Implementation Model Predictive Control (MPC) Algorithm-3 for Inverted Pendulum

Cahyantari Ekaputri
Department of Electrical Engineering
Bandung Institute of Technology
Bandung, Indonesia
cahyantari.ekaputri@gmail.com

Arief Syaichu-Rohman
Department of Electrical Engineering
Bandung Institute of Technology
Bandung, Indonesia

***Abstract*— Model Predictive Control (MPC) is based on the idea to produce control input as a solution to real-time optimization problem. Optimization itself is based on the system model. MPC is used to solve multivariable control problem which has more than one variable that may have significant effect on the process. The advantage of MPC is that MPC can works effectively within constraints of the real actuator which are relatively narrow. The disadvantage of MPC lies on its complex algorithm that needs longer time than the other controller. This paper discusses the MPC application for inverted pendulum. The algorithm used is algorithm-3 for floating point. The general guidelines are modeling the system, designing the system using MATLAB, simulating the design using MATLAB, implementing the MPC algorithm on AVR ATmega32 and analyzing the result. After the guideline have been implemented, it can be seen that MPC algorithm-3 with 9 horizon and 1 iteration of quadratic programming can return the pendulum rod at the balance point by the time external force applied.**

Keywords—Model Predictive Control (MPC); algorithm-3; inverted pendulum; AVR ATmega32

I. INTRODUCTION

Model Predictive Controller (MPC) is a control theory which is quite often used to control the complex system domain. MPC has been used in recent years in industrial processes. MPC is very effective to improve the optimization and address the problem of control is usually in the form of the constraints which is owned by the actuator. These constraints are the problem in the design of the system as most systems operate optimally in the working area close to the limit. The main principle of MPC is how to select appropriate control measures to solve the problem of repetition in the optimum control. The goal is the minimization of the performance criteria of the system to a certain horizon, which is usually caused by limitations in the input and output of future behavior which computed based on a process model. The problem that encountered often in the design of MPC is that how to guarantee the stability of the closed loop system, cope with the uncertainty of the model was designed, and reduce the computation performed.

In this paper, the application of MPC will be implemented in microcontroller for controlling the inverted pendulum ECP model 505.

This paper has five sections, introduction section, basic theory section, design and simulation section, implementation and analysis section, and conclusion section.

Section I introduces the general information about this paper. Section II discusses the basic concept of MPC, the computation of MPC tracking problem and MPC algorithm-3 structure. Section III discusses the design and implementation; the plant which is inverted pendulum ECP model 505 will be model and the design of the MPC controller. Section IV discusses the system implementation (including hardware and the algorithm) and the analysis of implemented system. Section V contains the conclusion based on the test from section IV.

II. MODEL PREDICTIVE CONTROL (MPC)

A. The Concept of MPC

MPC will calculate a value which will be used for the input process as a real-time solution of the optimization problem. MPC predict the next output from a process in a certain length of the horizon of a time with using the input and output from previous process. After that the calculation for the control signal in that horizon of time and optimization will be start. Figure 1 show the MPC base diagram of the optimization problem.

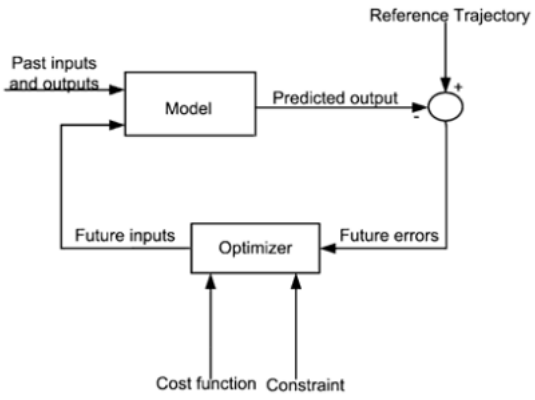

Figure 1. MPC Base Diagram

Figure 2 shows the illustration of MPC base concept which predicts the behavior of plant within M horizons. The prediction's accuracy depends on the accuracy of plant model.

MPC computes the optimal output (control signal) for each input that will concern the constraints. The responses from future control signal computed as problem solvers for n-horizons open loop which determined with first element from optimal control signal responses that used as the next process inputs, meanwhile the other elements will not be used. After that the time will shifted and the input will be computed again so that the computation acquires a new optimal solution that will be used for the next computation. This process always repeated so the output will be optimal.

.

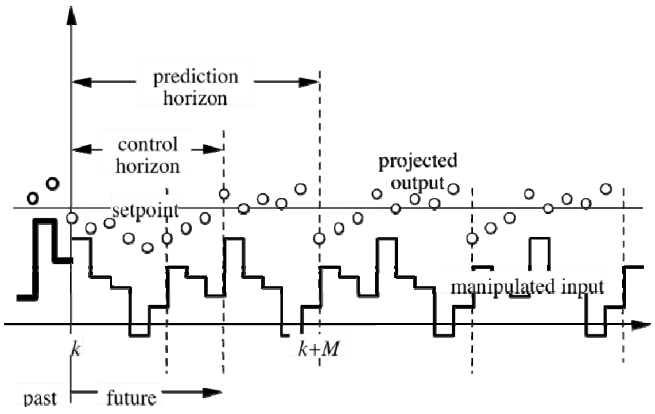

Figure 2. MPC Base Concept

B. MPC Tracking Problem

Consider a detectable plant

$$x_{k+1} = \tilde{A}x_k + \tilde{B}u_k \quad (1)$$

$$y_k = \tilde{C}x_k \quad (2)$$

where $x_k \in R^n$, $y_k \in R^l$, and $u_k \in R^m$.

The output y_k is required to track an input reference $w_k \in R^l$. To achieve zero tracking error, an integrator in the form of

$$u_k = u_{k-1} + \Delta u_k \quad (3)$$

is augmented to the plant so that the augmented plant can be written as

$$\bar{x}_{k+1} = \begin{bmatrix} x_{k+1} \\ u_k \end{bmatrix} = A\bar{x}_k + \Delta B u_k \quad (4)$$

$$y_k = C\bar{x}_k \quad (5)$$

where $A \coloneqq \begin{bmatrix} \tilde{A} & \tilde{B} \\ 0 & I_m \end{bmatrix}$, $B \coloneqq \begin{bmatrix} \tilde{B} \\ I_m \end{bmatrix}$, and $C \coloneqq [\tilde{C} \quad 0]$.

Define the objective function as

$$J = e_{k+Ny}^T \bar{Q}_N e_{k+Ny} + \sum_{t=0}^{Ny-1} e_{k+t}^T \bar{Q} e_{k+t} + \sum_{t=0}^{Nu-1} \Delta u_{k+t}^T \bar{R} \Delta u_{k+t} \quad (6)$$

where $e_k = y_k - z_k$, N_y = prediction horizon, N_u = control horizon, $N_u \leq N_y$, $\bar{Q} = \bar{Q}^T \geq 0$ and $\bar{R} = \bar{R}^T \geq 0$. For the simplicity, assume $N = N_u = N_y$ to define

$$Y = \begin{bmatrix} y_{k+1} \\ y_{k+2} \\ \vdots \\ y_{k+N} \end{bmatrix}, W = \begin{bmatrix} w_{k+1} \\ w_{k+2} \\ \vdots \\ w_{k+N} \end{bmatrix}, U_d = \begin{bmatrix} \Delta u_k \\ \Delta u_{k+1} \\ \vdots \\ \Delta u_{k+N-1} \end{bmatrix} \quad (7)$$

and have

$$Y = \Phi\, \bar{x}_k + \Gamma\, U_d \quad (8)$$

$$J = J_1 + (Y - W)^T Q (Y - W) + U_d^T R U_d \quad (9)$$

with

$$J_1 = (y_k - w_k)^T \bar{Q} (y_k - w_k) \quad (10)$$

$\mathrm{Q} = \mathrm{diag}\{\mathrm{Q_i}\}$, $\mathrm{Q_i} = \bar{\mathrm{Q}}$ for $\mathrm{i} = 1,2,\ldots,\mathrm{N}$

$\mathrm{R} = \mathrm{diag}\{\mathrm{R_i}\}$, $\mathrm{R_i} = \bar{\mathrm{R}}$ for $\mathrm{i} = 1,2,\ldots,\mathrm{N}$

$$\Phi = \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^N \end{bmatrix} = \begin{bmatrix} \tilde{C}\tilde{A} & \tilde{C}\tilde{B} \\ \tilde{C}\tilde{A}^2 & \tilde{C}(\tilde{A} + I_n)\tilde{B} \\ \vdots & \vdots \\ \tilde{C}\tilde{A}^N & \tilde{C}\left(\sum_{i=0}^{N-1} \tilde{A}^i\right)\tilde{B} \end{bmatrix} \coloneqq [\Phi_1 \quad \Phi_2] \quad (11)$$

$$\Gamma = \begin{bmatrix} CB & 0 & \cdots & 0 & 0 \\ CAB & CB & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ CA^{N-1}B & CA^{N-2}B & \cdots & CAB & CB \end{bmatrix} \quad (12)$$

If the amplitude limit is considered in the equation, firstly must define

$$U_d = DU - E_1 u_{k-1} \coloneqq GX + DU \quad (13)$$

with

$$X = \begin{bmatrix} x_k \\ u_{k-1} \\ -Z \end{bmatrix}, G = [0 \quad -E_1 \quad 0] \quad (14)$$

$$D = \begin{bmatrix} I_m & 0 & \cdots & 0 & 0 \\ -I_m & I_m & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & I_m & 0 \\ 0 & 0 & \cdots & -I_m & I_m \end{bmatrix} \quad (15)$$

they give

$$Y - W = [\Phi_1 \quad \Phi_2]\begin{bmatrix} x_k \\ u_{k-1} \end{bmatrix} + \Gamma(DU - E_1 u_{k-1}) - Z \quad (16)$$

$$Y - W = \bar{\Phi}\mathrm{X} + \bar{\Gamma}\mathrm{U} \quad (17)$$

With these defined variables, the beginning equation can be written to

$$J = J_1 + X^T(\bar{\Phi}Q\bar{\Phi} + G^T R G - F_X^T M^{-1} F_x)X + 2J_{To} \quad (18)$$

where $\bar{\Phi}$ and $\bar{\Gamma}$ are the functions from A, B, C matrices.

$$X := [x_k^T \quad u_{k-1}^T \quad -Z^T] \tag{19}$$

$$M := \bar{\Gamma}^T Q \bar{\Gamma} + D^T R D \tag{20}$$

$$F_1 := D^T \Gamma^T Q \Phi_1 \tag{21}$$

$$F_2 := D^T \Gamma^T Q \Phi_2 - D^T (\Gamma^T Q \Gamma + R) E_1 \tag{22}$$

$$F := [F_1 \quad F_2] \tag{23}$$

$$F_x := [F \quad H] \tag{24}$$

$$H := D^T \Gamma^T Q \tag{25}$$

$$\bar{\Phi} = [\Phi_1 \quad (\Phi_2 - \Gamma E_1) \quad I_N] \tag{26}$$

$$\bar{\Gamma} = \Gamma D \tag{27}$$

$$E_1 = [1 \quad 0 \quad \ldots \quad 0] \tag{28}$$

The benefit of computation is to find the U variable which minimize

$$J_{To} := \frac{1}{2} (U + M^{-1} F_X X)^T M (U + M^{-1} F_X X) \tag{29}$$

where $1u_{min} \leq U \leq 1u_{max}$

with definition of this equation first

$$U_O := -M^{-1} F_x X = -M^{-1} [F \quad H] \begin{bmatrix} \bar{x}_k \\ -Z \end{bmatrix} \tag{30}$$

According to the quadratic equivalent programming, MPC tracking can be implemented as the block diagram at Figure 3.

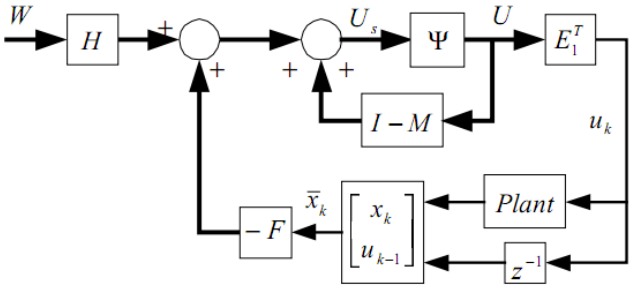

Figure 3. Block Diagram of the MPC Tracking Implementation

C. Algorithm-3

Algorithm-3 is the iteration of the quadratic programming algorithm. This algorithm can be represented in Figure 4.

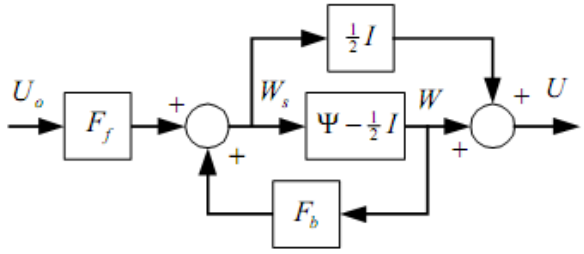

Figure 4. Block Diagram of the Algorithm-3

The iteration of the algorithm can be seen if there is a feedback. Figure 4 can be written to

$$W_s = F_f U_o + F_b W \tag{31}$$

$$V = \Psi_b(W_s) \tag{32}$$

The definition of the iteration implementation from that variable is

$$W_{k+1} = \Psi_b(F_f U_o + F_b W) \tag{33}$$

with Ψ_b that a nonlinear function from actuator saturation.

$$\Psi_b(x) := \Psi - \frac{1}{2} I := \Psi_a(x) - \frac{1}{2} x \tag{34}$$

If the iteration equation above has a convergent characteristic, the solution of that loop can be defined as

$$U^* = W^* + \frac{1}{2}(F_f U_o + F_b W^*) \tag{35}$$

$$F_f := 2(I + M)^{-1} M;\ F_b := 2(I + M)^{-1}(I - M) \tag{36}$$

III. Design and Simulation

A. Inverted Pendulum

The inverted pendulum that will be used is ECP model 505. The design is demanded to be automatically stay on the balance state with a tolerance where close to it. The system will start from the furthest position of pendulum rod from the balance point and will work to keep it on the balance state.

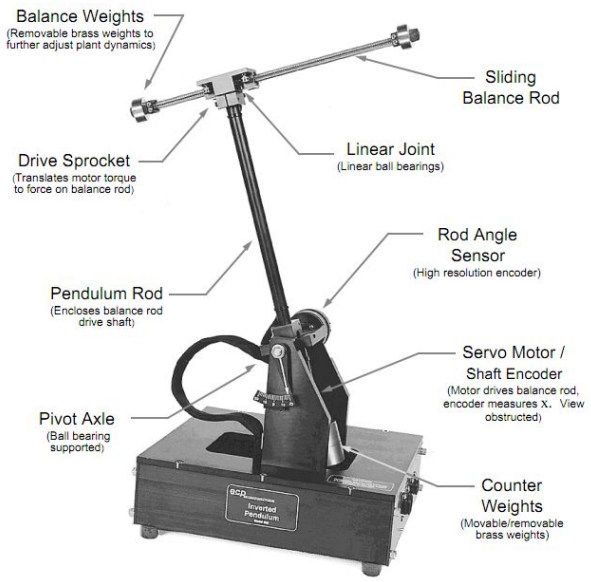

Figure 5. Inverted Pendulum ECP Model 505

The ECP model 505 has the transfer function shown on (37).

$$\frac{\theta(s)}{F(s)} = \frac{l_0}{J^*} \frac{-s^2 + \frac{g}{l_0}}{s^4 + \left(\frac{(m_1 l_0 - m_2 l_c) g}{J^*}\right) s^2 - \frac{m_1 g^2}{J^*}} \tag{37}$$

with

$$J^* = J_0 - m_1 {l_0}^2 \quad (38)$$

$$J_0|_{x=0} = J_1 + J_2 + m_1 {l_0}^2 + m_2 {l_c}^2 \quad (39)$$

In this system, there are four state, one input and one output. The input that used in this system is the amount of force which necessary to move the pendulum rod in order to achieve stability when standing vertical $F(t)$, while the output of this system is the large of pendulum angle / position of the pendulum rod to the vertical axis (θ). The states that used in this system written on (40).

$$x = [\theta \quad \dot{\theta} \quad x \quad \dot{x}]^T \quad (40)$$

where

θ = angle / position of the pendulum rod to the vertical axis

$\dot{\theta}$ = angular velocity while moving the pendulum rod

x = position sliding rod of the pendulum rod

$\dot{x}$ = shaft speed glide while moving

The parameters of inverted pendulum used for computation are:

l_0 : length of pendulum rod (0.33 m)

l_c : length of mass centre of pendulum rod (0.0281 m)

m_1 : mass of sliding rod (0.213 kg)

m_2 : mass of pendulum rod (1.785 kg)

g : gravitation force (9.81 m^2/s)

J_0 : moment of inertia (0.036 $kg.m^2$)

State space of the inverted pendulum system can be modeled mathematically in the form of discrete space equation as follows.

$$x_{k+1} = A_{bar}\, x_k + B_{bar}\, u_k \quad (41)$$

$$y_k = C_{bar}\, x_k + D_{bar}\, u_k \quad (42)$$

By using the parameter value sampling time 0.04 seconds and compute using MATLAB functions `tf2ss` and `c2d`, the system of mathematical equations in discrete form

$$x_{k+1} = \begin{bmatrix} 1.1303 & 0.6474 & 0 & 0 \\ -0.3354 & 1.1303 & 0 & 0 \\ 0 & 0 & 0.6031 & -0.3473 \\ 0 & 0 & 0.3473 & 0.9092 \end{bmatrix} x_k + \begin{bmatrix} -0.2573 \\ -0.7005 \\ -0.6962 \\ -0.0134 \end{bmatrix} u_k$$

$$y_k = [0.2337 \quad 0.8681 \quad -0.6962 \quad 0.0134]\, x_k + [0]\, u_k$$

The input and output of the equation are trough stabilization realization state space based on Gramian. The parameters that used in this paper define by:

- Sampling time : 0.04 seconds
- $\bar{Q} = 1$ and $\bar{R} = 1$
- Set point value : 0
- Simulation time : 16 seconds

B. MPC Designing in MATLAB

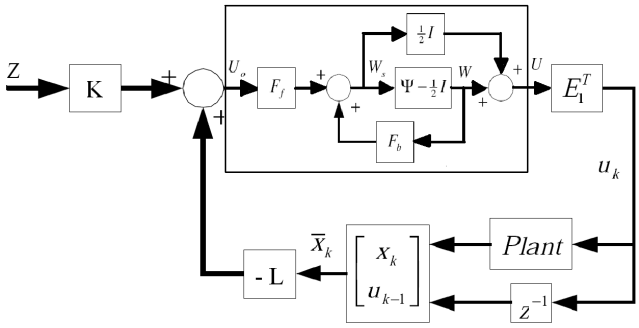

Figure 6. Block Diagram of the Algorithm-3 Implementation

MPC Designing is calculating the matrices values which become the bases of MPC Algorithm-3 calculation. Calculation of the matrices defines by designing the MPC formula in MATLAB. With simulink in MATLAB, the optimum horizon value calculated for the system with a certain range of iteration value, using the method that analyzing the response time and control signal of the system.

These conditions make the value of 9 horizons has the smallest settling time response system and input system. Table I shows the details.

TABLE I. SETTLING TIME RESPONSE SYSTEM, INPUT SYSTEM, AND INTERATION NUMBER AT EVERY HORIZON

Horizon	Response Time	Input Control Signal Time	Iteration Number
Horizon 8	13,84 s	12 s	1
Horizon 9	8,92 s	7,2 s	1
Horizon 10	9,2 s	7,48 s	1
Horizon 11	10,36 s	8,36 s	1
Horizon 12	11,76 s	9,44 s	1
Horizon 15	15,2 s	12,2 s	1

The most optimal control signal is reached when the horizon value is 9. The system characteristics shown in Figure 7 and figure 8.

Figure 7 shows the settling time response system which controlled by MPC algorithm 3 with 9 horizons. From the graphic, the response time of system is 8,92 s. Figure 8 shows the input control signal time of system which controlled by MPC algorithm 3 with 9 horizons. From the graphic, the input control signal time of system is 7,2 s.

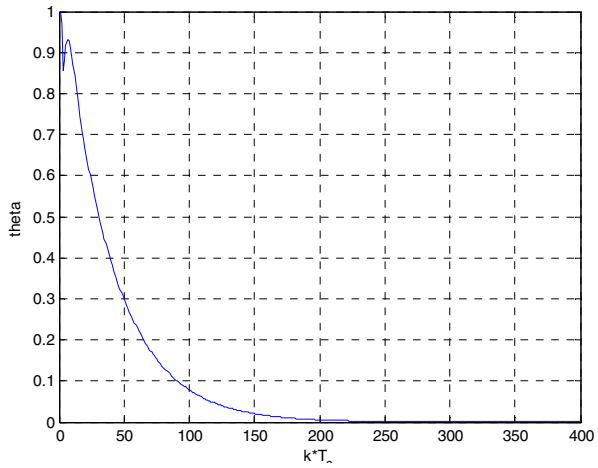

Figure 7. MPC Algorithm-3 with 9 horizons Response Time

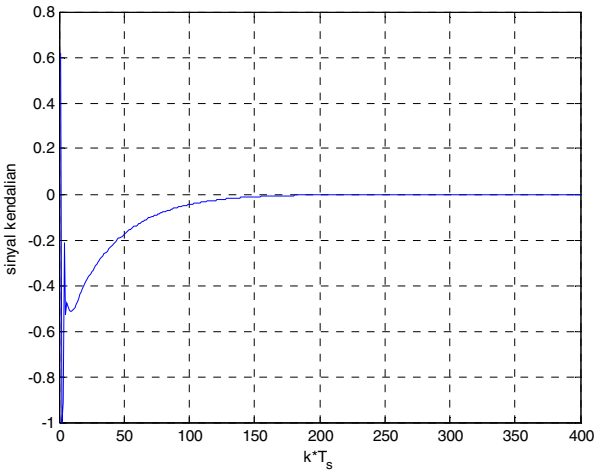

Figure 8. MPC Algorithm-3 with 9 horizons Input Control Signal

IV. IMPLEMENTATION AND ANALYSIS

A. Hardware System Design

The hardware system divided into three parts of modules, which are the AVR ATmega32 microcontroller, L298N motor driver IC, and inverted pendulum ECP model 505 as a plant. The block diagram (Figure 9) shows the flow of design.

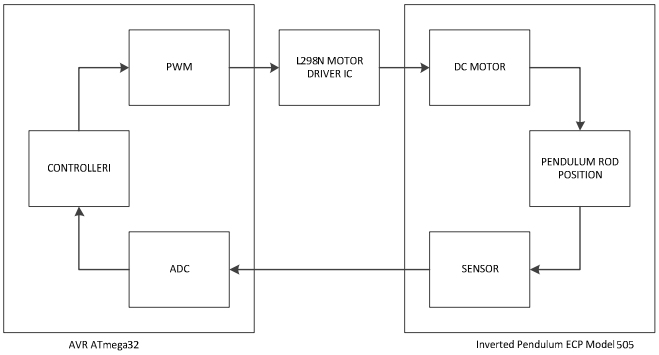

Figure 9. Block Diagram of the Hardware System

B. System Algorithm Design

The algorithm of the system design by the flowchart in Figure 10. This algorithm will be applied in microcontroller.

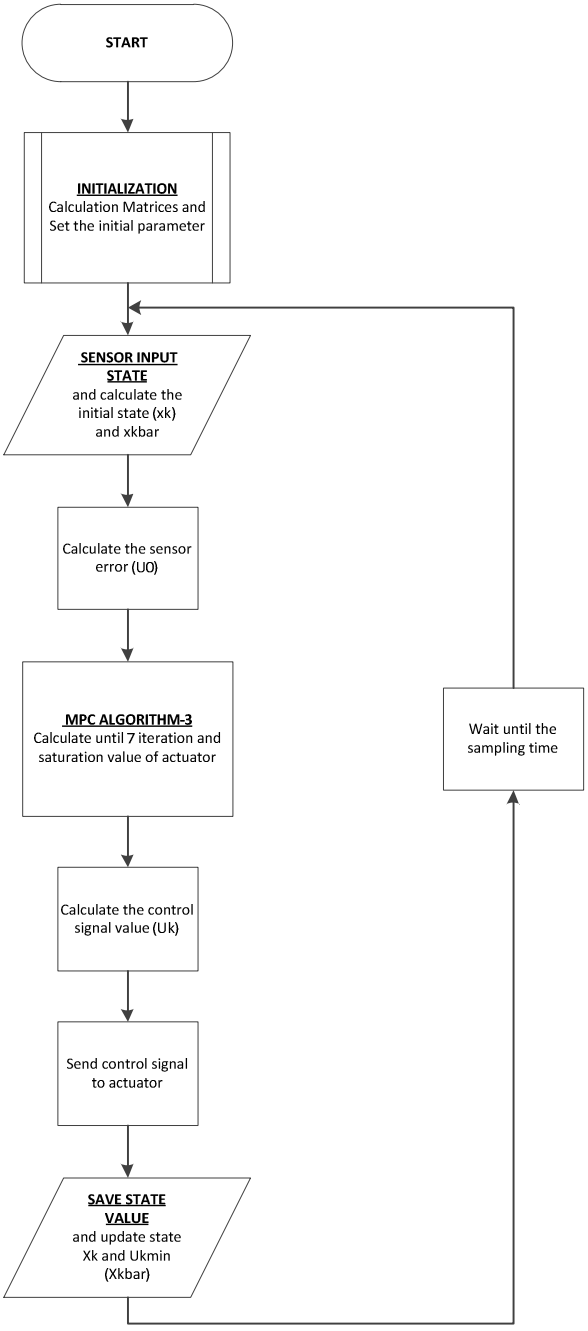

Figure 10. Flowchart of System Algorithm

C. System Implementation

The next step is implementing the design of hardware and algorithm. Then the system will be verified with the result of simulation. The verification step will be divided into three parts, which are input control signal verification, PWM verification and system verification.

The input control signal verification used for verified the input control signal with the input sensor value. From simulation, it's clearly show that if the pendulum rod is around the stable point (ADC conversion value near 0), then the input control signal value will be small (near 0). Otherwise, if the pendulum rod is away from the stable point (ADC conversion value near 1), then the input control signal value will be large. Table II shows the verification.

TABLE II. INPUT CONTROL SIGNAL VERIFICATION

Sensor Input (ADC)	ADC Conversion Value	Input Control Signal Value
166	-1	-0,66
176	-0,87	-0,57
192	-0,64	-0,42
216	-0,31	-0,2
240	0,03	0,016
267	0,41	0,267
312	1,04	0,68

The PWM verification used for verified the input control signal value with the output PWM from controller (apply to motor DC of the plant). The PWM value is about 1 to 1023. In simulation, if the input control signal value is small, then the output PWM will be small. Otherwise, if the input control signal value is large, then the output PWM will be large. Table III shows the verification.

TABLE III. PWM VERIFICATION

Input Control Signal Value	PWM Value
-0,66	679
-0,57	585
-0,42	434
-0,2	207
0,016	16
0,267	272
0,68	699

The system verification performed by giving interference of pendulum rod in the stable point, then the pendulum rod will be back to its stable point.

Figure 11 and 12 shows the system verification. In figure 11, the above graphic (CH1) shows the pendulum rod position (angle) and the bottom graphic (CH2) shows the PWM which applied as the input control signal of the system. It's clearly show that if there is an interference in the pendulum rod (pendulum rod position is not in stable point), the MPC controller will calculate the input control signal so that the pendulum rod will be in the stable point. The input control signal will be changed into PWM signal. Then PWM signal will be applied to DC motor in the plant. Figure 12 shows that the settling time of input control signal is 8 seconds (about 200 times with time plotting 0.04 seconds using serial communication in MATLAB).

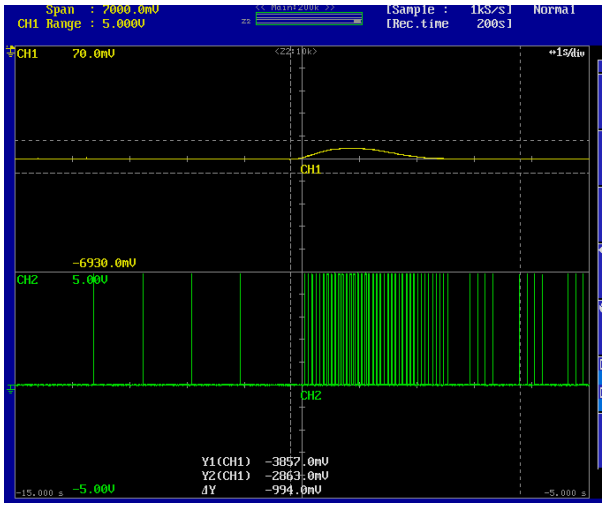

Figure 11. Pendulum Rod Position and PWM Value of System Verification

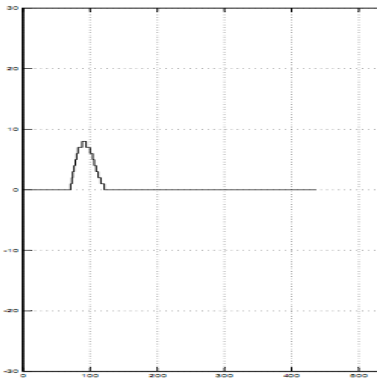

Figure 12. Input Control Signal Plot

D. *Analysis*

In designing MPC, sampling time is 0.04 seconds. Rule of Thumb says that the ideal sampling of a system is one-tenth of time constant of system (it can be seen from its characteristic equation). Nyquist-Shannon Theorem says the minimal sampling time is half of time constant of system. Based on Nyquist-Shannon Theorem, the chosen sampling time satisfy that theorem.

The settling time of input control signal is 8 seconds. But in the simulation, the settling time is 7.2 seconds. This is caused by delay of system (in actuator, sensor, etc.), the non-ideal actuator (slip, hysteresis, etc.), the accuracy sensor, the modeling system (system is non-linear, but the modeling is linearized) etc.

V. CONCLUSION

The paper shows that MPC algorithm 3 can be applied to inverted pendulum system ECP model 505 and implemented in AVR ATmega32 microcontroller. The Algorithm-3 calculate the optimal input control system based on the prediction of system's behavior. The algorithm-3 also solve the constraints problem in actuator. The system performed by giving interference of pendulum rod in the stable point, then the pendulum rod will be back to its stable point. For this system, the horizon value set at 9 based on the response time in the

simulation. For further research, the MPC algorithm can be implemented in more complex processor for complex computation.

REFERENCES

[1] Franklin, G. F. dan J. D. Powell, and A. Emami-Naeni (1997), *Feedback Control on Dynamic Systems, 3rd ed*, Addison Wesley, New York.

[2] Gorinevsky, Dimitry. *Model Predictive Control: Industrial MPC,* Stanford University. www.stanford.edu/class/ee392m/Lecture15_MPC.pdf.

[3] Nikolaou, Michael. *Model Predictive Controllers: a Critical Synthesis of Theory and Industrial Needs*. www.chee.uh.edu/faculty/nikolaou/MPCtheoryRevised.pdf.

[4] Ogata, Katsuhiko (1997), *Modern Control Engineering, 3rd ed*, Prentice-Hall, New Jersey

[5] Orukpe, P.E. *Basics of Model Predictive Control*, Imperial College, London. www3.imperial.ac.uk/portal/pls/portallive/docs/1/50918.PDF.

[6] Syaichu-Rohman, Arief dan Richard H. Middleton - *A Multivariable Nonlinear Algebraic Loop*. http://www.nt.ntnu.no/users/skoge/prost/proceedings/ecc03/pdfs/001.pdf

[7] Syaichu-Rohman, Arief, and Middleton, Richard H. *Convergence Study of Some Fixed Point Iteration QP Algorithms*. In Proceedings of the 43rd Conference on Decision and Control. Pradise Island, Bahamas.

[8] Syaichu-Rohman, Arief. *Plant Performance Optimization*. The 6th Asian Control Conference. July 18-21, 2006. Inna Grand Bali Beach Hotel, Sanur, Bali, Indonesia.

[9] Syaichu-Rohman. 2006. *Introduction to Model Predictive Control.* Indonesia: Workshop of the 6th Asian Control Conference.

[10] Taylor, Rod Jason. 2008. *Model Predictive Control dengan Algoritma-2 Quadratic Program pada dsPIC30F4011*. Institut Teknologi Bandung, Indonesia.