.02s sampling time.

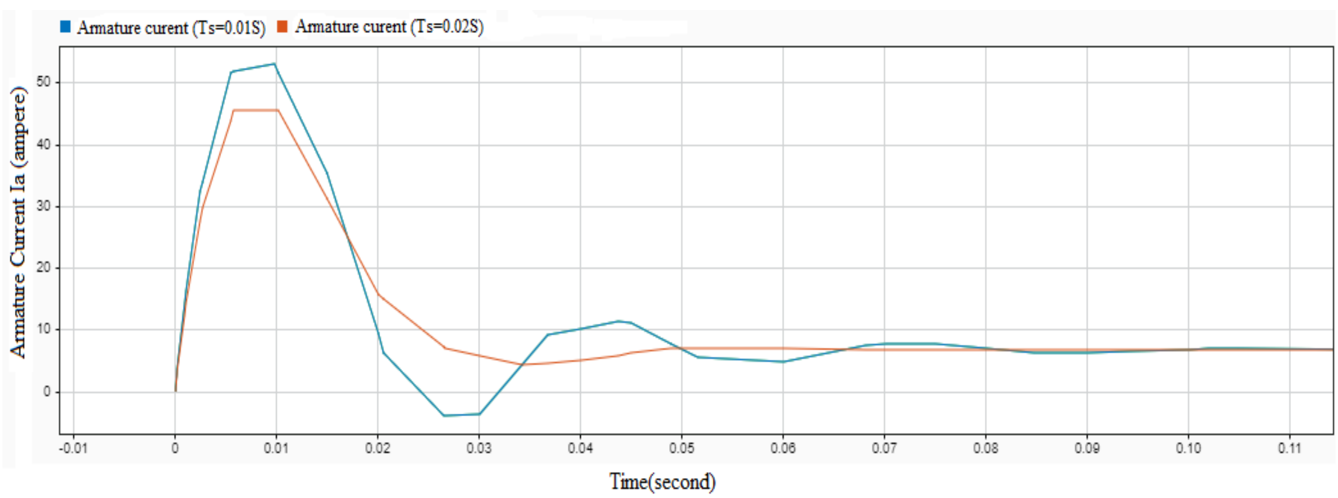

Fig 4.39: Initial armature current changing comparison at 0.01s and 0.02s sampling time.

Fig 4.39 shows initial armature current changing comparison of MPC for the proposed DC motor model at 0.01s and 0.02s sampling time. As sampling time doubled, the peak value of the armature current reduced. Fig 4.40 shows intermediate armature current changing comparison of MPC for the proposed DC motor model at 0.01s and 0.02s sampling time with interval of 5s.

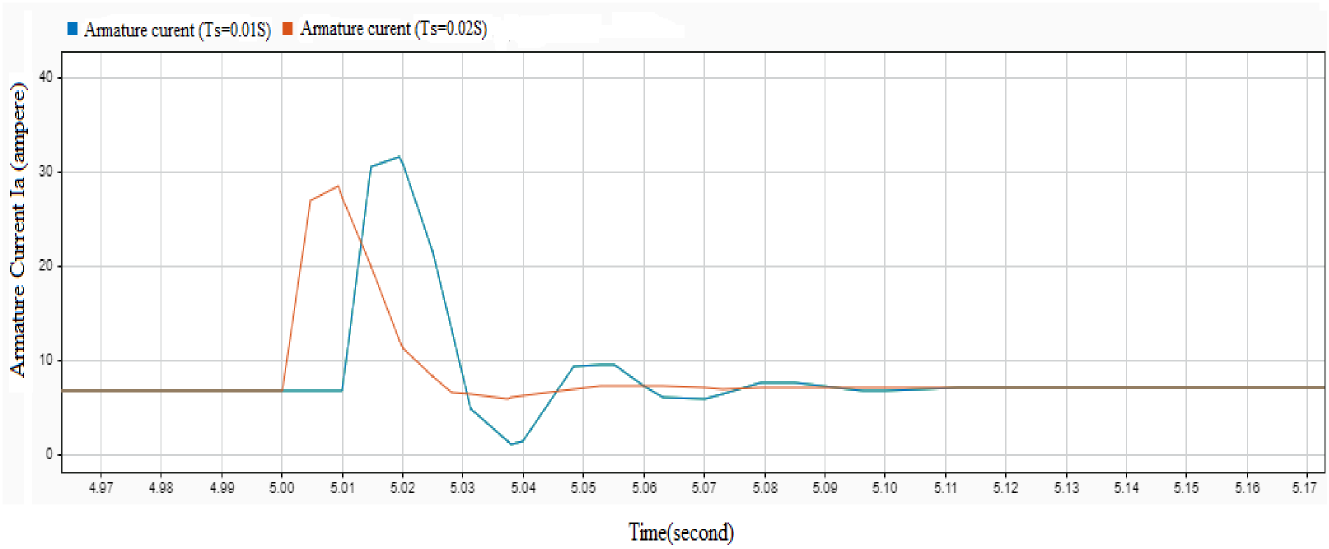

Fig 4.40: Intermediate armature current changing comparison at 0.01s and 0.02s sampling time.

4.7.5 Torque comparison of MPC for proposed DC motor model at sampling time 0.005s and 0.02s

Fig. 4.41 shows torque comparison of MPC for proposed DC motor model at 0.005s and 0.02s sampling time. The reference speed was initially 800 RPM but changed to 1200 RPM at the interval of 5s.

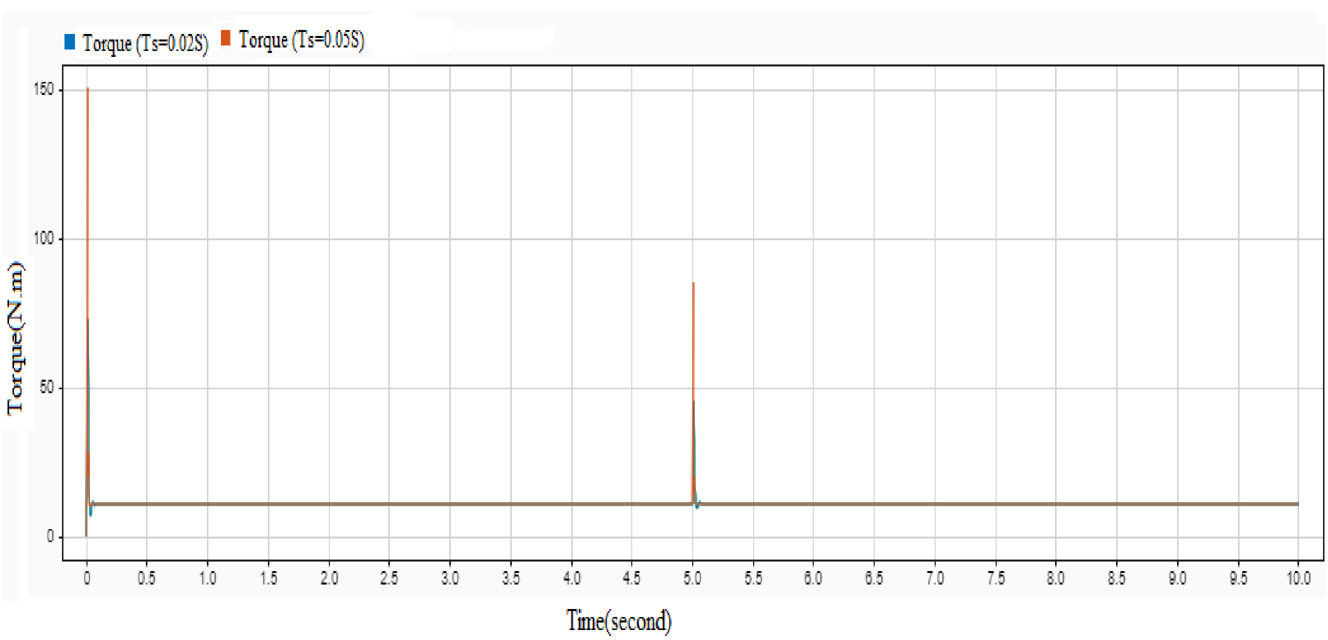

Fig 4.41: Output torque comparison at 0.005s and 0.02s sampling time.

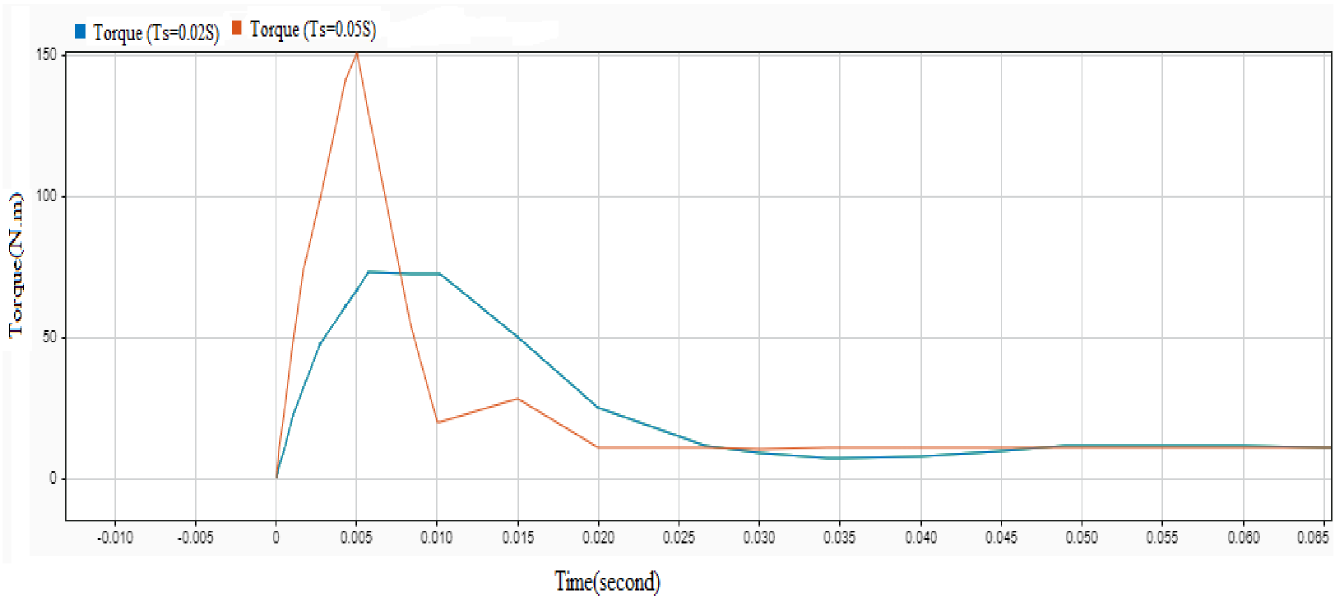

Fig 4.42: Initial torque comparison at 0.005s and 0.02s sampling time.

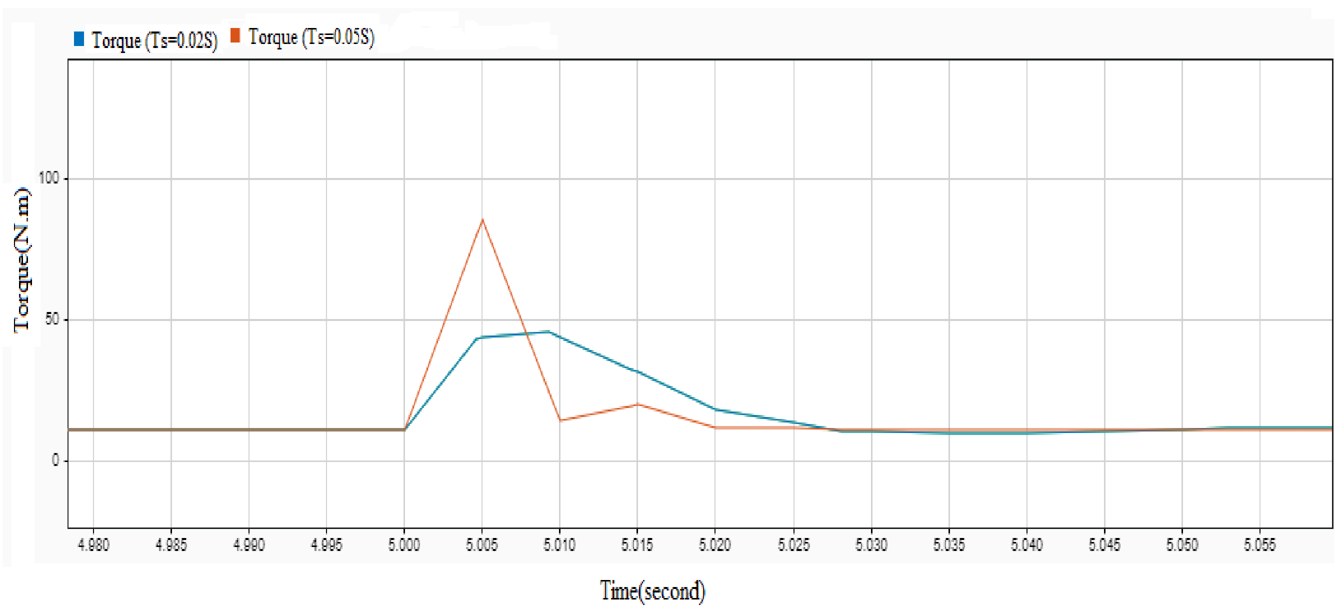

Fig 4.43: Intermediate torque changing comparison at 0.005s and 0.02s sampling time.

Fig 4.42 shows initial torque changing comparison of MPC for the proposed DC motor model at 0.005s and 0.02s sampling time. As sampling time doubled, the peak value of the torque reduced. Fig 4.43 shows intermediate torque changing comparison of MPC for proposed DC motor model at 0.005s and 0.02s sampling time with interval of 5s.

4.7.6 Torque changing Comparison at sampling time 0.01s and 0.02s

Fig. 4.44 shows torque comparison of MPC for the proposed DC motor model at 0.01s and 0.02s sampling time. Peak torque decreased with the sampling time increased.

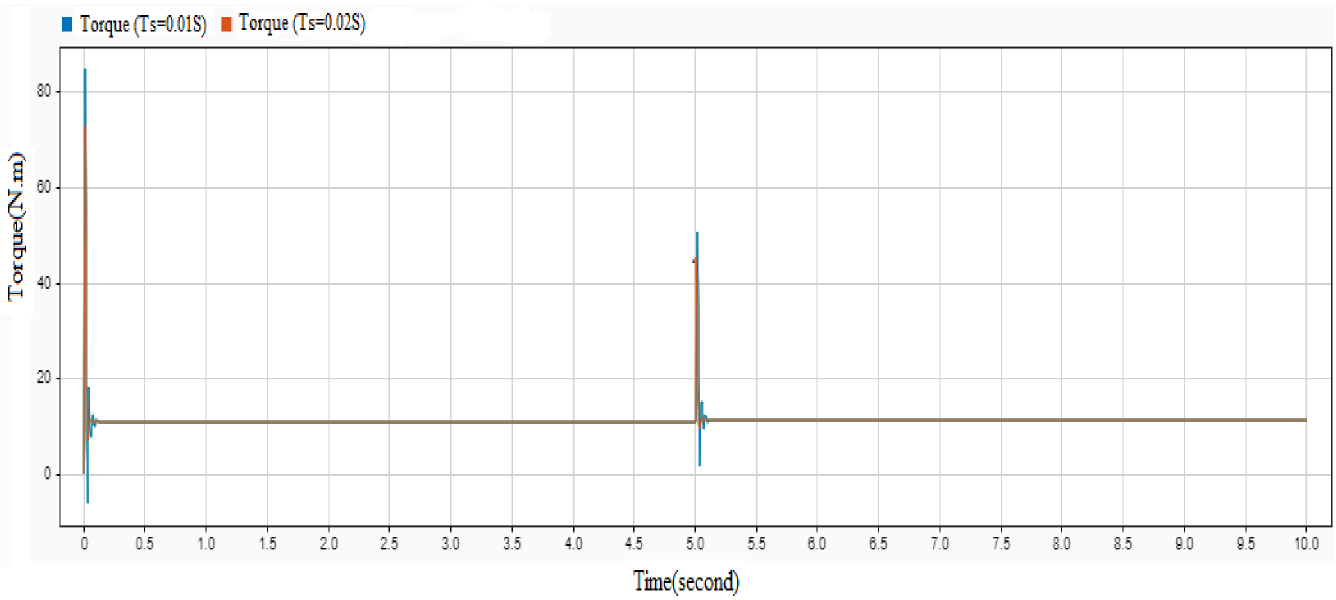

Fig 4.44: Output torque comparison of MPC for proposed DC motor model at 0.01s and 0.02s sampling time.

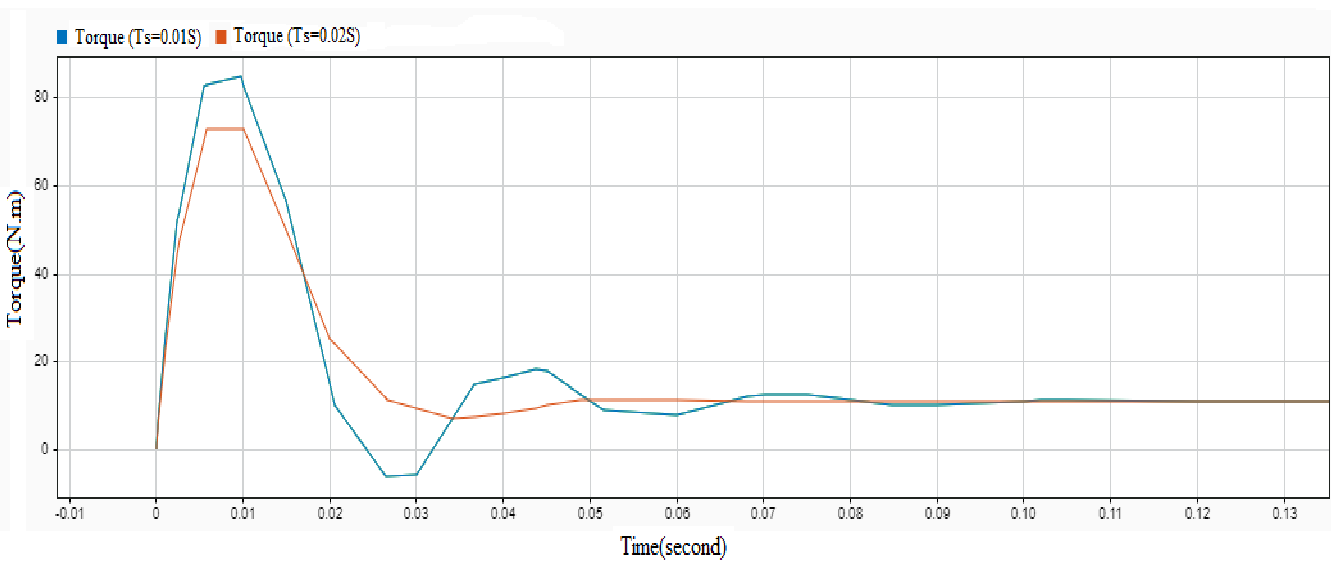

Fig 4.45: Initial torque comparison at 0.01s and 0.02s sampling time

Fig 4.45 shows initial torque changing comparison of MPC for proposed DC motor model at 0.01s and 0.02s sampling time. Peak torque decreases with the sampling time increases.

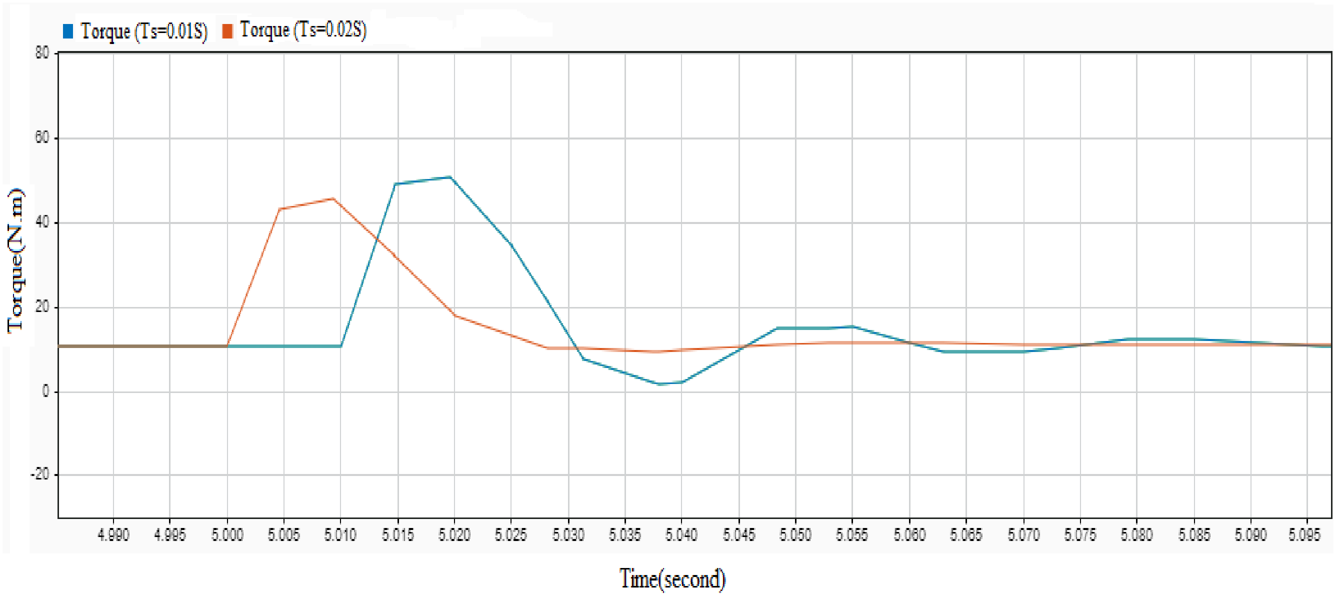

Fig 4.46: Intermediate torque changing comparison at 0.01s and 0.02s sampling time

Fig 4.46 shows the intermediate torque changing comparison of MPC for proposed DC motor model at 0.01s and 0.01s sampling time with interval of 5s. It is possible that the longer sampling time leads to a delay in detecting changes in the motor's behavior. This delay may cause the control actions to be less effective in regulating the motor's operation, resulting in a lower peak torque.

4.7.7 Overall Comparison of MPC for proposed DC motor model at different sampling time

Rise time, settling time, and overshoot are all important performance parameters for control systems engineering. Table 4.8 shows the comparison of MPC for proposed DC motor model at different sampling time. Sampling time reduction generates faster response and gives faster settling time, rise time of the system.

Peak armature current and torque comparison are also shown in the table. As sampling time increased, the peak value of the armature current decreased.

Peak torque decreased with the sampling time increased as torque is proportional to the armature current.

Table 4.8: Comparison of MPC for the proposed DC motor model at different sampling time.

Comparison Horizon (Sampling time)	MPC 0.005s	MPC 0.01s	MPC 0.02s
Rise time (Second)	0.042	0.057	0.064
Settling time(second)	0.055	0.081	0.093
Peak Armature current (A)	92.7	53.8	46.5
Peak Torque(N.m)	151	88.5	77

The choice of sampling time in a control system can greatly impact on its performance. If the sampling time is too long, the system is possibly unable to react fast enough to change in the controlled process, leading to suboptimal performance.

Increasing the sampling time in MPC affects the system's performance by reducing the control accuracy and slow the response time.

Conversely, if the sampling time is too short, the system may become computationally overloaded, potentially leading to instability or slow execution. Therefore, choosing an appropriate sampling time is critical to achieving optimal control performance.

4.8 Speed limit control by Adding Constraint

MPC controller for DC motor speed control with output constraints can be a powerful tool for achieving precise control of the motor speed while ensuring that the motor does not operate outside of the specified limits. MPC can be used as a control approach to optimize control operations within a limited time frame. When controlling the speed, an MPC controller is used to modify the motor's input voltage in order to meet output requirements such as restricting the maximum speed.

4.8.1 Adding Constraint to output speed

Table 4.9 shows the output speed limit of MPC for the proposed DC motor model. Without any constraint the output minimum and maximum limit is infinity. If the input reference speed 300 RPM is applied, output speed will be also 300 RPM without any constraint. But if the output minimum speed limit is constrained at 600 RPM then input 300 RPM will generate output speed of 600 RPM as minimum output is 600 RPM.

Again, without any constraint if the input speed 1200 RPM is applied to the motor model, output speed will be also 1200 RPM. But, if the output maximum speed limit is constrained at 1000 RPM then input 1200 RPM will generate output speed of 1000 RPM as maximum output speed is 1000 RPM.

Table 4.9: Output speed limit of MPC for the proposed DC motor model

Logic	Input speed (RPM)	Output min speed (RPM)	Output max speed (RPM)	Output Speed (RPM)
Without constraint	300	0	5000	300
With constraint	300	600	1000	600
Without constraint	1200	0	5000	1200
With constraint	1200	600	1000	1000

4.8.2 Output speed constraint simulation result

One of the challenges in DC motor speed control is ensuring that the motor operates within certain speed constraints. Fig 4.47 shows the output speed comparison of MPC for the proposed DC motor model with constraint and without constraint. Here, minimum output speed is 600 RPM and maximum output speed is 1000 RPM. If the reference input speed 300 RPM for the first 5s and 1200 RPM for the rest time are applied, without constraint the output speed will be same as input. But, with output speed constraints the output speed will be initially 600 RPM and after 5s it will be 1000 RPM.

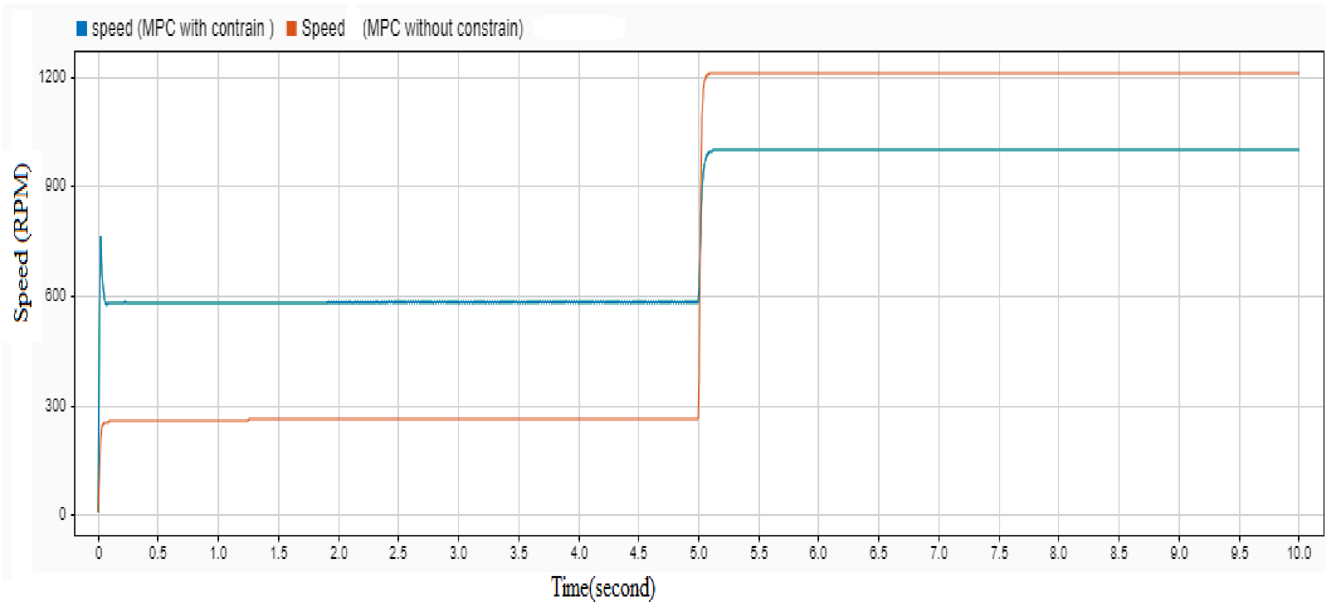

Fig 4.47: Output speed of MPC for proposed DC motor model with constraints and without constraints.

4.8.3 Applications of MPC for DC motor with Speed limit

1. **Electric vehicles**: In electric vehicles, the MPC DC motor controller with speed limit can be used to regulate the vehicle speed within a certain range. The controller can take into account various factors such as road conditions, traffic density, and vehicle dynamics, and calculate the optimal speed that ensures the vehicle stays within the speed limit while reaching the destination in the shortest time possible.
2. **Industrial DC motors:** In industrial machinery, the MPC for DC motor controller with speed limit can be used to regulate the speed of machines such as conveyor belts, cranes, and robots.
3. **Wind turbines**: In wind turbines, the MPC for DC motor controller with speed limit can be used to regulate the speed of the rotor blades. The controller can take into account various factors such as wind speed, turbine load, and power output, and calculate the optimal blade speed that ensures the turbine operates within the speed limit while maximizing power generation.
4. **Elevators**: In elevators, DC cumulative compound motor is used. The MPC for DC motor controller with speed limit can be used to regulate the speed of the elevator. The controller can take into account various factors such as the weight of the passengers, the floor distance, and the elevator's mechanical properties, and calculate the optimal speed that ensures the elevator operates within the speed limit while providing smooth and safe rides.

Chapter 5
Hardware Implementation

5.1 Hardware testing

Different approaches are available now for creating digital systems. These approaches cover system validation, verification, and tuning techniques. Two methods are primarily used while testing new systems [39]. Using software simulation tools, the system's performance and functionality are monitored and optimized. The system is implemented on the desired platform to check for effectiveness and gauge performance under accurate circumstances. The term "hardware testing" is another name for this strategy.

5.1.1 Hardware in the Loop Simulation

Complex process systems are developed and tested using a technique called hardware in the loop testing. By complicating test platform with the controlled plant, HIL simulation offers a powerful platform. An illustration of the related dynamic systems in mathematics is added to test and development for the complexity of the controlled plant [40-41]. Time to market and complexity are the two main drivers that are driving the implementation of a HIL process across all industries.

The following are the primary steps in HIL Simulation:

1. Creating a mathematical model.
2. HIL simulation (Software + Hardware)
3. Real time hardware implementation (Hardware Only)

5.1.2 Experimental picture of the proposed HIL platform

In this experiment, a 12 volt DC gear motor is used. The motor gear ratio is 4:1. If motor speed is 800 RPM the output shaft speed will be 200 RPM. DC motor speed is controlled by PWM signal via IBT2 motor driver.

The use of the speed sensor is to measure the motor output speed and send the data to the Arduino. It's a process of a feedback loop. Speed sensor obtains data and transfer it to Arduino. Following speed computation, Arduino delivers PWM signals to motor driver to operate the motor at a reference speed.

Since Arduino cannot perform high speed & complex processing, Raspberry Pi may be utilized for the high speed MPC computational process.

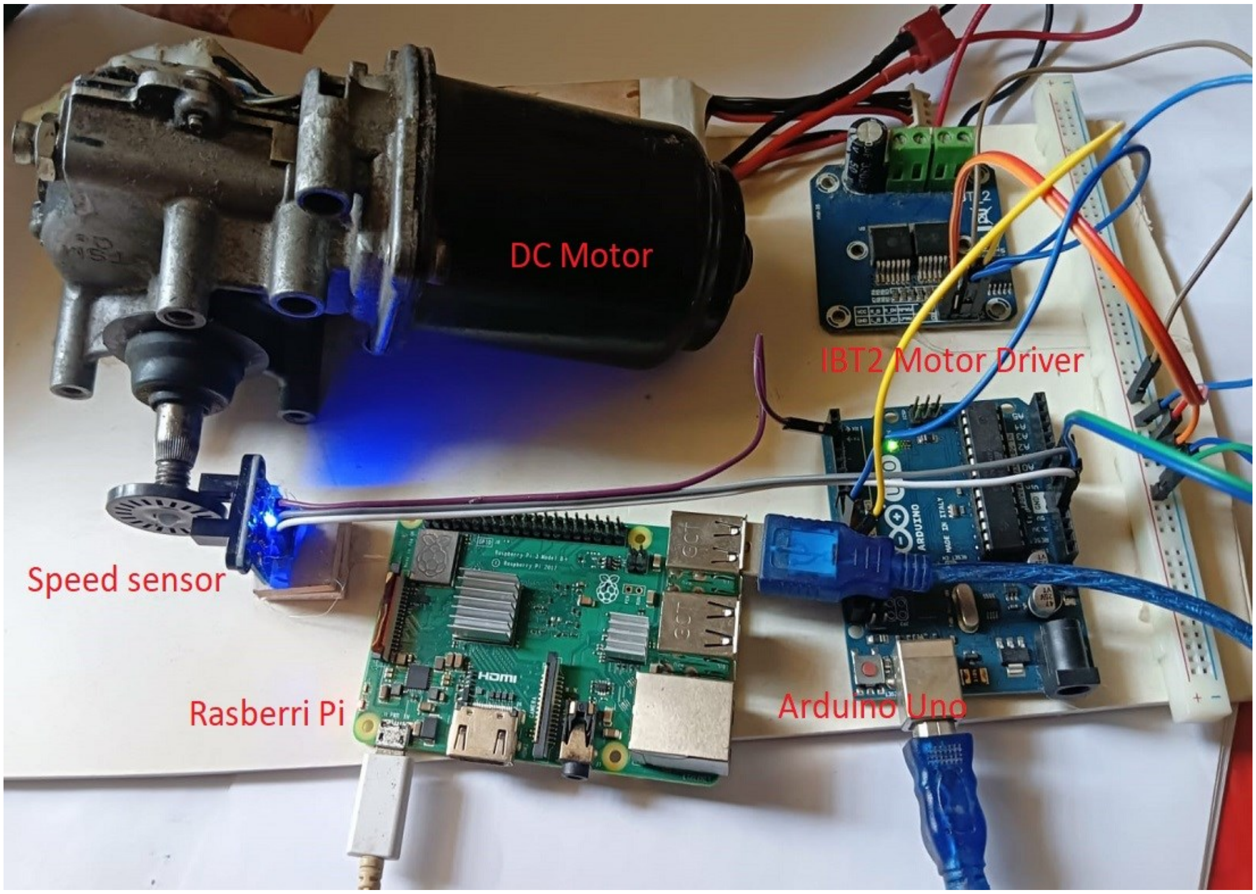

Fig 5.1: Experimental picture of the hardware connections.

5.2 Hardware Experimental Results (Without any control method)

Fig. 5.2 shows the output speed of the DC motor. The reference speed was initially 800 RPM but changed to 1200 RPM at the interval of 5s. The motor gear ratio is 4:1. Gear motors have a built-in gear reduction gearbox connected to a shaft. So the output shaft speed will be reduced by 4 times the original motor speed.

Experimentally the shaft speed was initially 200 RPM but changed to 300 RPM at the interval of 5s. Gearbox reduces the shaft speed but increases the output torque.

The process was done without any feedback control. The motor driver operates the motor in accordance with the PWM signals that Arduino delivered to the motor controller.

Output speed is fluctuating as there is noise and disturbances around the environment. The steady state error was large enough to make the model unstable.

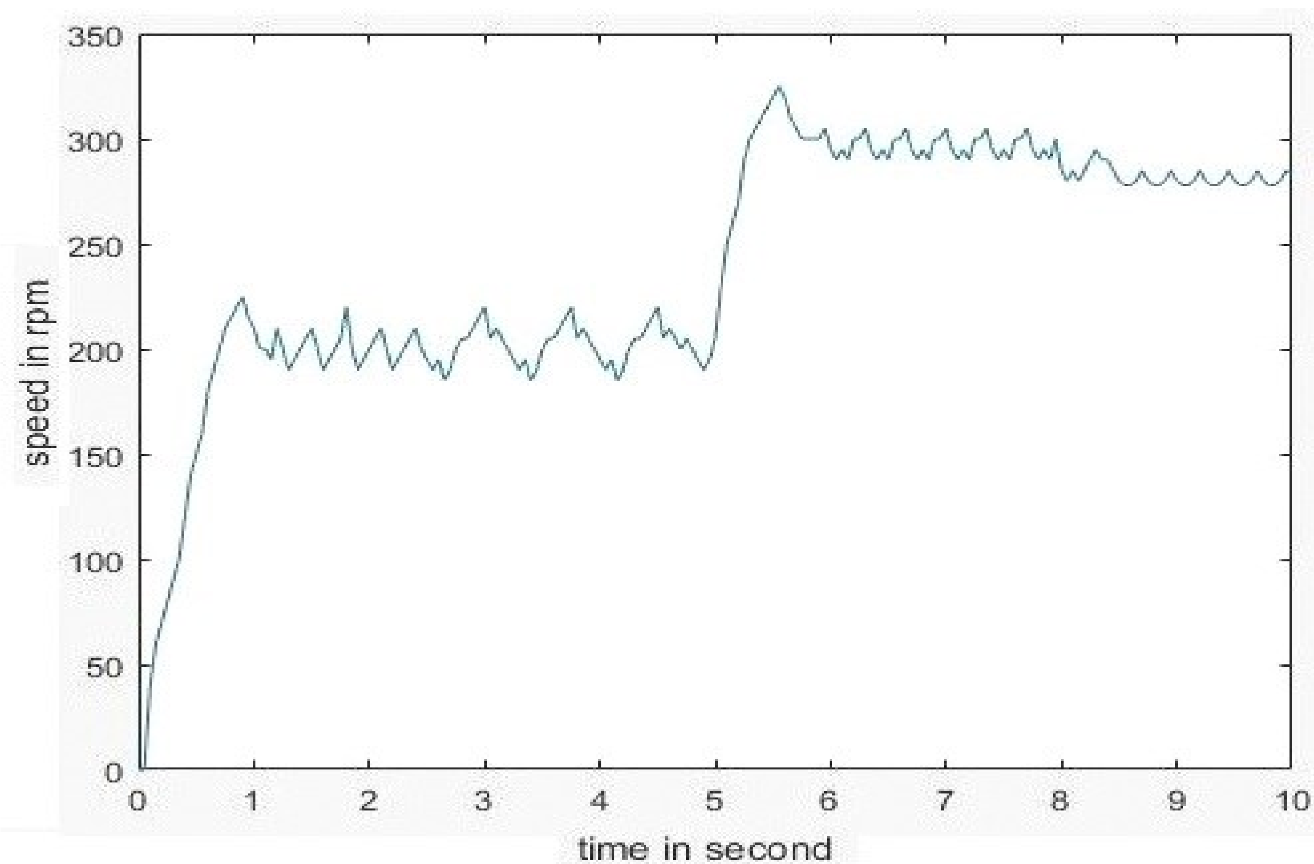

Fig 5.2: Hardware experimental result of DC motor speed control.

5.3 Hardware testing Based on PID Controller

Fig. 5.3 shows the hardware experimental result of PID based DC motor speed control. The reference speed was initially 800 RPM but changed to 1200 RPM at the interval of 5s. Gear motor have shaft output with built-in gear reduction gearboxes. The motor gear ratio is 4:1. So the output shaft speed will be reduced by 4 times the original motor speed. Experimentally the shaft speed was initially 200 RPM but changed to 300 RPM at the interval of 5s.

The process was done with feedback control. Arduino sensed the speed of the shaft connected to the motor via speed sensor and provided PWM signals for the motor driver.

The model output is PID controlled. The output shaft speed fluctuated due to noise and external disturbances.

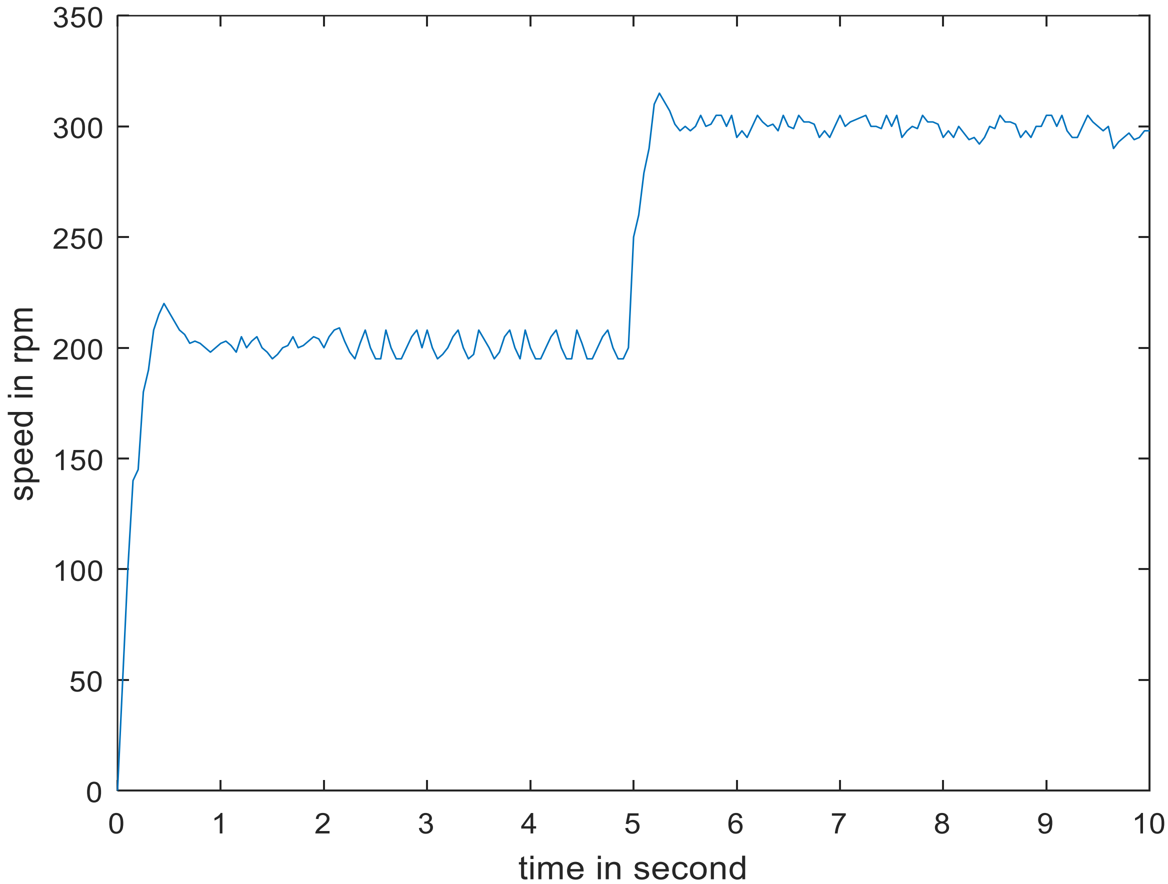

Fig 5.3: Hardware experimental result of PID based DC motor speed control.

5.4 Hardware testing Based on MPC controller

MPC methods are classified into two types: online and offline. The choice between online and offline MPC depends on the specific application and the requirement of the control system. If real-time optimization and fast feedback are necessary, online MPC may be the best choice. However, if the system can tolerate some delay in the control inputs and the computational resources are limited, offline MPC may be a better option. In general, online MPC is more flexible and can handle more complex systems, while offline MPC is more computationally efficient and can handle larger-scale systems.

5.4.1 Offline MPC

Offline MPC involves solving the optimization problem offline to provide a set of control policies for many probable states using a mathematical model of the system and the constraints. Based on measurements made while the system is in operation and the current state as determined by those measurements, the appropriate control policy is selected from the precomputed set of policies. Offline MPC is less computationally intensive than online MPC and can be used in systems where real-time optimization is not necessary [39].

Fig 5.4 shows the hardware experimental result of offline MPC based DC motor speed control. At time the interval 5s the reference speed was increased to 1200 RPM from 800 RPM. The motor gear ratio is 4:1. So the output shaft speed will be reduced by 4 times the original motor speed. So, the shaft speed was initially 200 RPM but changed to 300 RPM at the interval of 5s. The output shaft speed seems to be stable but there are fluctuations due to noise and external disturbances.

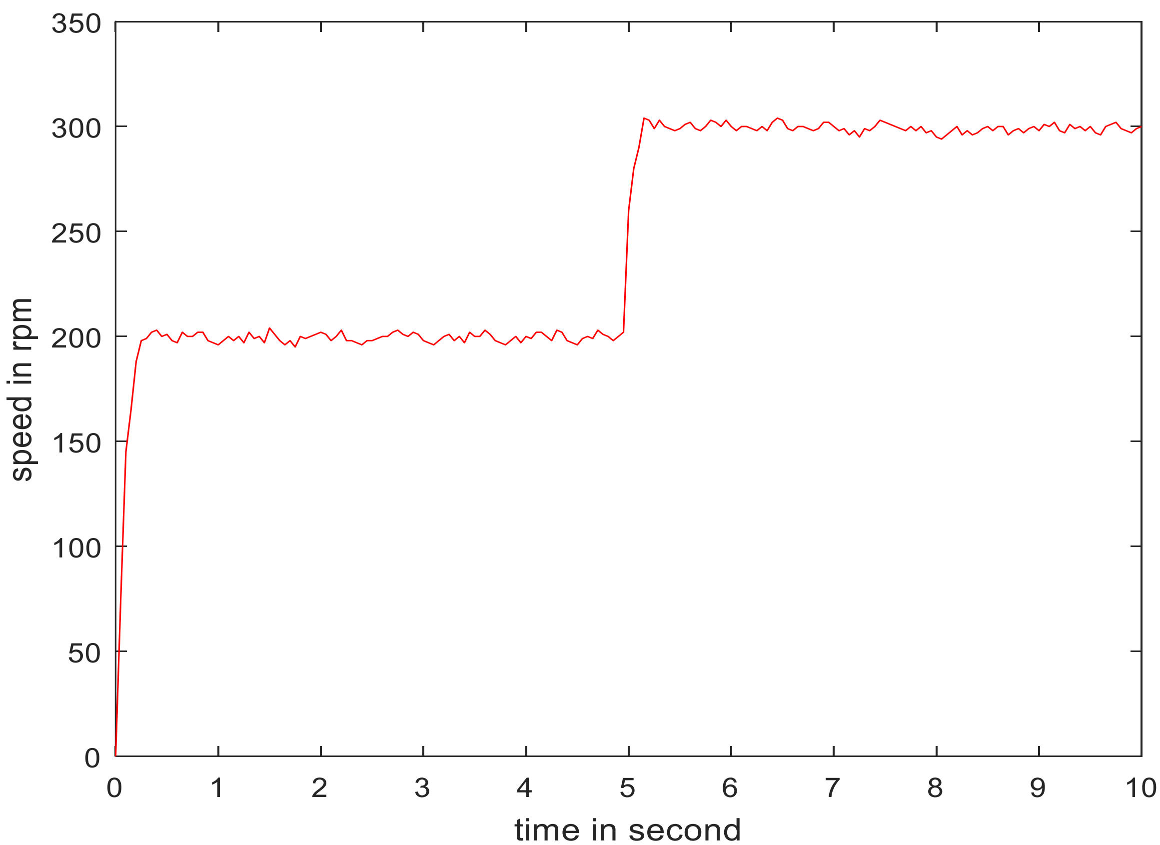

Fig 5.4: Hardware experimental result of offline MPC based DC motor speed control.

5.4.2 Online MPC

Online MPC involves continuously updating control inputs caused by real-time measurements of the system state. The model predictions are updated based on the latest measurements, and the optimization is resolved immediately to generate optimal control inputs for the next time step [42]. Online MPC is computationally intensive and requires a fast and reliable feedback loop, which can be challenging to implement in real-time systems.

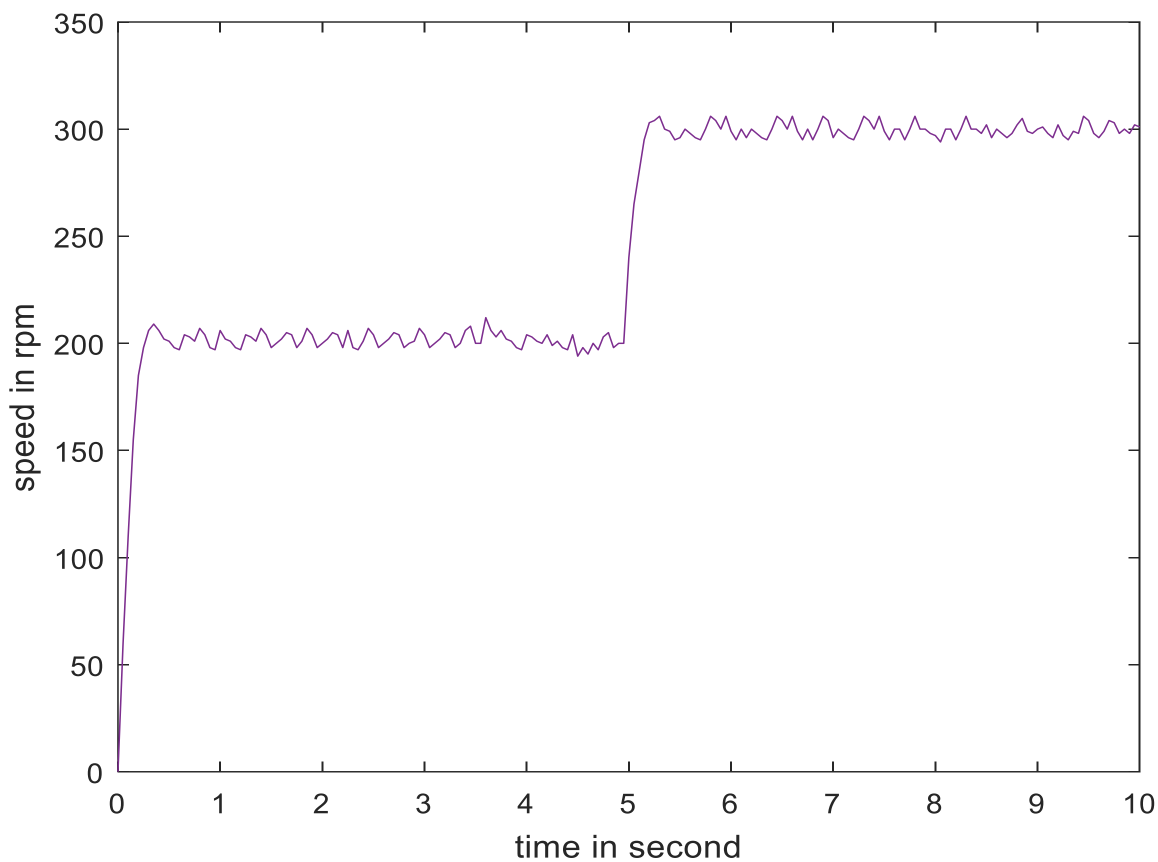

Fig 5.5: Hardware experimental result of online MPC based DC motor speed control

Fig 5.5 shows the hardware experimental result of online MPC based DC motor speed control. With time interval of 5s the reference shaft speed was increased to 300 RPM from 200 RPM. Here, rise time and settling time are less than PID control DC motor.

5.5 Offline MPC with Speed Constraints

Fig 5.6 shows the hardware experimental result of offline MPC based DC motor speed control with output speed limit. At time the interval of 5s the reference speed was increased to 1200 RPM from 800 RPM. The motor gear ratio is 4:1. So the output shaft speed should be initially 200 RPM but changed to 300 RPM at the interval of 5s.

But if output constraints were added as minimum speed limit to 800 RPM and the maximum speed at 1000 RPM, then the output speed would not be able to follow the reference. So, practically the output shaft speed will be constrained at the range of 200 RPM to 250 RPM (reduced by 4 times via gear transmission).

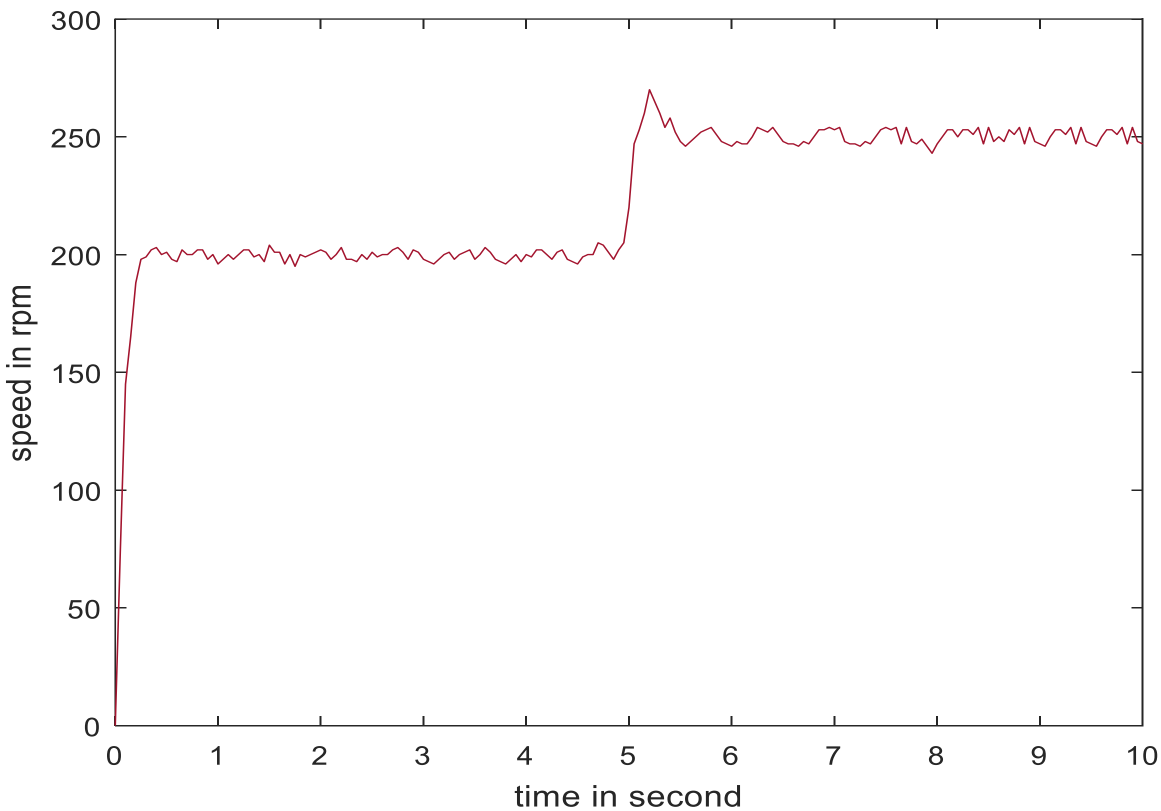

Fig 5.6: Hardware experimental result of offline MPC based DC motor speed control with constraints.

But, practically the shaft speed was higher than 200 RPM in speed changing situations.This may be resulted from the sensor data acquisition and MPC hardware processing delay.

Chapter 6
Conclusion & Future Work

6.1 Conclusion of this work

The MPC and PID control strategies were developed and successfully implemented on the DC motor model. The controller's performances were evaluated for situations including reference tracking, limitations handling that changed over time. According to simulation studies, MPC outperformed PID in terms of performance since it could handle constraints by default, required less effort to keep track of the reference, and provided smooth output without oscillations. Table 6.1 shows the comparison between PID and MPC (Practical vs Simulation data) for DC motor speed output. The rise time, settling time, and overshoot percentage have been minimized with the use of MPC for DC motor.

If the sampling period is too small, the computational load may be too high, leading to slow execution or instability. Alternatively, if the sample period is too lengthy, the control inputs may not be updated quickly enough to maintain pace with the system changes, resulting in poor performance.

Table 6.1: Simulation vs Experimental data of PID and MPC

Comparison Horizon	PID Simulation	PID Experimental	MPC Simulation	MPC (offline) Experimental	MPC(online) Experimental
Rise time (Second)	0.046	0.39	0.042	0.33	0.35
Settling time(second)	0.127	0.67	0.055	0.51	0.58
% of OS	5.22	7.2	NA	2.3	4.3
Peak(RPM)	315	321	299	307	313

The MPC for proposed DC motor model showed better experimental performance when compared with conventional PID controllers. However, majority of the analysis were done based on closed loop topology.

6.2 Recommendation for future work

A filter that reduces noise might be used in future studies to get a better response. Optimal observers, such as the Kalman filter, can be used to predict the value of the states in order to further improve the efficiency of the MPC controller.

From the simulation, it has been observed that by decreasing the sampling time, MPC has given better results.Digital signal processors with greater clock speed would generate more accurate and fast control signalsto run the motor as sampling time can be minimized on it.

The proposed approach may be extended to handle sensor failures and actuator saturation, and the system will still be able to maintain stable control.

References:

[01] M. Morari and J. H. Lee, "MPC: Past, present, and future," Computers and Chemical Engineering, vol. 23, no. 4–5, pp. 667–682, 1999.

[02] Stefan Kozak, "State –of-the-art in control engineering", Journal of Electrical System and Information Technology 1, pp.1-9, 2014.

[03] Silverio Bolognani, Luca Peretti and Mauro Zigliotto, "Design and Implementation of MPC for Electrical Motor Drives." IEEE Transactions on Industrial Electronics, pp.7-9 June 2009.

[04] Stefan Kozak, "From PID to MPC: Control engineering methods development and applications", Cybernetics & Informatics (K&I), Levoca, pp. 1-7, 2016.

[05] S. K. Kim, D. K. Choi, K. B. Lee, and Y. I. Lee, "Off set free MPC for the power control of three phase ac/DC converters," IEEE Transactions on Industrial Electronics, vol. 62, no. 11, pp. 7114–7126, Nov. 2015.

[06] S. Saeidi, and R. Kennel, "A novel algorithm for MPC of ac electric drives," 2nd International Electric Drives Production Conference (EDPC), Nuremberg, Germany, Oct. 15–18, 2012.

[07] M. Preindl, and S. Bolognani, "Comparison of direct and PWM MPC for power electronic and drive systems," IEEE Applied Power Electronics Conference and Exposition (APEC), Long Beach, CA, USA, Mar. 17–21, 2013.

[08] F. Wang et al., "Model-based predictive direct control strategies for electrical drives: An experimental evaluation of PTC and PCC methods," IEEE Transaction on Industrial Informatics, vol. 11, no. 3, pp. 671–681, Jun. 2015.

[09] Abdelsalam A. Ahmed, Byung Kwon Koh & Young Il Lee "Continuous Control Set-MPC for Torque Control of Induction Motors in a Wide Speed Range", Journal on Electrical power system, Volume 46, pp. 66-67, 2018.

[10] Nocedal J, Wright S., "Numerical optimization," Springer Science & Business Media, pp. 95-99 ,2006.

[11] Bimal K. Bose,“Power Electronics and Variable Frequency Drives: Technology and Applications” Wiley-IEEE Press, pp. 56-59,1997.

[12] M. A. Rahman, and M. A. Hoque, “Sensor less direct torque control of induction motors used in electric vehicle‖”, IEEE Trans. On Energy Conversion, Vol. 18, No. 1, pp. 1-9, March 2003.

[13] Dr. Settar S. Keream , Khalid G. Mohammed, Mayyada Sahib Ibrahim, “Analysis Study In Principles Of Operation Of DC Machine” Journal of Advance Research in Dynamical & Control Systems, Vol. 10, 02-Special Issue, pp.44-49 , 2018

[14] P. Dhinakaran, D. Manamalli, “Novel strategies in the Model-based Optimization and Control of Permanent Magnet DC motors”, Computers & Electrical Engineering, Volume 44, pp. 34-41, 2015.

[15] A. G. M. A. Aziz, A. A. Z. Diab, and M. A. E. Sattar, "Characteristics of DC motor, speed vs torque", Nineteenth International Middle East Power Systems Conference (MEPCON), 2017, pp. 1059-1068.

[16] V. Vishal, V. Kumar, K. P. S. Rana and P. Mishra, “Comparative study of some optimization techniques applied to DC motor control,” Advance Computing Conference (IACC), 2014 IEEE International, Gurgaon, pp. 1342-1347, 2014

[17] Ramu Krishnan, “Electric motor drives: modeling, analysis, and control”, Prentice Hall, pp.68-71, 2001.

[18] Messay Tadese et al. “Performance Investigation of DC Motor Angular Velocity using Optimal and Robust Control Method” International Journal of Advance Research and Innovative Ideas in Education, Vol. 6, Issue. 2, pp. 1016- 1022, 2020

[19] Linder, Arne, Rahul Kanchan, Peter Stolze, and Ralph Kennel. "Model-based predictive control of electric drives." Cuvillier, 2010.

[20] Mariethoz, Sebastien, Alexander Domahidi, and Manfred Morari. "High-bandwidth explicit MPC of electrical drives." IEEE Transactions on Industry Applications, vol. 9, pp. 87-89,2012

[21] Ahmed A. Zaki Diab, Denis A. Kotin and Vladimir V. Pankratov. "Speed control of sensorless induction motor drive based on MPC." In 2013 14th International Conference of Young Specialists on Micro/Nanotechnologies and Electron Devices, pp. 269-274. IEEE, 2013.

[22] Aakriti Khanna, Tamanna Gaur ,"MPC Of DC motor model In Matlab" International Journal of Scientific & Engineering Research, Volume 8, pp. 56-59, Issue 4, April-2017.

[23] Mohamed I.S., Rovetta S., Do T.D., Dragicevic T., Diab A.A.Z.," A neural-network-based MPC of three-phase inverter with an output LC Filter", IEEE Access 7 , art. No. 8819887, pp. 124737-124749, 2019

[24] J. Rodriguez and P. Cortes, "Predictive Control of Power Converters and Electrical Drives.", John Wiley & Sons, 2012

[25] A. G. M. A. Aziz, H. Rez, and A. A. Z. Diab, "Robust Sensorless Model-Predictive Torque Flux Control for HighPerformance Induction Motor Drives," Mathematics, vol. 9, pp. 4-7, 2021.

[26] Mohamed F. Abu-Elyazeed, Sherif A. Zaid, Assistant Dr. Hany M. Elsayed, "Implementation of MPC for Three Phase Inverter with Output LC Filter Using DSP" IEEE transaction on power electronics, vol. 20, pp. 5-9 ,June 2014.

[27] Ahmed Rubaai , Marcel J. Castro and Abdul R. Ofoli "DSP-Based Laboratory Implementation of Hybrid Fuzzy-PID Controller Using Genetic Optimization for High-Performance Motor Drives" IEEE transaction on industry applications, vol. 44, no. 6, pp. 64-68, December 2008.

[28] V. Vishal, V. Kumar, K. P. S. Rana and P. Mishra, "Comparative study of some optimization techniques applied to DC motor control," Advance Computing Conference (IACC), 2014 IEEE International, Gurgaon, pp. 1342-1347, 2014

[29] Ramu Krishnan, “Electric motor drives: modeling, analysis, and control”, Prentice Hall, pp.444-450, 2001.

[30] Siddhesh Dani, Dayaram Sonawane, Deepak Ingole and Sanjaykumar Patil, “Performance Evaluation of PID1, LQR and MPC for DC Motor speed Control” 2017 2nd International Conference for Convergence in Technology (I2CT), 2017.

[31] P. Dhinakaran, D. Manamalli, “Novel strategies in the Model-based Optimization and Control of Permanent Magnet DC motors”, Computers & Electrical Engineering, Volume 44, pp. 34-41, 2019.

[32] J.Nilsson. “Real-Time Control Systems with Delays”, PhD thesis, Department of Automatic Control, Lund Institute of Technology, pp. 45-55, 1998.

[33] Yu-Geng Xi, De-Wei Li, Shu Lin, “MPC–status and challenge,” Acta Automatica Sinica, 39(3), pp. 222-236, 2013

[34] A. Syaichu-Rohman and R. Sirius, “MPC implementation on a programmable logic controller for DC motor speed control,” Electrical Engineering and Informatics (ICEEI), International Conference on, Bandung, pp. 1-4, 2011

[35] Ahmed A. Zaki Diab, D. A. Kotin, V. N. Anosov, and V. V. Pankratov. "A comparative study of speed control based on MPC and PI-controller for Indirect Field oriented control of induction motor drive." In 2014 12th International Conference on Actual Problems of Electronics Instrument Engineering (APEIE), pp. 728-732. IEEE, 2014.

[36] Ahmed G. Mahmoud A. Aziz , Yehia Sayed Mohammed , Hamdi Ali, Ahmed A. Zaki Dia, “Investigation of the Performance of MPC for Induction Motor Drives” International Journal of Engineering and Information Systems (IJEAIS) ISSN Vol. 5 Issue 2, pp. 107-116, February – 2021.

[37] Mujtaba Jaffery , Sohaib Aslam , Moinuddin Ghauri , M. Shahzad Khuram , Sikandar Rafiq , Subhan Khan, “Real-time implementation of model predictive control on a 16-bit microcontroller for speed control of a DC motor”, Journal of Engineering Technology Volume 6, Issue 1,pp. 415-434, Jan. 2018.

[38] Paul I -Hai Lin, Santai Hwang and John Chou "Comparison on Fuzzy Logic and PID controls for a DC Motor Position Controller", National Taipei Institute of Technology, Taipei, Taiwan,Republik of china, 1994 IEEE.

[39] A.K. Abes, F. Bouani, M. Kasouri, "A microcontroller Implementation of constrained MPC" World Academy of Science, Engineering and Technology, vol 5, pp. 662-665, 2011

[40] L. Wang. MPC system design and implementation using MATLAB. Springer, pp. 84-90, 2009

[41] Malouchee, I. Kherji, F. Bouani, "Automatic MPC Implementation in a Higher Performance Microcontroller", In Proceedings of 12th International Conference on Systems, Signals and Devices, Tunisia, pp. 1-6, 2015

[42] N.N.P.Mahalik and K.K.Kim, "A prototype for hardware in the loop simulation of a distributed control architecture," IEEE Transactions on Systems, vol.38, no.2, pp.189– 200, 2008.

APPENDICES

Codes for Arduino

```
#include "TimerOne.h"
unsigned int counter=0;
void docount()  // counts from the speed sensor
{
  counter++;  // increase +1 the counter value
}

void timerIsr()
{
  Timer1.detachInterrupt();  //stop the timer
  Serial.print("Motor Speed: ");
  int rotation = (counter / 10)*60;  // divide by number of holes in Disc
  Serial.print(rotation,DEC);
  Serial.println(" Rotation per mins");
  counter=0;  //  reset counter to zero
  Timer1.attachInterrupt( timerIsr );  //enable the timer
}
void setup()
{
  Serial.begin(9600);
  pinMode(Left_p,OUTPUT);
  pinMode(Left_n,OUTPUT)
  pinMode(ENA,OUTPUT);
  Timer1.initialize(1000000); // set timer for 1sec
  attachInterrupt(0, docount, RISING);  // increase counter when speed sensor pin goes High
  Timer1.attachInterrupt( timerIsr ); // enable the timer
}
```

Fig C.1: Arduino code for motor shaft speed measurement

```
#define ENCA 2
#define ENCB 3
#define PWM 10
#define IN1 8
#define IN2 9

// globals
long prevT = 0;
int posPrev = 0;
// Use the "volatile" directive for variables
// used in an interrupt
volatile int pos_i = 0;volatile float velocity_i = 0; volatile long prevT_i = 0;

float v1Filt = 0;float v1Prev = 0;float v2Filt = 0;float v2Prev = 0; float eintegral = 0;

void setup() {
  Serial.begin(9600);  pinMode(ENCA,INPUT);
  pinMode(ENCB,INPUT); pinMode(PWM,OUTPUT);
  pinMode(IN1,OUTPUT); pinMode(IN2,OUTPUT);

  attachInterrupt(digitalPinToInterrupt(ENCA),
                  readEncoder,RISING);
}
```

```
void loop() {

  // read the position and velocity
  int pos = 0;
  float velocity2 = 0;
  noInterrupts(); // disable interrupts temporarily while reading
  pos = pos_i;
  velocity2 = velocity_i;
  interrupts(); // turn interrupts back on

  // Compute velocity with method 1
  long currT = micros();
  float deltaT = ((float) (currT-prevT))/1.0e6;
  float velocity1 = (pos - posPrev)/deltaT;
  posPrev = pos;
  prevT = currT;

  // Convert count/s to RPM
  float v1 = velocity1/450.0*60.0;
  float v2 = velocity2/450.0*60.0;

  // Low-pass filter (25 Hz cutoff)
  v1Filt = 0.854*v1Filt + 0.0728*v1 + 0.0728*v1Prev;
  v1Prev = v1;
  v2Filt = 0.854*v2Filt + 0.0728*v2 + 0.0728*v2Prev;
  v2Prev = v2;


  // Convert count/s to RPM
  float v1 = velocity1/450.0*60.0;
  float v2 = velocity2/450.0*60.0;

  // Low-pass filter (25 Hz cutoff)
  v1Filt = 0.854*v1Filt + 0.0728*v1 + 0.0728*v1Prev;
  v1Prev = v1;
  v2Filt = 0.854*v2Filt + 0.0728*v2 + 0.0728*v2Prev;
  v2Prev = v2;

  // Set a target
  float vt = 50;

  // Compute the control signal u
  float kp = 2;
  float ki = 0.025;
  float e = vt-v1Filt;
  eintegral = eintegral + e*deltaT;

  float u = kp*e + ki*eintegral;

  // Set the motor speed and direction
  int dir = 1;
  if (u<0){
    dir = -1;
  }
```

```
  int pwr = (int) fabs(u);
  if(pwr > 255){
    pwr = 255;
  }
  setMotor(dir,pwr,PWM,IN1,IN2);

  Serial.print(vt);
  Serial.print(" ");
  Serial.print(v1Filt);
  Serial.println();
  delay(1);
}

void setMotor(int dir, int pwmVal, int pwm, int in1, int in2){
  analogWrite(pwm,pwmVal); // Motor speed
  if(dir == 1){
    // Turn one way
    digitalWrite(in1,HIGH);
    digitalWrite(in2,LOW);
  }
  else if(dir == -1){
    // Turn the other way
    digitalWrite(in1,LOW);
    digitalWrite(in2,HIGH);
  }

  else{
    // Or dont turn
    digitalWrite(in1,LOW);
    digitalWrite(in2,LOW);
  }
}

void readEncoder(){
  // Read encoder B when ENCA rises
  int b = digitalRead(ENCB);
  int increment = 0;
  if(b>0){
    // If B is high, increment forward
    increment = 1;
  }
  else{
    // Otherwise, increment backward
    increment = -1;
  }
  pos_i = pos_i + increment;

  // Compute velocity with method 2
  long currT = micros();
  float deltaT = ((float) (currT - prevT_i))/1.0e6;
  velocity_i = increment/deltaT;
  prevT_i = currT;
}
```

Fig C.2: Arduino code for motor shaft speed PID feedback control.