ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, & Teknik Elektronika

ISSN(p): 2338-8323 | ISSN(e): 2459-9638 | Vol. 10 | No. 1 | Halaman 228 - 242
DOI : http://dx.doi.org/10.26760/elkomika.v10i1.228 Januari 2022

Sintesis Model *State-Space* ke *Embedded System* dengan Metode *Backward Difference*

FERIYONIKA

Jurusan Teknik Elektro, Politeknik Negeri Bandung, Indonesia
Email : feriyonika@gmail.com

Received 31 Juli 2021 **|** *Revised* 23 Agustus 2021 **|** *Accepted* 4 Oktober 2021

ABSTRAK

Realisasi sistem kendali berbasis state-space (seperti state feedback, LQR, MPC, SMC, dll) pada embedded system memerlukan informasi state hanya berdasarkan pembacaan output plant. Penelitian ini bertujuan mendapatkan nilai-nilai state suatu plant dengan menganalisis prosedur mengubah persamaan state-space ke persamaan difference (suatu bentuk persamaan yang dapat direalisasikan ke embedded system) menggunakan metode backward difference. Plant yang digunakan adalah kendali kecepatan yang modelnya didapat dari proses identifikasi. State observer ditambahkan pada model state-space plant untuk mengoreksi kesalahan dalam proses identifikasi. Verifikasi persamaan difference dilakukan dengan membandingkan output model terhadap sinyal plant, dimana didapat nilai RMSE-nya sebesar 0,94. Nilai state yang didapat selanjutnya diujicobakan pada kendali state feedback dengan risetime 3,23 detik, settling time 4,82 detik, overshoot 3,1 %, dan error steady state = 0. Berdasarkan hasil pengujian dapat disimpulkan bahwa metode backward difference dapat digunakan untuk mendapatkan nilai-nilai state plant sehingga berbagai algoritma kendali berbasis nilai state bisa direalisasikan pada embedded system.

***Kata kunci**: state-space, backward difference, plant, modern control system, state feedback*

ABSTRACT

Realization of state-space-based control systems (such as state feedback, LQR, MPC, SMC, etc.) in embedded systems requires state information only based on plant output readings.This study aims to obtain states of a plant by using the backward difference method. The plant used is a speed control whose model is obtained from the identification process. State observer is also added to correct the derived model. The verification shows that the obtained states can result output model with RMSE value is 0.94. The states are then tested on the feedback state control with a risetime of 3.23 seconds, a settling time of 4.82 seconds, an overshoot of 3.1%, and a steady state error = 0. This method can be used to obtain plant state values so that various state value-based control algorithms can be realized at the embedded system.

***Keywords**: state-space, backward difference, plant, modern control system, state feedback.*

1. PENDAHULUAN

Pada analisis sistem kendali, dinamika suatu *plant* direpresentasikan dengan persamaan *differential* yang solusinya didapat menggunakan analisis dalam domain Laplace dalam bentuk *transfer function*. Selain dengan cara tersebut digunakan juga teknik *state-space* dengan memecah orde dari persamaan *differential* menjadi per-satuan orde yg disebut *state* (misal persamaan orde 3 maka akan ada tiga *state*) dan dinyatakan dalam bentuk matrik **(Dorf & Bishop, 2010)**. Model berbasis *state-space* dapat mengatasi masalah perubahan parameter pada model *transfer function*. Dengan metode ini nilai *state* selalu di-*update* sehingga kesalahan model karena nilai parameter yang berubah bisa teratasi. Beberapa contoh algoritma kendali yang menggunakan basis *transfer function* adalah *root locus*, *lead-lag compensator*, dan *Proportional Integral Derivative* (PID). Sedangkan algoritma kendali yang berbasis *state-space* diantaranya *state feedback* **(Zanma, dkk, 2020)(Sun, dkk, 2020)**, Linear Quadratic Regulator **(Li, dkk, 2021)(Rizvi & Lin, 2020)**, Linear Quadratic Gaussian **(Nguyen & Chen, 2020)**, MPC **(Nicolis, dkk, 2020),** Sliding Mode Control **(Kaci, dkk, 2019),** dan lainnya. Selain itu, penggunaan domain *state-space* juga didasarkan jenis struktur *plant* yang dikendalikan. Metode ini biasanya digunakan untuk *plant* yang struktur nya berupa *Multi Input Multi Output* (MIMO) **(Anjali, dkk, 2016)(Soeroso, dkk, 2018)**.

Solusi algoritma kendali berbasis *state-space* umumnya direalisasikan ke dalam *software toolbox* Simulink **(Hernando, dkk, 2019)(Fahmizal, dkk, 2019)(Winarno & Endryansyah, 2020)**. Hal ini dikarenakan pada Simulink sudah tersedia blok integrator dan aritmatika lainnya sehingga proses penjabaran *state* dari persamaan *differential* dapat disimulasikan lebih mudah. Selain itu, keadaan mula (*initial condition*) dari *state*-nya dapat disimulasikan pada blok integrator sehingga pengujian algoritma kendali berbasis *state-space* tidak akan terlalu banyak mendapatkan kendala.

Pada saat ini, sintesis algoritma kendali banyak yang direalisasikan ke mikrokontroller yang berukuran lebih kecil dan praktis. Sudah ada beberapa penelitian yang menggunakan algoritma yang berbasis *state-space* ke dalam *plant* (mikrokontroller) **(Dorrah, dkk, 2018)**, **(KL Cezar, dkk, 2020)**. Pada penelitian-penelitian tersebut informasi terkait prosedur dalam mensintesis persamaan *state-space* ke persamaan *difference* belum dijelaskan secara detail. Oleh karena itu, kekosongan informasi tersebut akan menjadi kontribusi utama dari penelitian ini.

Penelitian ini menggunakan metode *backward difference* dalam mensintesis persamaan *state-space* ke persamaan *difference*. Model *plant* yang digunakan adalah kecepatan motor DC yang model-nya didapat dengan teknik sistem identifikasi. Pada tahap awal, model *plant* akan diverifikasi terhadap keluaran *plant* menggunakan blok simulink. Penambahan *state observer* dilakukan untuk mengkoreksi eror pada model *plant*. Selanjutnya proses sintesis dilakukan terhadap model *state-space plant* dan direalisasikan ke mikrokontroller arduino. Untuk memastikan *state* yang dihasilkan dapat diaplikasikan, *state-state* tersebut diverifikasi pada kendali *state feedback*.

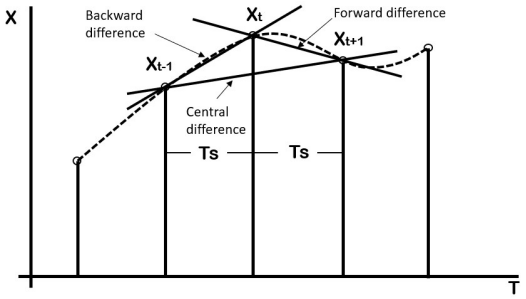
2. METODE

Penelitian ini menggunakan prinsip dasar solusi persamaan *differential* secara *finite difference method*. Hasil sintesis direalisasikan pada mikrokontroller arduino dengan waktu *sampling* yang diukur menggunakan fungsi program. Detail poin-poin diatas dijelaskan pada sub-bagian berikut.

2.1 Sintesis dengan *Backward Difference*

Solusi persamaan *differential* dapat dilakukan dengan pendekatan numerik menggunakan *Finite Difference Method*. Gambar 1 merupakan ilustrasi dari metode **ini (Cho, 2008), (Haniyah & Hamdin, 2020).** Metode ini memiliki tiga pendekatan, antara lain *forward difference, backward difference,* dan *central difference*, masing-masing diformulasikan pada Persamaan (1) s.d (3).

Untuk menjelaskan proses sintesis *state-space* pada *plant*, ilustrasi sinyal pembacaan dari *embedded system* ditunjukkan seperti pada Gambar 2 dan 3. Pada proses *plant,* sinyal di akuisisi secara *real-time* sehingga sinyal terakhir yang tersedia adalah sinyal saat ini (X_t) dan sinyal-sinyal sebelumnya (X_{t-1} , X_{t-2}, X_{t-3} ...). Pada sistem *real-time* sinyal dimasa akan datang (X_{t+1}) sifatnya masih estimasi atau prediksi. Oleh karena itu, berdasarkan sifat pembacan diatas, metode cocok untuk sintesis *state-space* ke *embedded system* adalah *backward difference.*

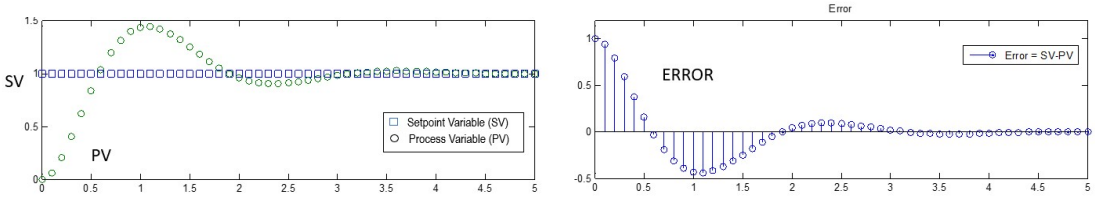


Gambar 1. Solusi Persamaan *Differential* dengan *Finite Difference Method*

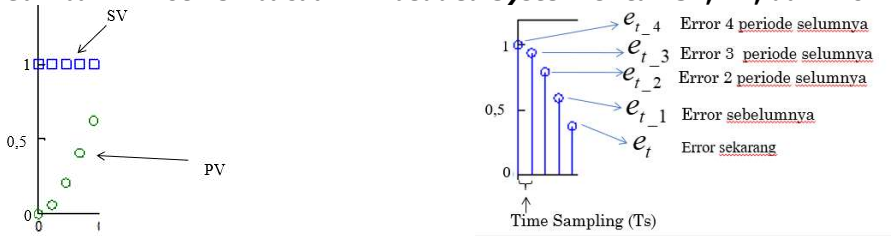
$$\frac{dX_t}{dt} \approx \frac{X_{t+1}-X_t}{T_s} \quad (1)$$

$$\frac{dX_t}{dt} \approx \frac{X_t-X_{t-1}}{T_s} \quad (2)$$

$$\frac{dX_t}{dt} \approx \frac{X_{t+1}-X_{t-1}}{2T_s} \quad (3)$$



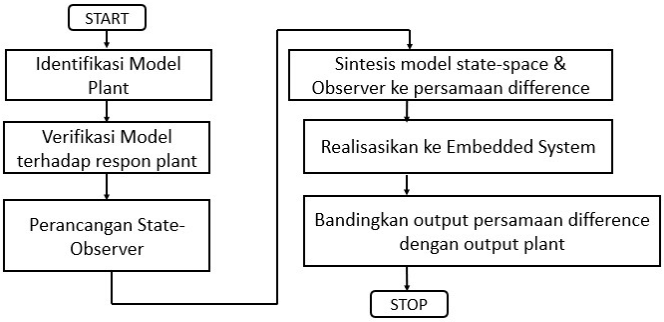
Gambar 2. Plot Pembacaan *Embedded System* Untuk SV, PV, dan Eror



Gambar 3. Ilustrasi Pembacaan SV, PV, dan Eror Secara *Realtime* pada *Embedded System*

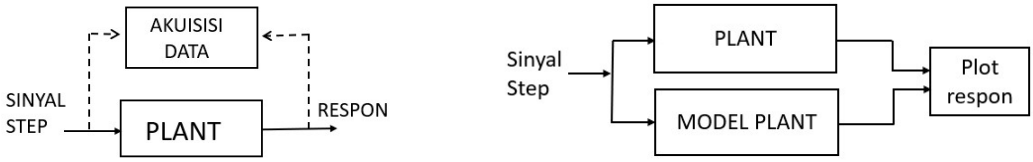
2.2 Perancangan Eksperimen

Pada penelitian ini *plant* yang digunakan adalah kecepatan motor DC dengan spesifikasi tegangan *input* 3 s.d 5,8 V. Umpan balik kecepatan menggunakan tacho-generator, dengan tipe motor DC yang sama, dengan tegangan yg dihasilkan 0 s.d 3,3 V (1 volt =100 rpm). *Embedded system* yang digunakan adalah 8-bit mikrokontroller (arduino-uno) dengan pin *input* 10-bit ADC dan pin *output* 8-bit PWM. Gambar 4 merupakan tahapan eksperimen yang akan dilakukan.



Gambar 4. Diagram Alir Pelaksanaan Eksperimen

Pada tahap pertama dan kedua, seperti diilustrasikan pada Gambar 5, proses identifikasi model *plant* dimulai dengan memberikan sinyal *input* berupa nilai PWM. Besarnya sinyal *input* dan kecepatan yang terbaca oleh tacho-generator di akuisi menggunakan mikrokontroller. Kedua data ini selanjutnya akan digunakan sebagai data dalam mengidentifikasi model *plant* menggunakan *toolbox system identification* pada Matlab sehingga didapatkan model *plant* dalam bentuk *transfer function* (dalam domain 's') dan *state-space*. Tahap berikutnya adalah melakukan verifikasi model dengan membandingkan dengan sinyal *plant.*



(a) **(b)**

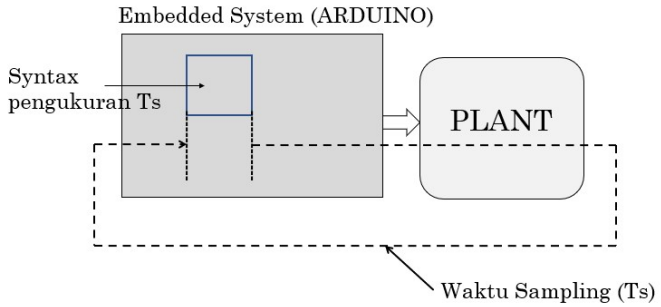
Gambar 5. Identifikasi dan Verifikasi Model, a). Pengambilan Data; b). Verifikasi Model

Dikarenakan model yang dihasilkan dengan metode identifikasi sering terdapat perbedaan dengan sinyal plant, maka tahap berikutnya adalah merancang *state observer* sehingga diperoleh *estimated state* yang menjamin *output* model sama dengan *output plant*. Sampai pada tahap ini, realisasi model *state observer* masih direalisasikan di Simulink. Tahap berikutnya adalah melakukan sintesis persamaan *state-space* dan *observer* menjadi persamaan *difference* (suatu persamaan yang bisa direalisasikan ke *embedded system*). Tahap terakhir adalah mengevaluasi *estimated state* yang dihasilkan oleh persamaan *difference* di *embedded system* dengan membandingkan *output* yang dihasilkan model dengan *output plant*.

2.3 Perhitungan *Sampling Time*

Waktu *sampling* sangat diperlukan dalam proses sintesis model *state-space*. Pada penelitian ini waktu *sampling* pada *embedded system* didefinisikan sebagai waktu yang diperlukan dari setelah sinyal di ukur sampai sesaat sebelum diukur kembali. Gambar 6 merupakan ilustrasi

pengukuran waktu *sampling*. Pada penelitian ini nilai waktu *sampling* tidak diukur terus menerus, nilainya di ukur terlebih dahulu baru kemudian di tuliskan secara manual.



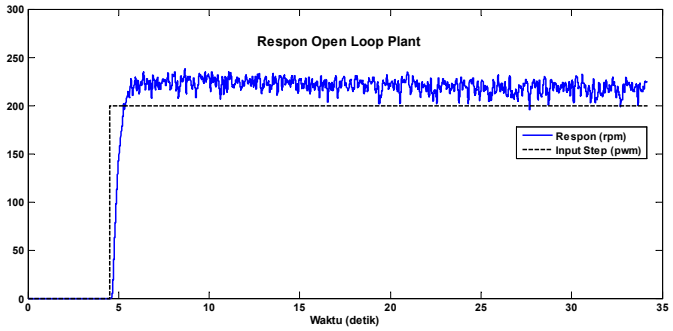
Gambar 6. Deskripsi Pengukuran Waktu *Sampling*

3. HASIL DAN PEMBAHASAN

Model hasil identifikasi *plant* (dalam domain Laplace) terlebih dahulu di ubah ke domain *state-space*. Selanjutnya dilakukan verifikasi model dengan membandingkan terhadap sinyal *plant*. Setelah model *plant* diverifikasi, tahap selanjutnya adalah melakukan prosedur sintesis ke dalam *embedded system*. Penjelasan secara detail dijabarkan pada sub-bab berikut.

3.1 Hasil Identifikasi dan Verifikasi Model *Plant*

Dengan menggunakan prosedur seperti dijelaskan pada Gambar 5, didapatkan respons *plant* Gambar 7. Informasi terkait model *plant* (sistem orde 2 dengan dua pole dan tidak ada zero, data *input-ouput,* dan informasi rentang waktu antar data [0,01 detik]) digunakan sebagai data untuk proses identifikasi model *plant*. Gambar 8 merupakan cuplikan proses identifikasi model *plant* dalam domain laplace (Persamaan (4)). Selanjutnya model ini di konversi ke persamaan *state-space* menggunakan fungsi yang di Matlab (*tf2ss)* sehingga didapatkan Persamaan (5) dan (6). Proses verifikasi model, seperti ilustrasi pada Gambar 5.(b), selanjutnya dilakukan sehingga dihasilkan respons seperti Gambar 9.

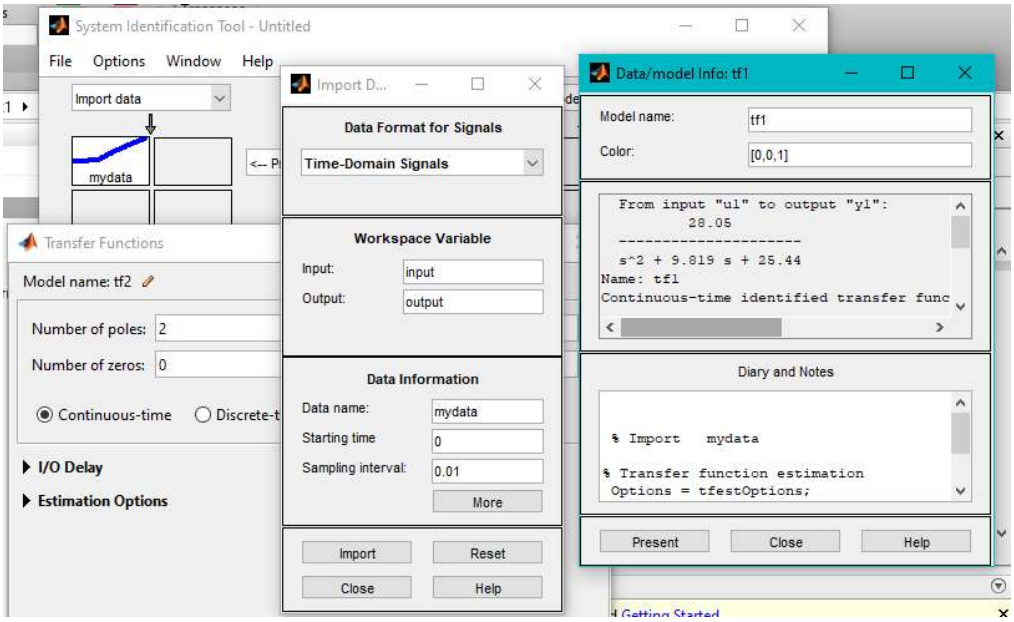


Gambar 7. Respons Plant Saat Diberi *Input Step*

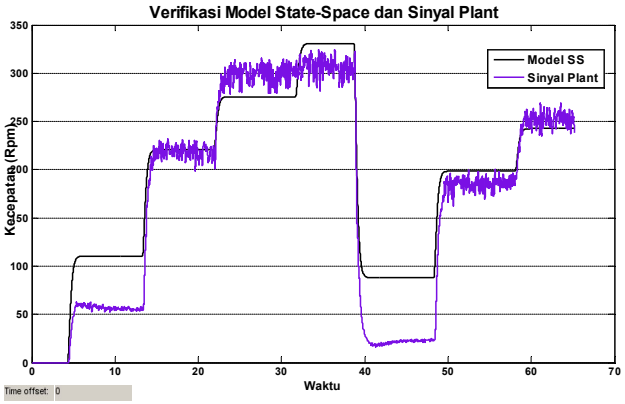
$$G = \frac{28{,}05}{S^2+9{,}819S+25{,}44} \qquad (4)$$

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} = \begin{bmatrix}-9{,}819 & -25{,}44\\ 1 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \begin{bmatrix}1\\ 0\end{bmatrix}u \quad (5)$$

$$y = [0 \quad 28{,}05]\begin{bmatrix}x_1\\ x_2\end{bmatrix} \quad (6)$$



Gambar 8. Proses Identifikasi Model *Plant*



Gambar 9. Verifikasi Sinyal dari Model *State-Space* dan *Plant*

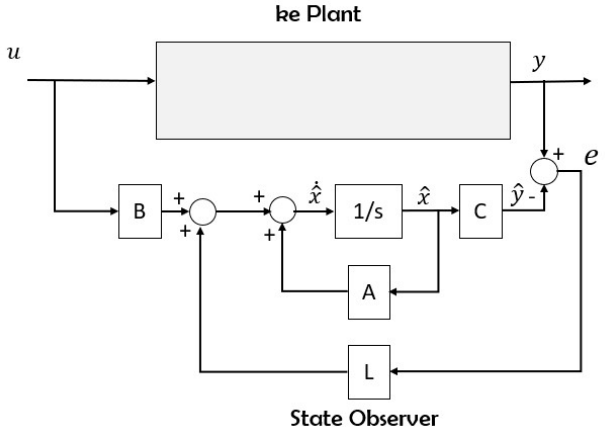
Dari Gambar 9 terlihat bahwa dengan *input* yang sama perbedaan *output* model dan *plant* tidak konsisten. Pada kecepatan tertentu, misalkan 220 rpm, keluaran model dan *plant* sama, akan tetapi terdapat perbedaan yang signifikan pada kecepatan lain. Permasalahan ini biasanya disebabkan oleh proses identifikasi *plant* (data, metode, dll). Untuk mengatasi hal tersebut ditambahkan *state observer* yang akan memastikan model *plant* memiliki keluaran yang sama, yang berarti bahwa *state* yang dihasilkan model mewakili *state plant* sebenarnya.

Tahap pertama penggunaan *state observer* adalah menguji apakah *plant* bersifat *observable*. Berdasarkan Persamaan (7), karena matrik berukuran 2x2 dan *rank* matrik = 2 (*full rank*), maka *plant* bersifat *observable*, yang berarti dengan informasi pembacaan *ouput plant*, maka seluruh *state* sistem dapat diketahui.

$$rank\left(\begin{bmatrix} C \\ CA \end{bmatrix}^T\right) \quad (7)$$

dengan $C = [0 \quad 28{,}05]$; A= $\begin{bmatrix} -9{,}819 & -25{,}44 \\ 1 & 0 \end{bmatrix}$

Realisasi *state observer* ditunjukkan pada Gambar 10. Pada proses ini nilai *state* yang dihasilkan oleh *state observer* disebut *estimated state* (di atas variablenya diberi lambang ˆ , contoh $\hat{x}$), yang di tunjukkan pada Persamaan (8). Perubahan nilai eror *state* dinyatakan berdasarkan Persamaan (9). Dengan mensubstitusikan persamaan *state* dan *estimated state* didapatkan Persamaan (10).



Gambar 10. Blok *State Observer*

$$\dot{\hat{x}} = A\hat{x} + Bu + L\,(y - \hat{y}) \quad (8)$$

$$\dot{e}_x = \dot{x} - \dot{\hat{x}} \quad (9)$$

$\dot{x} - \dot{\hat{x}} = (Ax + Bu) - (A\hat{x} + Bu + L\,(y - \hat{y}))$
$\dot{x} - \dot{\hat{x}} = (Ax + Bu) - (A\hat{x} + Bu + L\,([Cx] - [C\hat{x}]))$
$\dot{x} - \dot{\hat{x}} = Ax - A\hat{x} + LCx - LC\hat{x}$
$\dot{x} - \dot{\hat{x}} = (A - LC)(x - \hat{x})$

$$\dot{e}_x = (A - LC)e_x \quad (10)$$

Berdasarkan Persamaan (10) dapat dilihat bahwa perubahan eror *state* ($\dot{e}_x$) akan menuju ke nol jika *eigen-value* (akar) dari matrik transisi (A-LC) bernilai negatif. Jika diketahui $L = \begin{bmatrix} L_{11} \\ L_{21} \end{bmatrix}$ maka,

$$A - LC = \begin{bmatrix} -9{,}8191 & -25{,}44 \\ 1 & 0 \end{bmatrix} - \begin{bmatrix} L_{11} \\ L_{21} \end{bmatrix} [0 \quad 28{,}05]$$

$$A - LC = \begin{bmatrix} -9{,}8191 & -25{,}44 \\ 1 & 0 \end{bmatrix} - \begin{bmatrix} 0 & 28{,}05L_{11} \\ 0 & 28{,}05L_{21} \end{bmatrix}$$

$$A - LC = \begin{bmatrix} -9{,}8191 & -25{,}44 - 28{,}05L_{11} \\ 1 & -28{,}05L_{21} \end{bmatrix}$$

Untuk mendapatkan eigen-value dari matrik transisi diatas digunakan Persamaan (11). Persamaan (12) merupakan persamaan kuadrat yang akan digunakan sebagai acuan letak akar *state observer*.

$$det\left[\begin{bmatrix}-9{,}8191 & -25{,}44-28{,}05L_{11}\\ 1 & -28{,}05L_{21}\end{bmatrix}-\begin{bmatrix}\lambda & 0\\ 0 & \lambda\end{bmatrix}\right] \quad (11)$$

Dari Persamaan (11) dihitung determinannya sebagai berikut

$\big((-9{,}8191-\lambda)(-28{,}05L_{21}-\lambda)\big)$ - $(-25{,}44-28{,}05L_{11})$

$$\lambda^2+(9{,}8191+28{,}05L_{21})\lambda+(25{,}44+28{,}05L_{11}) \quad (12)$$

Untuk mencari nilai matrik L, terlebih dahulu menentukan letak akar *observer* berdasarkan letak akar *plant*. Aturan umum yang berlaku adalah meletakkan akar *observer* 5 kali lebih ke kiri dari pada akar paling kanan *plant* (pole dominan), secara fisis hal ini dimaksudkan agar proses *observer* lebih cepat dari respons *plant*. Berdasarkan Persamaan (4) diketahui letak akar *plant* adalah

$S_1=-4{,}9095+1{,}156i$; $S_2=-4{,}9095-1{,}156i$;

Letak pole dari *state observer* adalah

$O_1=-24{,}5475+1{,}156i$; $O_2=-24{,}5475-1{,}156i$;

Dari letak akar *state observer* tersebut kita dapatkan Persamaan (13) sebagai persamaan karakteristiknya.

$$s^2+49{,}09s+603{,}9 \quad (13)$$

Dari Persamaan (12) dan (13) maka nilai L dapat dihitung sbb:

$(25{,}44+28{,}05L_{11})=603{,}9$

$L_{11}=\frac{(603{,}9-25{,}44)}{28.05}=20{,}6225$

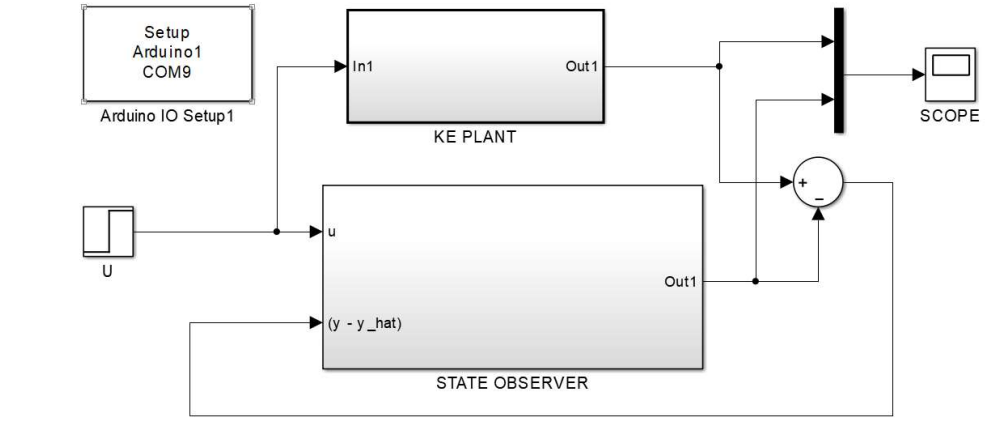
$(9{,}8191+28{,}05L_{21})=49{,}09$

$L_{21}=\frac{(49{,}09-9{,}8191)}{28{,}05}=1{,}4$

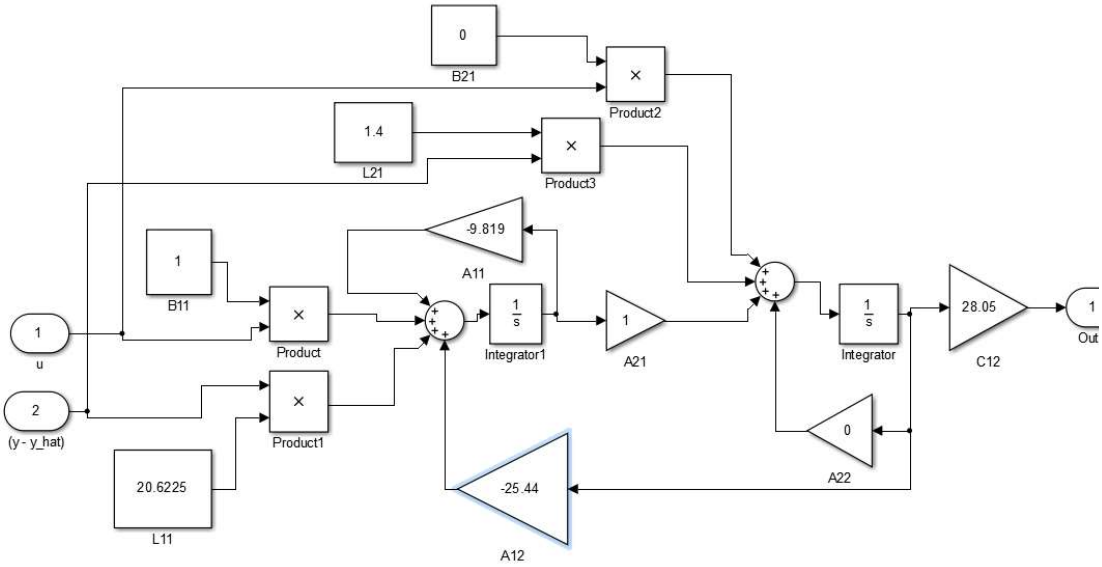
Dengan diketahuinya nilai matrik L, maka persamaan *state observer* menjadi Persamaan (14).

$$\begin{bmatrix}\hat{x}_1\\ \hat{x}_2\end{bmatrix}=\begin{bmatrix}-9{,}819 & -25{,}44\\ 1 & 0\end{bmatrix}\begin{bmatrix}\hat{x}_1\\ \hat{x}_2\end{bmatrix}+\begin{bmatrix}1\\ 0\end{bmatrix}u+\begin{bmatrix}20{,}6225\\ 1{,}4\end{bmatrix}(y-\hat{y}) \quad (14)$$

Gambar 11 merupakan blok realisasi *state observer* dan proses verifikasi dengan sinyal *plant*. Gambar 12 adalah hasil perbandingan kedua sinyal. Terlihat bahwa model *state observer* yang didesain mewakili sifat *plant*. Selanjutnya model inilah yang akan di sintesiskan ke *embedded system*.

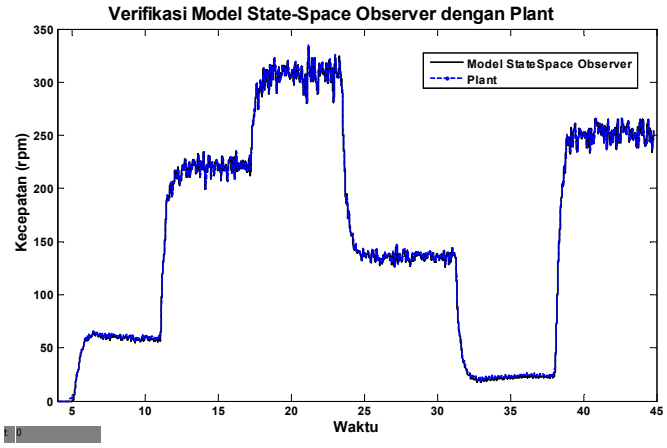


(a)



(b)

Gambar 11. Verifikasi *State Observer*, a). Blok Simulink; b). Sub-sistem *State Observer*



Gambar 12. Sinyal Model *State-Space* dan *Plant*

3.2 Sintesis Model *State-Space* dengan *Backward Difference*

Proses sintesis dimulai dengan terlebih dahulu mengubah Persamaan (14) menjadi Persamaan (15).

$$\begin{bmatrix}\dot{\hat{x}}_{1k}\\ \dot{\hat{x}}_{2k}\end{bmatrix} = \begin{bmatrix}a_{11} & a_{12}\\ a_{21} & a_{22}\end{bmatrix}\begin{bmatrix}\hat{x}_{1k}\\ \hat{x}_{2k}\end{bmatrix} + \begin{bmatrix}b_{11}\\ b_{21}\end{bmatrix}u_t + \begin{bmatrix}L_{11}\\ L_{21}\end{bmatrix}(y_t - \hat{y}_k) \quad (15)$$

Dengan nilai $a_{11} = -9{,}819$, $a_{12} = -25{,}44$, $a_{21} = 1$, $a_{22} = 0$, $b_{11} = 1$, $b_{21} = 0$, $L_{11} = 20{,}6225$, $L_{21} = 1{,}4$.

Persamaan (15) dijabarkan menjadi persamaan masing-masing *state* menjadi Persamaan (16) dan (17).

$$\dot{\hat{x}}_1 = a_{11}\hat{x}_1 + a_{12}\hat{x}_2 + b_{11}\,u_t + L_{11}\,(y_t - \hat{y}_k) \quad (16)$$

$$\dot{\hat{x}}_2 = a_{21}\hat{x}_1 + a_{22}\hat{x}_2 + b_{21}\,u_t + L_{21}\,(y_t - \hat{y}_k) \quad (17)$$

Berdasarkan Persamaan (2), maka Persamaan (16) menjadi Persamaan (18).

$$\frac{\hat{x}_{1k} - \hat{x}_{1k-1}}{T_s} = a_{11}\hat{x}_{1k} + a_{12}\hat{x}_{2k} + b_{11}\,u_t + L_{11}\,(y_t - \hat{y}_k)$$
$$\hat{x}_{1k} - \hat{x}_{1k-1} = T_s a_{11}\hat{x}_{1k} + T_s a_{12}\hat{x}_{2k} + T_s\,b_{11}\,u_t + T_s L_{11}(y_t - \hat{y}_k)$$
$$\hat{x}_{1k} - T_s a_{11}\hat{x}_{1k} = \hat{x}_{1k-1} + T_s a_{12}\hat{x}_{2k} + T_s\,b_{11}\,u_t + T_s L_{11}(y_t - \hat{y}_k)$$
$$(1 - T_s a_{11})\hat{x}_{1k} = \hat{x}_{1k-1} + T_s a_{12}\hat{x}_{2k} + T_s\,b_{11}\,u_t + T_s L_{11}(y_t - \hat{y}_k)$$

$$M_1\hat{x}_{1k} = \hat{x}_{1k-1} + M_2\hat{x}_{2k} + M_3\,u_t + M_4(y_t - \hat{y}_k) \quad (18)$$

Dengan cara yang sama Persamaan (17) dijabarkan menjadi Persamaan (19).

$$\frac{\hat{x}_{2k} - \hat{x}_{2k-1}}{T_s} = a_{21}\hat{x}_{1k} + a_{22}\hat{x}_{2k} + b_{21}\,u_t + L_{21}\,(y_t - \hat{y}_k)$$
$$\hat{x}_{2k} - \hat{x}_{2k-1} = T_s a_{21}\hat{x}_{1k} + T_s a_{22}\hat{x}_{2k} + T_s\,b_{21}\,u_t + T_s L_{21}(y_t - \hat{y}_k)$$
$$\hat{x}_{2k} - T_s a_{22}\hat{x}_{2k} = \hat{x}_{2k-1} + T_s a_{21}\hat{x}_{1k} + T_s\,b_{21}\,u_t + T_s L_{21}(y_t - \hat{y}_k)$$
$$\hat{x}_{2k} - T_s a_{22}\hat{x}_{2k} = \hat{x}_{2k-1} + T_s a_{21}\hat{x}_{1k} + T_s\,b_{21}\,u_t + T_s L_{21}(y_t - \hat{y}_k)$$
$$(1 - T_s a_{22})\hat{x}_{2k} = \hat{x}_{2k-1} + T_s a_{21}\hat{x}_{1k} + T_s\,b_{21}\,u_t + T_s L_{21}(y_t - \hat{y}_k)$$
$$N_1\hat{x}_{2k} = \hat{x}_{2k-1} + N_2\hat{x}_{1k} + N_3\,u_t + N_4(y_t - \hat{y}_k)$$

$$\hat{x}_{2k} = \frac{\hat{x}_{2k-1}}{N_1} + \frac{N_2}{N_1}\hat{x}_{1k} + \frac{N_3}{N_1}\,u_t + \frac{N_4}{N_1}(y_t - \hat{y}_k) \quad (19)$$

Selanjutnya substitusikan Persamaan (19) ke Persamaan (18) sehingga didapat Persamaan (20).

$$M_1\hat{x}_{1k} = \hat{x}_{1k-1} + M_2\left[\frac{\hat{x}_{2k-1}}{N_1} + \frac{N_2}{N_1}\hat{x}_{1k} + \frac{N_3}{N_1}u_t + \frac{N_4}{N_1}(y_t - \hat{y}_k)\right] + M_3\,u_t + M_4(y_t - \hat{y}_k)$$

$$M_1\hat{x}_{1k} = \hat{x}_{1k-1} + \left[\frac{M_2\hat{x}_{2k-1}}{N_1} + \frac{M_2N_2}{N_1}\hat{x}_{1k} + \frac{M_2N_3}{N_1}u_t + \frac{M_2N_4}{N_1}(y_t - \hat{y}_k)\right] + M_3\,u_t + M_4(y_t - \hat{y}_k)$$

$$(M_1 - \frac{M_2N_2}{N_1})\hat{x}_{1k} = \hat{x}_{1k-1} + \frac{M_2}{N_1}\hat{x}_{2k-1} + (M_3 + \frac{M_2N_3}{N_1})u_t + (M_4 + \frac{M_2N_4}{N_1})(y_t - \hat{y}_k)$$
$$M_5\hat{x}_{1k} = \hat{x}_{1k-1} + M_6\hat{x}_{2k-1} + M_7u_t + M_8(y_t - \hat{y}_k)$$

$$\hat{x}_{1k} = \frac{\hat{x}_{1k-1} + M_6\hat{x}_{2k-1} + M_7 u_t + M_8(y_t - \hat{y}_k)}{M_5} \quad (20)$$

dimana diketahui

$$M_1 = 1 - T_s a_{11} \qquad N_1 = 1 - T_s a_{22} \qquad M_5 = M_1 - \frac{M_2 N_2}{N_1}$$

$$M_2 = T_s a_{12} \qquad N_2 = T_s a_{21} \qquad M_6 = \frac{M_2}{N_1}$$

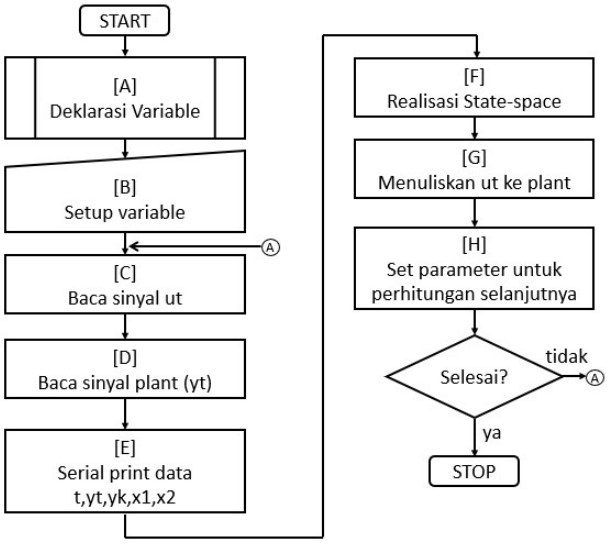
$$M_3 = T_s\, b_{11} \qquad N_3 = T_s\, b_{21} \qquad M_7 = M_3 + \frac{M_2 N_3}{N_1}$$

$$M_4 = T_s L_{11} \qquad N_4 = T_s L_{21} \qquad M_8 = M_4 + \frac{M_2 N_4}{N_1}$$

Persamaan *output* model *state* space nya menjadi Persamaan (21)

$$\hat{y}_k = c_{11}\hat{x}_{1k} + c_{12}\hat{x}_{2k} \quad (21)$$

Persamaan (19) – (21) direalisasikan kepemrograman *embedded system* dengan *flowchart* seperti pada Gambar 13. Contoh bahasa yang digunakan disini adalah bahasa C pada IDE Arduino.



Gambar 13. *Flowchart* Realisasi ke *Embedded system*

Pada bagian '[A]' berisi deklarasi variable yang digunakan, terdapat parameter untuk filter digital IIR orde 1, dengan fc=frekuensi *cut-off* (dalam radian). Jika sinyal *plant* tidak ada *noise* bagian ini bisa diabaikan.

```
// [A]
//1a. Deklarasi perhitungan filter
    float PV, PVf, PVf_1,fc,RC,a,Ts,t;
```

```
//1b. Deklarasi Sintesis State
    float a11, a12, a21, a22, b11, b21, c11, c12, L11, L21;
    float X1k, X1k_1, X2k, X2k_1;
    float M1, M2, M3, M4, M5, M6, M7, M8, N1, N2, N3, N4;
    float yt, yk, ut;
```

Pada bagian '[B]' berisi *setup* parameter yang akan digunakan untuk perhitungan sintesis *state-space* maupun untuk keperluan data akuisisi. Data matrik *state-space*, *observer*, dan *initial condition* juga diletakkan pada bagian ini.

```
void setup() {
//[B]
//2a. Setup untuk sistem
    Serial.begin(9600);
//2b. Setup pin yg dipakai
    pinMode(10, OUTPUT);//sinyal kendali
//2c. Setup parameter Filter
    fc=0.4;//nilai fc lbh ke kanan dr frekuensi noise
    RC=1/(6.28*fc);
    Ts=0.01;//diukur manual terlebih dahulu
    a=RC/Ts;
    PVf_1=0;
//2d. Set parameter matrik A,B,C
    a11=-9.819;
    a12=-25.44;
    a21=1;
    b11=1;
    b21=0;
    c11=0;
    c12=28.05;
    L21=1.4002;
//2e. Set parameter State observer, L
    L11=20.6225;
//2f. Set nilai untuk perhitungan state
    M1=1-(Ts*a11);
    M2=Ts*a12;
    M3=Ts*b11;
    M4=Ts*L11;
    N1=1-(Ts*a22);
    N2=Ts*a21;
    N3=Ts*b21;
    N4=Ts*L21;
    M5=M1-((M2*N2)/N1);
    M6=M2/N1;
    M7=M3+((M2*N3)/N1);
    M8=M4+((M2*N4)/N1);
//2g. Set nilai awal untuk beberapa parameter
    yk=0;
    PVf_1=0;
    X1k=0;//initial condition state X1
    X2k=0;//initial condition state X2
    X1k_1=0;
    X2k_1=0;
    t=0;
}//akhir dari bagian set up
```

Pembacaan sinyal u_t diset pada pin analog 0 (A0). Nilai u_t dipetakan langsung sesuai spesifikasi pwm *output*. Detail bagian '[C]' dapat dilihat pada sintaks berikut.

```
void loop() {
//[C] Baca ut
    ut=analogRead(0);
    ut=map(ut,0,1023,0,255);
```

Kode program pada bagian '[D]' berisi pembacaan *output plant* dan proses filter. Keluaran dari *plant* di set sebagai y_t.

```
//[D] Membaca PV
    PV=analogRead(5)*0.0049*100;
    X2k=(X2k_1 + N2*X1k + N3*ut + N4*(yt - yk))/N1;
    yk=c11*X1k + c12*X2k;
```

Kode program pada bagian '[E]' adalah proses akusisi *output plant* dan waktu. Karena plot di *software* Arduino tidak terlalu detail dan tidak terdapat menu legenda dan informasi axis, maka data dari *embedded system* diambil secara serial lalu di plot menggunakan *software* Matlab script.

```
//[E] Akuisi sinyal
    t=t+Ts;//untuk keperluan plot waktu
    Serial.print(t);
    Serial.print(" ");
    Serial.print(yt);
    Serial.print(" ");
    Serial.print(yk);
    Serial.print(" ");
    Serial.print(X1k);
    Serial.print(" ");
    Serial.println(X2k);
```

Kode program pada bagian '[F]'. Merupakan bagian inti dari proses sintesis dengan algoritma *backward difference*.

```
//[F] Persamaan state
    X1k=(X1k_1+(M6*X2k_1)+(M7*ut)+(M8*(yt - yk)))/M5;
    X2k=(X2k_1 + N2*X1k + N3*ut + N4*(yt - yk))/N1;
    yk=c11*X1k + c12*X2k;
```

Kode program pada bagian '[G]'adalah menuliskan sinyal $u_t$ke *plant*.

```
//[G].Menuliskan ut ke pin output, disini menggunakan pin 10
    analogWrite(10,ut);
```

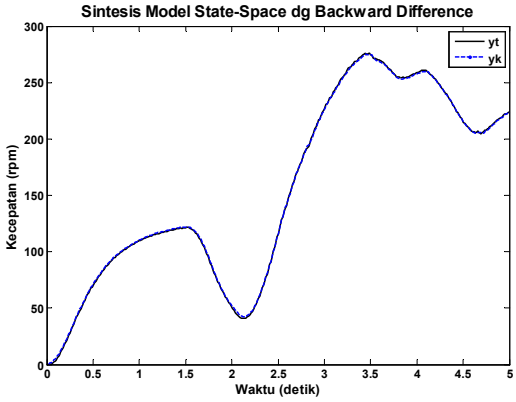
Dengan sifat pemrograman mikrokontroller arduino yang *looping* terus menerus, maka kode program pada bagian '[H]' bertujuan men-set nilai untuk perhitungan di *looping* berikutnya.

```
//[H] Set untuk nilai perhitungan di loop berikutnya
    X1k_1=X1k;
    X2k_1=X2k;
    PVf_1=PVf;
`
```

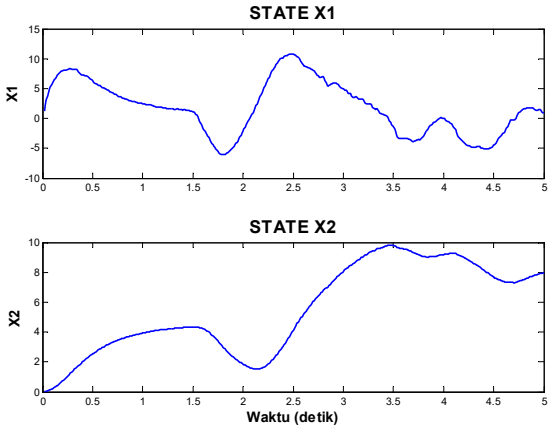
Gambar 14 dan 15 adalah hasil realisasi persamaan *difference* ke dalam program *embedded system*. Evaluasi model menggunakan Persamaan (22) untuk menghitung rata-rata akar kuadrat eror (RMSE).

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_{ti}-y_{ki})^2}{n}} \quad (22)$$

Dengan n = 499 data, nilai RMSE-nya adalah 0,94. Berdasarkan grafik *output* dapat dilihat model bersifat *observable* sehingga dengan menggunakan pembacaan *output plant*, nilai *state* sistem dapat diketahui. Nilai-nilai *state* ini sangat penting untuk menerapkan metode kendali berbasis *state-space* seperti *state feedback*, LQR, LQG, MRAC, SMC, dll.



Gambar 14. Sinyal *Plant* (yt) dan Hasil Sintesis (yk)



Gambar 15. *State* x1 dan x2

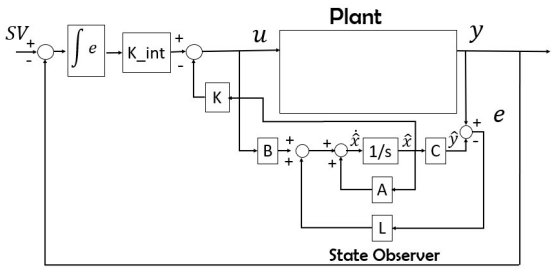
Untuk menguji nilai *state* yang dihasilkan, nilai *state* akan digunakan pada kendali *state feedback* dengan struktur kendali seperti pada Gambar 16. Penentuan nilai gain K tidak dibahas secara detail pada artikel ini, nilai-nilainya didapatkan dengan menggunakan fungsi Matlab setelah ditentukan letak akar yang diinginkan.

```
>> A=[-9.819 -25.44;1 0];
>> B=[1;0];
>> C=[0 28.05];
>> p1=-5+i;
>> p2=-5-i;
```

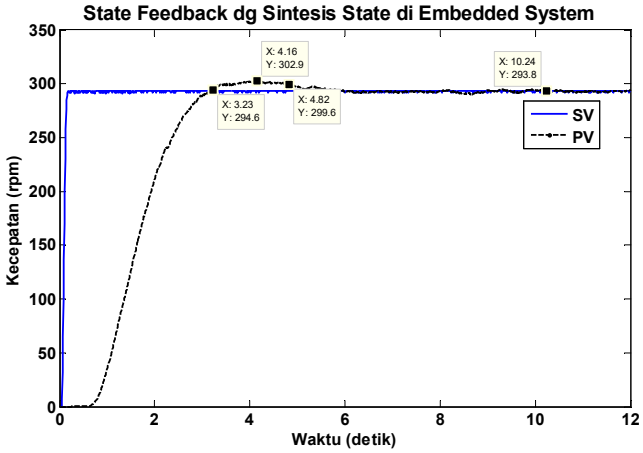
```
>> K=place(A,B,[p1 p2])
K =
   0.1810    0.5600
```

Pada kendali ini besarnya sinyal kendali u_t dihasilkan berdasarkan Persamaan (23).

$$u_t = K_{int}e - K_1\hat{x}_{1k} - K_2\hat{x}_{2k} \quad (23)$$



Gambar 16. Struktur *State Feedback* dengan Integral *Gain*



Gambar 17. Respons Kendali *State Feedback* Menggunakan *State* x1 dan x2

Gambar 17 merupakan respons kendali *state feedback* dengan $K_{int} = 0{,}5$. Dari Gambar 17 dapat diketahui parameter respons nya adalah *risetime*= 3,23 detik; *settling time* (2%) = 4,82 detik, *overshoot* = 3,1 %, *error steady state* = 0. Dari respons tersebut dapat kita simpulkan bahwa *state* yang disintesiskan pada *embedded system* dapat digunakan dalam merealisasikan salah satu metode kendali dalam domain *state-space*.

4. KESIMPULAN

Suatu model sistem yang akan direalisasikan ke *embedded system* harus dirubah terlebih dahulu ke persamaan *difference*. Pada penelitian ini sintesis persamaan *state-space* ke persamaan *difference* dapat dilakukan dengan menggunakan metode *backward difference*. Model *state-space plant* yang di sintesis adalah *plant* kecepatan motor DC yang modelnya didapatkan dari proses identifikasi. Ketidaksempurnaan model hasil identifikasi dapat diatasi dengan menambahkan *state observer* sehingga hanya dengan memanfaatkan *output plant* maka semua *state* sistem dapat diketahui. Nilai *state* hasil sintesis berhasil di uji coba ke kendali *state feedback* dengan respons *plant* adalah *risetime*= 3,23 detik; *settling time* (2%) = 4,82 detik, *overshoot* = 3,1 %, *error steady state* =0.

UCAPAN TERIMAKASIH

Penulis mengucapkan terimakasih kepada P3M Politeknik Negeri Bandung atas skema Penelitian Mandiri Dosen 2021.

DAFTAR RUJUKAN

Anjali, B., Vivek, A., & Nandagopal, J. (2016). Simulation and Analysis of Integral LQR Controller for Inner Control Loop Design of a Fixed Wing Micro Aerial Vehicle (MAV). *Procedia Technology, 25*, 76-83.

Cho, H. (2008). *Introduction of Numerical Analysis: Numerical Differentiation.* Seoul National University, Department of Nuclear Engineering. Seoul: Seoul National University.

Dorf, & Bishop. (2010). *Modern Control System; Chapter 3: State Variable.* Upper Saddle River, New Jersey: Prentice Hall.

Dorrah, H., Gabr, W., & Elsayed, M. (2018). Derivation of symbolic-based embedded feedback control stabilization expressions with experimentation. *Journal of Electrical Systems and Information Technology, 5*(3).

Fahmizal, F., Arrofiq, M., Adrian, R., & Mayub, A. (2019). Robot Inverted Pendulum Beroda Dua (IPBD) dengan Kendali Linear Quadratic Regulator (LQR) . *ELKOMIKA, 7*(2).

Haniyah, A., & Hamdin, S. B. (2020). Finite Difference Method for the Biharmonic Equation with Different Types of Mixed Boundary Conditions. *IOSR Journal of Mathematics, 16*(1), 6-13.

Hernando, G. J., Feriyonika, & Sudarsa, Y. (2019). *Model Predictive Control pada kecepatan Motor DC.* Politeknik Negeri Bandung, Jurusan Teknik Elektro. Bandung: Politeknik Negeri Bandung.

Kaci, A., Giraud-Audine, C., Giraud, F., Amberg, M., & Lemaire-Semail, B. (2019). LQR based MIMO PID controller for the vector control of and underdamped harmonic oscillator. *Mechanical Systems and Signal Processing, 134*.

KL Cezar, A. C., Cordeiro, M., Santor, C. D., Cezar, K., Ochoa, A., & Michima, P. (2020). Development of a novel flow control system with arduino micocontroller embedded in double effect absorption chillers using the LiBr/ H2) pair. *International Journal of Refrigeration, 111*, 124-235.

Li, L., Zhang, H., & Wang, Y. (2021). Stabilization and optimal control of discrete time systems with multiplicative noise and multiple input delays. *Systems & Control Letters, 147*.

Nguyen, M. L., & Chen, X. (2020). MPC Inspired Dynamical Output Feedback and Adaptive Feedforward Control Applied to Piezo-Actuated Positioning System. *IEEE Transactions on Industrial Electronics, 67*(5), 3921 - 3931.

Nicolis, D., Allevi, F., & Rocco, P. (2020). Operational Space Model Predictive Sliding Mode Control for Redundant Manipulators. *IEEE Transactions on Robotics, 36*(4), 1348 - 1355.

Rizvi, S. A., & Lin, Z. (2020). Reinforcement Learning-Based Linear Quadratic Regulation of Continuous-Time Systems Using Dynamic Output Feedback. *IEEE Transaction on Cybernetics, 50*(11), 4670 - 4679.

Soeroso, K. K., Feriyonika, & Sudarsa, Y. (2018). *Kendali LQR pada Kecepatan Motor DC.* Politeknik Negeri Bandung, Jurusan Teknik Elektro. Bandung, Indonesia: Politeknik Negeri Bandung.

Sun, X., Hu, C., Lei, G., Guo, Y., & Zhu, J. (2020, Jan). *State* Feedback Control for a PM Hub Motor Based on Gray Wolf Optimization Algorithm. *IEEE Transaction on Power Electronics, 35*(1), 1136 - 1146.

Winarno, Y., & Endryansyah. (2020.). System kontrol kecepatan Motor DC pada lift Konvensional dengan Kontrol Linear Quadratic Regulator (LQR) Berbasis Arduino Uno. *JURNAL TEKNIK ELEKTRO, 9*(2).

Zanma, T., Ohtsuka, T., & Liu, K. (2020). Set-Based *State* Estimation in Quantizied *State* Feedback Control System with Quantized Measurement. *IEEE Transaction on Control Systems Technology, 28*(2), 550 - 557.