Perancangan dan Implementasi Pengaturan Kecepatan Motor *Brushless DC* Menggunakan Metode *Model Predictive Control* (MPC)

Fachrul Arifin, Josaphat Pramudijanto, dan Ali Fatoni
Jurusan Teknik Elektro, Fakultas Teknologi Industri, Institut Teknologi Sepuluh Nopember (ITS)
Jl. Arief Rahman Hakim, Surabaya 60111
E-mail: jos@elect-eng.its.ac.id, fatoni@ee.its.ac.id.

***Abstrak*— Saat ini, penelitian dan pengembangan mobil listrik (electric vehicle) sudah menjadi pusat perhatian bagi kalangan industri dan civitas academica. Sudah banyak kendaraan listrik yang sudah diproduksi secara massal di seluruh dunia. Beberapa dari kendaraan listrik tersebut menggunakan motor BLDC sebagai penggerak utamanya. Pada Tugas Akhir kali ini, salah satu kemampuan yang akan diteliti dan dianalisa adalah respon kecepatan motor BLDC pada mobil listrik saat mobil diberi suatu beban. Ketika mendapat pembebanan, respon kecepatan dari motor BLDC akan turun sehingga performa dari motor BLDC tidak sesuai dengan setpoint yang diharapkan.**

Oleh karena itu, dibutuhkan suatu kontroler yang dapat mengatasi permasalahan diatas dan melakukan pengaturan kerja pada motor BLDC agar bekerja sesuai dengan kebutuhan. Kontroler ini diharapkan mampu untuk mengembalikan respon kecepatan pada motor BLDC kembali kepada setpoint ketika motor BLDC diberi pembebanan.

Penulis menggunakan metode kontroler *Model Predictive Control* (MPC) yang dapat memprediksi perilaku sistem pada masa depan yang bergantung pada informasi sistem saat ini dan model *state-space* dari sistem. Berdasarkan hasil implementasi, kontroler MPC dapat memberikan respon yang sesuai dengan *tracking setpoint* yang diberikan dengan rata-rata nilai *steady-state error* sebesar 9,8% untuk semua parameter pembebanan.

Kata Kunci*—Kendaraan Listrik, Motor *Brushless DC*, *Model Predictive Control

I. PENDAHULUAN

Saat ini, seperti kita telah ketahui bersama bahwa cadangan dan persediaan bahan bakar fosil dari tahun ke tahun cenderung menipis. Kabar ini tentu saja menjadi hal yang buruk bagi dunia otomotif yang mengandalkan konsumsi bahan bakar fosil yang saat ini menjadi sumber utama penggerak kendaraan bermotor. Tentu saja, hal ini mendorong umat manusia untuk mencari bahan bakar alternatif dan lebih efisien dibandingkan bahan bakar fosil. Saat ini, telah banyak ditemukan berbagai macam kendaraan bermotor berbasiskan energi alternatif yang lebih murah dan efisien dibandingkan bahan bakar fosil. contohnya adalah mobil listrik yang saat ini menjadi tren dan diproyeksikan menjadi kendaraan masa depan.

Kendaraan listrik merupakan kendaraan yang menggunakan murni energi listrik sebagai penggeraknya. Pada kendaraan konvensional, bahan bakar dari minyak bumi yang telah diproses, contohnya premium dan pertama, digunakan untuk menjalankan mesin diesel biasa. Sedangkan pada kendaraan listrik, digunakan sumber energi listrik yang berasaldari baterai untuk menggerakkan kendaraan listrik tersebut.

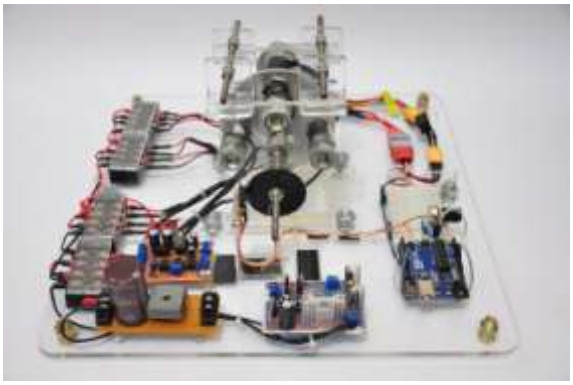
Dalam penggunaanya, kendaraan listrik tersebut tidak lepas dari adanya suatu beban yang ditanggung oleh kendaraan tersebut. Beban tersebut tentu dapat mengurangi performa dan respon kecepatan yang akan turun akibat adanya beban terebut. Oleh karena itu, diperlukan suatu kontroler yang berfungsi untuk mengembalikan respon kecepatan motor BLDC kembali pada referensi yang diinginkan ketika mobil listrik tersebut diberi pembebanan.

Metode *Model Predictive Control* (MPC) digunakan untuk mengatur respon kecepatan motor BLDC kembali ke referensi yang diinginkan walaupun mobil listrik diberi pembebanan. Metode ini memprediksi perilaku sistem pada masa depan yang bergantung pada informasi sistem saat ini dan model *state-space* dari sistem tersebut [1]. Metode ini diharapkan dapat mengembalikan respon kecepatan motor BLDC ke referensi yang diinginkan.

II. PENGENALAN PLANT DAN METODE KONTROL

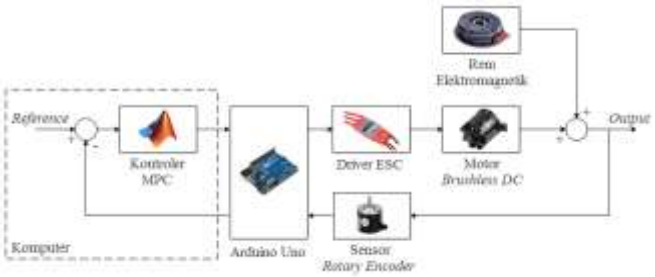
A. Plant BLDC V-1

Pada pelaksanaan Tugas Akhir kali ini, akan digunakan sebuah *plant* atau sistem yang terdiri dari komponen utama berupa motor *Brushless DC* (BLDC). Sistem ini dikerjakan oleh 5 orang dalam satu tim dan kami beri nama *plant* BLDC V-1. Motor BLDC ini digunakan sebagai penggerak utama dari kendaraan listrik. Selain itu, digunakan rem elektromagnetik untuk memberikan efek pembebanan pada motor BLDC. Gambar 1 menunjukkan *Plant* BLDC V-1 yang digunakan pada Tugas Akhir kali ini.



Gambar 1. Bentuk Fisik *Plant* BLDC-V1.

Sistem ini terdiri dari motor BLDC, *driver* motor BLDC berupa ESC (*Electronic Speed Control*), sensor *rotary encoder* dan rem elektromagnetik sebagai komponen utamanya. Mikrokontroler Arduino digunakan sebagai alat akuisisi data pada sistem. Untuk pengaturan tegangan yang masuk ke dalam rem, digunakan *driver* yang diberi masukan PWM yang berasal dari Arduino. Gambar 2 menunjukkan blok diagram sistem pengaturan motor BLDC.



Gambar 2. Blok Diagram Sistem Pengaturan Motor BLDC.

B. Identifikasi Dinamis [2]

Identifikasi dinamis digunakan untuk mendapatkan pemodelan dari sistem motor BLDC yang sudah diberi beban tertentu. Identifikasi dinamis dilakukan dengan cara memberi sistem sebuah masukan sinyal acak berupa sinyal *Pseudo-Random Binary Sequence* (PRBS). Sinyal PRBS mirip dengan bilangan acak secara nyata, tapi juga dapat disebut semu atau *pseudo* karena bersifat deterministik.

Setelah mendapatkan output dari sistem setelah diberi sinyal PRBS, langkah selanjutnya adalah menentukan pemodelan yang akan digunakan. Pada Tugas Akhir kali ini, digunakan pendekatan ARX untuk mendapatkan fungsi alih dari sistem. Pendekatan ARX dapat tuliskan sebagaimana Persamaan 1.

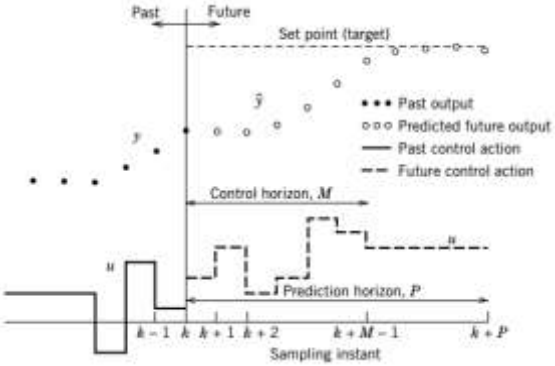
$$\begin{aligned} &y(t)+a_1y(t-1)+...+a_{n_a}y(k-n_a)= \\ &b_1u(t-n_k)+...+b_{n_b}u(t-n_b-n_k+1)+e(t) \end{aligned} \quad (1)$$

$y(t)$: *Output* pada waktu *t*
n_a : Banyaknya jumlah *pole*
n_b : Banyaknya jumlah *zero* ditambah satu
n_k : Nilai *dead time.* Nilai ini merupakan jumlah masukan yang terjadi sebelum memberikan pengaruh pada keluaran.

C. Model Predictive Control (MPC) [1]

Tujuan utama dari sebuah *Model Predictive Control* (MPC) adalah untuk menghitung trayektori dari sinyal kontrol *u* (*manipulated variable*) yang akan datang untuk mengoptimalkan perilaku yang akan datang (*future behavior*) pada sinyal *output y* pada sebuah *plant*. Perhitungan pada kontroler MPC didasarkan pada nilai pengukuran saat ini dan prediksi dari nilai *output* yang akan datang. Objektif dari kontroler MPC adalah untuk menentukan nilai sinyal kontrol (*sequence of control moves*) sehingga nilai *output* yang diprediksi akan mendekati nilai *setpoint* dengan optimal.

Pada Gambar 3, dapat dilihat susunan dari nilai *output* saat ini (*actual output*) *y*, nilai *output* terprediksi (*predicted output*) $\hat{y}$, dan *manipulated input* atau sinyal kontrol *u*. Pada setiap waktu *sampling k*, kontroler MPC menghitung himpunan dari nilai *M* atau *control horizon* (selanjutnya disebut N_c) dari *input* $\{u(k+i-1), i=1,2,...,M\}$. Nilai *input* akan ditahan pada nilai konstan setelah *M* langkah pada sinyal kontrol tersebut. Nilai *input* akan dihitung sedemikian sehingga nilai himpunan dari *P* keluaran atau *output* terprediksi $\{y(k+i), i=1,2,...,P\}$ akan mencapai nilai *setpoint* yang diinginkan. *P* merupakan nilai dari *prediction horizon* (selanjutnya disebut N_p) pada kontroler MPC. Perhitungan nilai kontrol pada kontroler MPC dihitung berdasarkan nilai optimal dari suatu fungsi objektif atau indeks performansi *J*. [3]



Gambar 3. Konsep Dasar dari Kontroler *Model Predictive Control* (MPC). [3]

Pertama, kita mendefinisikan sebuah sistem *single-input single-output* sebagaimana Persamaan 2.

$$\begin{aligned} x_m(k+1) &= A_m x_m(k)+B_m u(k) \\ y(k) &= C_m x_m(k)+\mathrm{D}_m u(k) \end{aligned} \quad (2)$$

Untuk dapat menerapkan kontroler MPC pada suatu *plant*, maka kita perlu mengubah bentuk *state space* Persamaan 2 menjadi bentuk *augmented model*. Adapun bentuk *augmented model* dapat didefinisikan sebagai Persamaan 3.

$$\begin{aligned} \begin{bmatrix} \Delta x_m(k+1) \\ y(k+1) \end{bmatrix} &= \begin{bmatrix} A_m & 0_m^T \\ C_m A_m & 1 \end{bmatrix} \begin{bmatrix} \Delta x_m(k) \\ y(k) \end{bmatrix} + \begin{bmatrix} B_m \\ C_m B_m \end{bmatrix} \Delta u(k) \\ y(k) &= \begin{bmatrix} 0_m & 1 \end{bmatrix} \begin{bmatrix} \Delta x_m(k+1) \\ y(k+1) \end{bmatrix} \end{aligned} \quad (3)$$

Adapun $o_m = \begin{bmatrix} 0 & 0 & \cdots \end{bmatrix}$ dan matriks *A,B,C* biasa disebut *augmented model*. Model *state space* inilah yang akan digunakan selanjutnya untuk merancang sebuah kontroler MPC.

Setelah mendefinsikan augmented model, langkah selanjutnya adalah menentukan matriks *predicted output* (*F*) dan matriks *future control* (*Φ*) berdasarkan Persamaan 4.

$$Y = Fx(k_i) + \Phi\Delta U \tag{4}$$

$$F = \begin{bmatrix} CA \\ CA^2 \\ CA^3 \\ \vdots \\ CA^{N_p} \end{bmatrix} \quad ; \quad \Phi = \begin{bmatrix} CB & 0 & \cdots & 0 \\ CAB & CB & \cdots & 0 \\ CA^2B & CAB & \cdots & 0 \\ \vdots & \vdots & \ddots & \\ CA^{N_p-1}B & CA^{N_p-2}B & \cdots & {}^{N_p,-N_c}B \end{bmatrix} \tag{5}$$

N_c merupakan nilai *control horizon*, yaitu jumlah langkah kontrol berkelanjutan yang diterapkan dan diprediksi oleh kontroler MPC dalam sebuah *sampling time.* Selain itu, variabel *output* terprediksi dapat diperkirakan dan diprediksi dalam jumlah sampel N_p, dimana N_p merupakan nilai *prediction horizon*. Sebagai catatan, nilai N_c harus lebih kecil atau sama dengan nilai N_p.

Dalam sebuah kontroler MPC, diperlukan proses optimasi yang mempunyai objektif kontrol untuk meminimalkan *error* yang terbentuk dari selisih nilai referensi dengan nilai keluaran dari *plant*. Optimasi tersebut dilakukan dengan mendeskripsikan sebuah nilai dan parameter indeks performansi J yang merefleksikan objektif kontrol dari kontroler MPC. Indeks performansi tersebut dapat didefinisikan sebagai berikut:

$$J = (R_s - Y)^T (R_s - Y) + \Delta U^T \overline{R} \Delta U \tag{6}$$

$$R_s^T = \overbrace{\begin{bmatrix} 1 & 1 & \cdots \end{bmatrix}}^{N} \tag{7}$$

R_s merupakan vektor yang mempunyai informasi sinyal referensi atau *set-point* yang didefinisikan seperti tertulis pada Persamaan 6. Matriks $\overline{R}$ adalah matriks diagonal yang berbentuk $\overline{R} = r_w I_{N_c \times N_c} (r_w \geq 0)$ dan digunakan sebagai parameter *tuning* kontroler MPC. Untuk mendapatkan nilai kontrol optimal yang akan meminimalisasi indeks performansi J, kita dapat mengekspresikan indeks performansi J pada Persamaan 8 sebagai berikut:

$$\begin{aligned} J &= (R_s - Fx(k_i))^T (R_s - Fx(k_i)) - 2\Delta U^T \Phi^T \\ &\quad (R_s - Fx(k_i)) + \Delta U^T (\Phi^T \Phi + \overline{R}) \Delta U \end{aligned} \tag{8}$$

Untuk mendapatkankan kontrol optimal dari kontroler MPC, maka Persamaan 8 diturunkan terhadap ΔU dan memenuhi kondisi $\frac{\partial J}{\partial \Delta U} = 0$. Oleh karena itu, solusi optimal dari sinyal kontrol pada kontroler MPC yang meminimalkan indeks performansi J tertulis pada Persamaan 9.

$$\Delta U = (\Phi^T \Phi - \overline{R})^{-1} (\Phi^T \overline{R}_s - \Phi^T Fx(k_i)) \tag{9}$$

ΔU merupakan vektor yang mengandung sinyal kontrol $\Delta u(k_i),\ \Delta u(k_i + 1), \ldots \quad + N_c - 1)$. Dikarenakan prinsip dati *Receding Horizon Control* (RHC), kita hanya dapat mengimplementasikan sampel pertama dari urutan atau *sequence* tersebut, contoh $\Delta u(k_i)$, dan mengabaikan nilai atau urutan selanjutnya seperti tertulis pada Persamaan 10.

$$\begin{aligned} \Delta u(k_i) &= \overbrace{\begin{bmatrix} 1 & 0 & \ldots \end{bmatrix}}^{N} {}^{T} \Phi - \overline{R})^{-1} (\Phi^T \overline{R}_s - \Phi^T Fx(k_i)) \\ &= K_y r(k_i) - K_{MPC} x(k_i) \end{aligned} \tag{10}$$

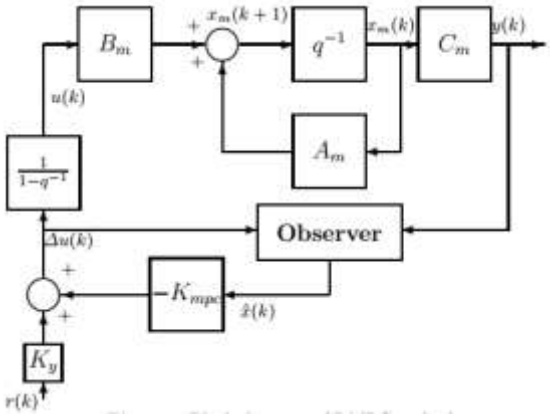
Gain MPC K_{MPC} merupakan baris pertama dari $(\Phi^T \Phi + \overline{R})^{-1} (\Phi^T F)$. Sedangkan K_y adalah elemen pertama dari $(\Phi^T \Phi + \overline{R})^{-1} (\Phi^T \overline{R}_s)$.

Untuk mencari persamaan sistem *closed-loop* pada kontroler MPC seperti yang terlihat pada Gambar 4, digunakan *augmented model* yang tertulis pada Persamaan 11.

$$x(k+1) = Ax(k) + B\Delta u(k) \tag{11}$$

Sistem *closed-loop* dapat dicari dengan mensubtitusi Persamaan 10 kedalam Persamaan 11 dan mengganti indeks k_i ke dalam k sebagaimana Persamaan 12.

$$\begin{aligned} x(k+1) &= Ax(k) - BK_{MPC} x(k) + BK_y r(k) \\ &= (A - BK_{MPC}) x(k) + BK_y r(k) \end{aligned} \tag{12}$$



Gambar 4. Diagram Blok Sistem Kontrol *Loop* Tertutup *Model Predictive Control* untuk Waktu Diskrit dengan menggunakan *Observer*.

III. IDENTIFIKASI DAN PERANCANGAN KONTROLER

A. Metode Pembebanan Plant

Pada Tugas Akhir kali ini, motor BLDC diberikan beban berupa rem elektromagnetik. Beban ini menggambarkan beban awal pada kendaraan elektrik yang ditenagai oleh motor BLDC. Pembebanan dilakukan dengan 3 nilai yang berbeda, yaitu beban minimal, nominal dan maksimal. Potensiometer digunakan untuk mengatur pemberian *input* PWM ke *driver* rem elektromagnetik melalui pin A0. Skala yang digunakan mulai dari nol untuk menon-aktifkan rem hingga skala 1000 untuk pemberian *input* maksimal pada rem. Nilai tegangan yang masuk ke dalam rem dapat dilihat pada Tabel 1.

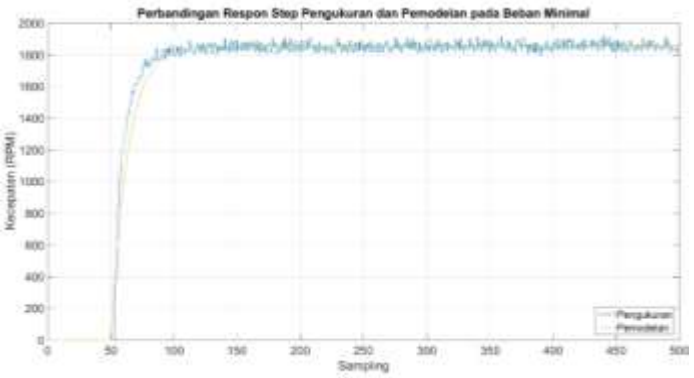
Tabel 1. Metode Pembebanan pada Sistem

Metode Pembebanan	*Input* PWM	Nilai
Minimal	400	4,4 Volt
Nominal	600	7,89 Volt
Maksimal	800	12,56 Volt

B. Identifikasi Motor BLDC

Setelah menentukan pembebanan, langkah selanjutnya adalah melakukan identifikasi dinamis pada motor BLDC yang sudah diberi nilai pembebanan. Identifikasi dinamis dilakukan dengan memberikan sinyal PRBS pada sistem. Adapun *input* yang diberikan berupa nilai pulsa PWM yang dimasukkan ke dalam ESC. Pengambilan data dilakukan dengan membaca nilai sensor *rotary encoder* yang telah dihubungkan ke dalam Arduino untuk membaca nilai *output* kecepatan dari motor BLDC.

Setelah respon berhasil didapatkan, respon tersebut lalu dimodelan dengan pendekatan ARX pada orde 1 sampai dengan 3 menggunakan bantuan *System Identification Toolbox* pada *software* MATLAB. Lalu, model yang digunakan untuk analisa kontroler adalah model dengan nilai *error* yang paling kecil. Gambar perbandingan respon pengukuran dan pemodelan sistem dapat dilihat pada Gambar 5.



Gambar 5. Perbandingan Respon Hasil Pengukuran dan Pemodelan pada Beban Minimal.

C. Pengujian dan Validasi Model

Setelah dilakukan pemodelan menggunakan pendekatan ARX, langkah selanjutnya adalah memvalidasi fungsi alih tersebut. Semakin baik suatu pemodelan, maka respon yang dihasilkan akan semakin menyerupai respon aslinya. Keakuratan suatu pemodelan dapat dilihat dari nilai *error*-nya, dalam hal ini dikalkulasi menggunakan metode *Root Mean Square Error* (RMSE). Semakin kecil nilai RMSE, semakin baik pula pemodelan yang dihasilkan.

Berdasarkan hasil perhitungan identifikasi dinamis untuk tiap pembebanan, didapatkan pemodelan motor BLDC sebagaimana Tabel 2.

Tabel 2. Pemodelan Motor BLDC pada Berbagai Macam Kondisi Pembebanan

Pembebanan	Fungsi alih	RMSE (%)
Minimal	$\frac{0,002513z+0,004715}{z^2-0,9957z-0,003521}$	6,98
Nominal	$\frac{0,003239z+0,004284}{z^2-0,9956z-0,003566}$	5,68
Maksimal	$\frac{0,002823z+0,006564}{z^2-0,9936z-0,005205}$	5,43

D. Perancangan Kontroler MPC

Kontroler MPC merupakan kontroler berbasis model. Artinya, diperlukan sebuah pemodelan fungsi alih yang baik agar kontroler yang telah didesain dapat bekerja secara optimal. Fungsi alih yang digunakan merupakan fungsi alih motor BLDC pada pembebanan maksimal. Fungsi alih ini tidak bisa langsung digunakan untuk merancang kontroler MPC, melainkan harus diubah terlebih dahulu ke dalam bentuk *state-space*. Pada pembebanan maksimal, representasi fungsi alih yang bernilai RMSE paling kecil adalah sebagai berikut:

$$\frac{Y(z)}{U(z)}=\frac{b_0z^2+b_1z+b_2}{z^2+a_1z+a_2}=\frac{0,002823z+0,006564}{z^2-0,9936z-0,005205} \qquad (13)$$

Bentuk *state-space* yang digunakan pada Tugas Akhir ini adalah bentuk *controllable canonical form* [4]. Bentuk *state-space* untuk orde 2 tersebut mempunyai struktur sebagai berikut:

$$\begin{aligned}\begin{bmatrix}x_1(k+1)\\x_2(k+1)\end{bmatrix} &= \begin{bmatrix}0 & 1\\-a_2 & -a_1\end{bmatrix}\begin{bmatrix}x_1(k)\\x_2(k)\end{bmatrix}+\begin{bmatrix}0\\1\end{bmatrix}u(k)\\ y(k) &= \begin{bmatrix}b_2-a_2b_0 & b_1-a_1b_0\end{bmatrix}\begin{bmatrix}x_1(k)\\x_2(k)\end{bmatrix}+b_0u(k)\end{aligned} \qquad (14)$$

Berdasarkan bentuk diatas, maka model *state-space* dari Persamaan 3.1 dapat kita simpulkan sebagai berikut:

$$\begin{aligned}\begin{bmatrix}x_1(k+1)\\x_2(k+1)\end{bmatrix} &= \begin{bmatrix}0 & 1\\0,005205 & 0,9936\end{bmatrix}\begin{bmatrix}x_1(k)\\x_2(k)\end{bmatrix}+\begin{bmatrix}0\\1\end{bmatrix}u(k)\\ y(k) &= \begin{bmatrix}0,006564 & 0,002823\end{bmatrix}\begin{bmatrix}x_1(k)\\x_2(k)\end{bmatrix}\end{aligned}$$

Setelah mendapatkan nilai *state-space* dari sistem, langkah selanjutnya adalah mengubah *state-space* tersebut ke dalam bentuk *augmented model*. Berdasarkan Persamaan (3), dapat disimpulkan bahwa *augmented model* dari sistem adalah sebagai berikut:

$$\begin{aligned}\begin{bmatrix}\Delta x_m(k+1)\\y(k+1)\end{bmatrix} &= \begin{bmatrix}0,005205 & 0,9936 & 0\\0 & 0,0094 & 1\end{bmatrix}\begin{bmatrix}\Delta x_m(k)\\y(k)\end{bmatrix}+\begin{bmatrix}0\\1\\0,002823\end{bmatrix}\Delta u(k)\\ y(k) &= \begin{bmatrix}0 & 0 & 1\end{bmatrix}\begin{bmatrix}\Delta x_m(k+1)\\y(k+1)\end{bmatrix}\end{aligned}$$

Matriks *A,B,C* yang merupakan bentuk *augmented model* akan digunakan selanjutnya dalam merancang sebuah kontroler MPC.

Langkah selanjutnya dalam perancangan kontroler MPC adalah menentukan parameter dari kontroler MPC. Parameter yang dimaksud adalah *prediction horizon* (N_p), *control horizon* (N_c) dan *tuning parameter* pada indeks performansi (r_w). Pada perancangan kali ini, kita akan menggunakan parameter *prediction horizon* sebesar 5 langkah, *control horizon* senilai 2 langkah dan *tuning* parameter indeks performansi sebesar 1. Berdasarkan parameter diatas dan Persamaan 5, dapat ditentukan matriks F dan Φ sebagai berikut:

$$F = \begin{bmatrix} 0 & 0,0094 & 1 \\ 0,0001 & 0,0187 & 1 \\ 0,0001 & 0,0280 & 1 \\ 0,0002 & 0,0373 & 1 \\ 0,0002 & 0,0466 & 1 \end{bmatrix}; \quad \Phi = \begin{bmatrix} 0,0028 & 0 \\ 0,0122 & 0,0028 \\ 0,0215 & 0,0122 \\ 0,0308 & 0,0215 \\ 0,0401 & 0,0308 \end{bmatrix}$$

Setelah mendapatkan matriks F dan Φ dapat, parameter selanjutnya yang akan dicari adalah gain dari kontroler MPC. *Gain* tersebut adalah K_{MPC} dan K_y. Untuk mencari nilai K_{MPC}, terlebih dahulu kita harus mencari nilai dari matriks $(\Phi^T\Phi + \overline{R})^{-1}(\Phi^T F)$.

$$Y = (\Phi^T\Phi + \overline{R})^{-1}(\Phi^T F) \tag{15}$$

$$Y = \begin{bmatrix} 0 & 0,0039 & 0,107 \\ 0 & 0,0026 & 0,0607 \end{bmatrix}$$

Nilai *gain* K_{MPC} merupakan baris pertama dari matriks Y. Oleh karena itu, nilai *gain* K_{MPC} dapat kita simpulkan sebagai matriks berikut ini:

$$K_{MPC} = \begin{bmatrix} 0 & 0,0039 & 0,107 \end{bmatrix}$$

Setelah *gain* K_{MPC} ditemukan, langkah selanjutnya adalah mencari *gain* K_y. Penguatan atau *gain* ini dapat ditemukan dari nilai matriks $(\Phi^T\Phi + \overline{R})^{-1}(\Phi^T \overline{R}_s)$.

$$Z = (\Phi^T\Phi + \overline{R})^{-1}(\Phi^T \overline{R_s}) \tag{16}$$

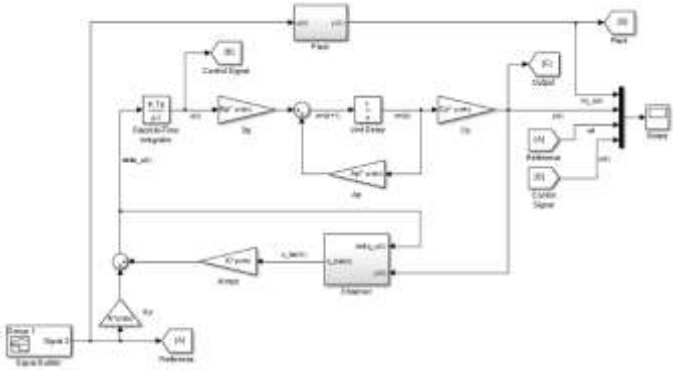
$$Z = \begin{bmatrix} 0,107 \\ 0,607 \end{bmatrix}$$

Gain K_y merupakan baris pertama dari matriks Z. Berdasarkan matriks diatas, dapat kita ambil bahwa nilai K_y mempunyai nilai:

$$K_y = 0,107$$

IV. SIMULASI, IMPLEMENTASI DAN ANALISA

Tahap simulasi digunakan untuk mengetahui performa dari kontroler MPC dalam meminimalkan nilai *error* antara respon dengan referensi. Tahap simulasi penting dilakukan sebelum tahap implementasi untuk memastikan bahwa perancangan kontroler sudah memenuhi performa yang diharapkan. Blok diagram simulasi ditunjukkan seperti Gambar 6. Secara garis besar, diagram blok simulasi sistem terdiri dari *state-space* sistem, penguatan atau *gain* MPC dan subsistem *observer*. Blok diagram dari simulasi sistem dapat dilihat pada Gambar 6.

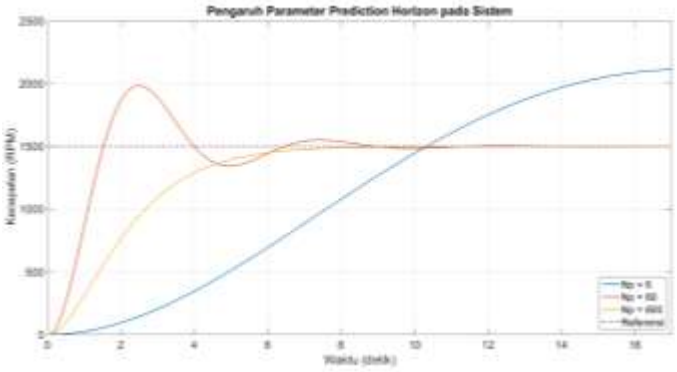


Gambar 6. Diagram Blok Simulasi Pengujian Kontroler MPC pada Sistem.

Variabel pertama yang akan diuji pada sistem dan kontroler MPC adalah pengaruh niali *prediction horizon* atau N_p. Terlihat dari Gambar 7, bahwa respon yang dihasilkan oleh sistem akan berbeda-beda seiring dengan parameter *prediction horizon* yang berbeda pula.

Pada nilai N_p = 60, terlihat bahwa respon sudah mengikuti *tracking setpoint* yang diberikan. Tetapi, seperti terlihat pada Gambar 7, *overshoot* yang dihasilkan oleh sistem sangat tinggi. Tentu saja respon seperti ini sangat dihindari karena akan merusak motor. Oleh karena itu, perlu di-*tuning* lagi nilai dari *prediction horizon* agar respon yang dihasilkan tidak mengalami *overshoot*.

Setelah melakukan beberapa kali pengujian, akhirnya ditemukan nilai N_p = 600. Respon yang dihasilkan jauh lebih baik dibandingkan nilai N_p = 60. Respon yang dihasilkan tidak mengalami *overshoot* sama sekali. Akan tetapi, terlihat bahwa respon yang dihasilkan lebih lambat daripada nilai N_p = 60.

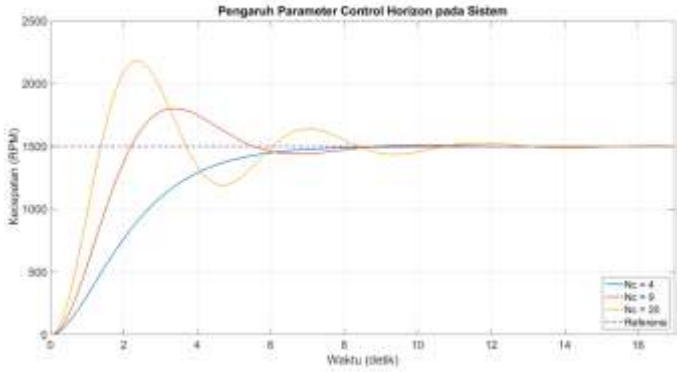


Gambar 7. Pengaruh Parameter *Prediction Horizon* pada Sistem.

Setelah menentukan pengaruh dari *prediction horizon* pada srespon sistem, variabel selanjutnya yang akan diuji adalah pengaruh dari *control horizon* atau N_c pada respon sistem. Gambar 8 menunjukkan pengaruh dari perubahan nilai *control horizon* pada respon sistem.

Terlihat pada gambar bahwa semua nilai variabel *control horizon* mengikuti dari *tracking setpoint*. Hanya saja, terlihat bahwa semakin kecil nilai dari *control horizon*, maka respon akan semakin lambat dan respon yang dihasilkan mirip seperti orde 1. Sebaliknya, semakin besar nilai *control horizon*, maka respon dari sistem akan semakin cepat. Akan tetapi, nilai

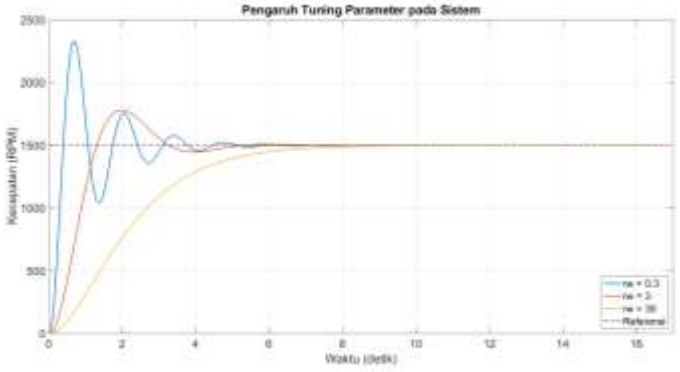
control horizon yang semakin tinggi akan mengakibatkan semakin tinggi pula *overshoot* yang dihasilkan.



Gambar 8. Pengaruh Parameter *Control Horizon* pada Sistem.

Variabel terakhir yang diuji pengaruhnya pada sistem adalah nilai *tuning parameter* r_w pada indeks performansi *J*. Gambar 9 menunjukkan pengaruh dari perubahan nilai *tuning parameter* pada respon sistem.

Terlihat pada Gambar 9, bahwa semua nilai variabel *tuning parameter* mengikuti dari *tracking setpoint*. Terlihat bahwa semakin kecil nilai r_w, semakin cepat respon yang didapatkan. Sebaliknya, semakin besar nilai r_w, maka respon dari sistem akan semakin lambat. Akan tetapi, nilai *tuning parameter* yang semakin kecil akan mengakibatkan meningkatnya *overshoot* yang dihasilkan. Performa kontroler pada referensi *tracking*, dapat dilihat pada Gambar 10.



Gambar 9. Pengaruh Parameter *Tuning* Indeks Performansi pada Sistem.

Gambar 10. Respon Sistem pada Pembebanan Minimal dengan Referensi *Tracking*.

Pada tahap implementasi, seperti terlihat pada Gambar 11, respon yang diberikan oleh sistem dapat mengikuti referensi *setpoint* yang diberikan walaupun sistem, dalam hal ini motor BLDC, sedang diberikan pembebanan. Akan tetapi, jika lebih diteliti secara seksama, terdapat *error steady-state* saat referensi berada pada kecepatan 2500 RPM. Respon yang diberikan juga terlihat tidak begitu halus mengikuti referensi *setpoint* yang diberikan. Ini dikarenakan beberapa faktor yang mempengaruhi sistem. Faktor pertama ada pada sensor *rotary encoder* yang menerima banyak *noise* dari lingkungan sekitar. Faktor kedua adalah pengestimasian nilai *state* lain yang tidak diketahui yang dilakukan oleh *observer* tidak dapat ditentukan secara akurat. Faktor terakhir adalah *data loss* yang terjadi pada komunikasi *serial*.

Tabel 3 menyajikan data perihal karakteristik respon di semua pembebanan pada tahap simulasi dan implementasi.

Tabel 3. Perbandingan Karakteristik Respon pada Tahap Simulasi dan Implementasi

Beban	$\%e_{ss}$		t_r (detik)		t_s (detik)		$\%M_p$	
	S*	I*	S	I	S	I	S	I
Min	0	10	7,66	0,78	10,3	1,08	0	0
Nom	0	7,4	5,70	0,69	7,74	1,27	0	0
Maks	0	12	5,01	0,94	6,65	0,98	0	0

*) S : simulasi
I : implementasi

Gambar 11. Hasil Implementasi Kontroler MPC pada Sistem dengan Pembebanan Minimal.

V. KESIMPULAN

Dari hasil simulasi dan implementasi, terlihat bahwa kecepatan motor BLDC dapat mengikuti referensi *tracking* pada semua parameter pembebanan setelah diberi kontroler MPC. Semakin besar nilai *prediction horizon* dan *tuning parameter*, respon yang dihasilkan akan semakin lambat dan halus. Sebaliknya, semakin kecil nilai *prediciton horizon* dan *tuning parameter* akan mengakibatkan respon yang semakin cepat walaupun dapat menimbulkan *overshoot* yang semakin tinggi. Jika nilai *control horizon* dan *tuning parameter* semakin besar, maka sistem akan memiliki waktu respon yang lebih cepat dengan kekurangan *overshoot* yang semakin tinggi. Adapun jika nilai *control horizon* dan *tuning parameter* semakin kecil, respon yang dihasilkan sistem akan semakin halus walaupun waktu respon akan semakin lambat.

VI. UCAPAN TERIMA KASIH

Ucapan terima kasih sebesar-besarnya ditujukan kepada Beny Setyiadi Hidayat, Habib Ibnu Hasan, Hudaibiy Hibban dan Muhamaad Ammar Huwaidi sebagai Team BLDC V-1 yang telah merancang dan mewujudkan *plant* BLDC V-1 yang digunakan pada *proceeding* ini.

DAFTAR PUSTAKA

[1] Wang, L., "*Model Predicitive Control System Design and Implementation Using MATLAB*", Springer, London, 2009
[2] Söderström, T. and Stoica, P., "*System Identification*", Prentice Hall, United Kingdom, 2001
[3] Seborg, D.E., Mellichamp, D.A., Edgar, T.F., dan Doyle, F.J., "*Process Dynamic and Control*", Wiley, London, 2011
[4] Ogata, Katsuhiko., "*Discrete-Time Control System*", Prentice Hall, New Jersey, 1994