# DESAIN UMPAN BALIK KEADAAN MENGGUNAKAN ALGORITMA *PARTICLE SWARM OPTIMIZATION* DAN *DIFFERENTIAL EVOLUTIONALGORITHM* STUDI KASUS GERAK LATERAL PESAWAT F-16

[1]Madchan Anis, [2]Widowati, [3]R. Heru Tjahjana
[1,2,3] Jurusan Matematika Universitas Diponegoro
Jl. Prof. Soedharto, SH, Tembalang Semarang 54275
Email :[1]anis.madchan@gmail.com, [2]wiwied_mathundip@yahoo.com,
[3]heru_tjahjana@undip.ac.id

**Abstract** : The purpose of *Linear Quadratic Regulator* (LQR) optimal control system is to stabilize the system, so that the output of the system towards a steady state by minimizing the performance index. LQR-*invinite horizon* is a special case of LQR in thecontinuous time area where the terminal time of the performance index value for infinite time and infinite outputsystem is zero. Performance index will be affected by the weighting matrix. In this paper will be discussed about the application of *Particle Swarm Optimization algorithm* (PSO) and *Differential Evolution Algorithm* (DEA) to determine the state feedback of a closed loop system and weighting matrices in the LQR to minimize performance index. PSO algorithm is a computational algorithm inspired by social behavior of flocks of birds and fishes in searching of food. While the DEA is an optimization algorithm that is adopted from evolution and genetics of organisms. Simulations of the PSO algorithm will be compared with DEA. Based on case study, DEA is faster then PSO to get convergence to the optimum solution.

Keywords: LQR-*invinite horizone*, weighting matrix, *Particle Swarm Optimization* (PSO), *Differential Evolution Algorithm* (DEA)

## 1. Pendahuluan

Dalam pendesainan sistem dan penyelesaian masalah di dalamnya, perlu memilih sebuah solusi sebagai respon optimal dari luasnya kumpulan solusi fisibel, semua solusi tidak mungkin untuk diuji satu per satu dan pengujian seharusnya dilakukan secara stokastik. Di sisi lain, proses stokastik harus dilakukan melalui cara yang mendekati solusi terbaik. Teori kontrol optimum linier kuadratik merupakan metode yang mudah diimplementasikan dalam masalah teknis dan merupakan dasar dari teori kontrol lainnya.Metode kontrol optimum yang sering digunakan adalah *Linear Quadratic Regulator (LQR)*.Dalam LQR masalah fundamentalnya adalah menentukan matriks pembobotan yang menemukan kondisi optimal sistem dalam meminimalkan indeks performansi.Masalah pemilihan matriks pembobotan yang tepat dalam mendesain kontroler telah diperkenalkan pada berbagai macam metode.

Pemilihan matriks pembobotan pada LQR adalah sangat penting dan akan mempengaruhi input kontrol. Algoritma *Particle Swarm Optimization (PSO)* dan *Differential Evolution (DE)* merupakan algoritma optimasi dan dapat digunakan untuk menentukan matriks pembobotan dalam meminimumkan indeks performansi pada pendesainan LQR.Algoritma PSO diinspirasi oleh sekawanan burung dan ikan dalam mencari sumber makanan.Sedangkan algoritma *Differential Evolution (DE)* diadopsi dari evolusi dan genetika pada makhluk hidup.PSO dan DEA dipilih dalam pendesainan LQR, karena PSO dan DEA cukup sederhana dan dianggap mempunyai kecepatan komputasi yang tinggi. Algoritma PSO dan DEA telah banyak diaplikasikan dalam sistem landing pesawat [1,2,4,5].

## 2. Linear Quadratic Regulator (LQR)

Sistem Linear Time Invariant (LTI) direpresentasikan dalam persamaan ruang keadaan sebagai berikut [1,2,4,5].

$$\dot{x} = Ax + Bu \quad (1)$$

$$u = -Kx$$

dengan$x$ adalah vektor keadaan berukuran $n\times1$, $u$ adalah vektor input berukuran $m\times1$, $A$ dan $B$ adalah matriks konstan dan $A$ maatriks yang stabil, $K$ adalah matriks umpan balik. Fungsi objektif LQR didefinisikan sebagai berikut.

$$J = \frac{1}{2}\int_0^{t_f} [x^T(t)Qx(t) + u^T(t)Ru(t)]\, dt \quad (2)$$

dengan$Q$ adalah matriks semidefinit positif berukuran $n\times n$, dan $R$ adalah matriks definit positif berukuran $m\times m$. Masalah Kontrol optimum linier kuadratik adalah menentukan input kontrol yang optimal $u^*$ sedemikian hingga fungsi objektif $J$ minimum. Untuk $t_f = \infty$ , matrik umpan balik keadaan $K$ dirumuskan sebagai berikut.

$$K = R^{-1}B^T\bar{P} \quad (3)$$

dengan$\bar{P}$ merupakan solusi dari persamaan aljabar Riccati yang didefinisikan sebagai berikut.

$$\bar{P}A + A^T\bar{P} - \bar{P}BR^{-1}B^T\bar{P} + Q = 0 \quad (4)$$

## 3. Algoritma Particle Swarm Optimization (PSO)

Proses di dalam algoritma PSO dapat dijelaskan sebagai berikut. Sebanyak $p$partikel disebar secara acak pada ruang solusi yang ada. Posisi partikel $i$ saat waktu $t$ , yaitu $x_i(t)$ akan diperbaiki menurut persamaan posisi sebagai berikut.

$$x_i(t+1) = x_i(t) + v_i(t+1), \quad (5)$$

dengan$v_i(t+1)$ adalah kecepatan partikel yang dihitung dengan persamaan berikut.

$$v_i(t+1) = wv_i(t) + r_1n_1\left(p_{best\ i}(t) - x_i(t)\right) + r_2n_2(g_{best\ i}(t) - x_i(t)) , \quad (6)$$

Titik $pbest_i(t)$ adalah solusi lokal terbaik yang dicapai oleh partikel $i$ sampai saat $t$, dan merepresentasikan kontribusi kognitif terhadap vector $v_i(t+1)$. Titik $gbest_i(t)$ adalah solusi global terbaik yang telah dicapai diantara partikel-partikel sampai saat $t$ dan merepresentasikan kontribusi sosial terhadap vektor kecepatan.$gbest_i(t)$merepresentasikan kontribusi sosial terhadap vektor kecepatan artinya bahwa $gbest_i(t)$ akan dijadikan sebagai acuan setiap partikel *i* untuk mengupdate kecepatannya. Bilangan acak $r_1$ dan $r_2$ terdistribusi seragam pada interval [0,1]. Faktor skala kognitif ($n_1$) dan faktor skala sosial ($n_2$) pada algoritma PSO biasanya bernilai 2 [7].Faktor skala kognitif ($n_1$) merupakan percepatan konstan partikel yang mendorong setiap partikel tersebut menuju $pbest$nya. Sedangakan faktor skala sosial ($n_2$) merupakan percepatan konstan partikel yang mendorong setiap partikel tersebut menuju $gbest$. Variabel *w* adalah bobot inersia.Inersia yang besar memfasilitasi eksplorasi global (pencarian dalam daerah yang luas) sedangkan inersia yang kecil menghasilkan eksplorasi lokal (pencarian dalam daerah yang sempit).Oleh karena itu, nilai *w* merupakan faktor kritis yang menentukan perilaku konvergen algoritma PSO.Untuk itu direkomendasikan untuk memilih nilai *w* yang besar pada awalnya agar menghasilkan

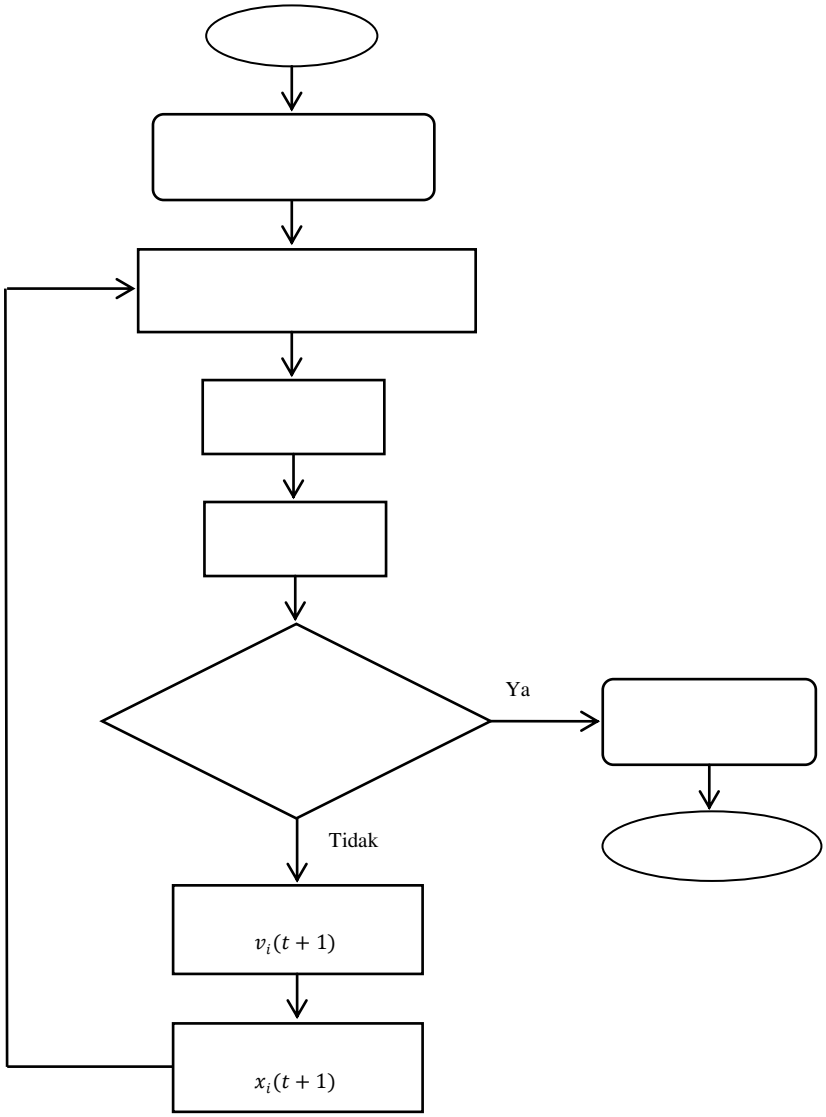
eksplorasi global pada ruang solusi, selanjutnya nilai *w* diturunkan secara bertahap untuk mendapatkan solusi yang lebih baik.

Menurut Xia dan Wu [8] bobot inersia *w* didefinisikan sebagai berikut .

$$w = w_{max} - \frac{w_{max} - w_{min}}{iterasi\ \ maksimum} d \ , \qquad (7)$$

dengan$d$ adalah iterasi.

Faktor pembelajaran kognitif yang dirumuskan sebagai $r_1 n_1 \left(p_{best\ i}(t) - x_i(t)\right)$ pada persamaan (3.2) merupakan memori partikel jangka pendek.Faktor pembelajaran kognitif ini menunjukkan inklinasi (kecenderungan) partikel untuk mengulang perilaku sebelumnya yang terbukti sukses pada partikel tersebut.Hal ini juga menunjukkan adanya pengaruh memori partikel tersebut.Faktor pembelajaran sosial yang diberikan oleh $r_2 n_2 (g_{best\ i}(t) - x_i(t))$ merupakan *"peer presure"* bagi sebuah partikel. Faktor pembelajaran sosial tersebut menunjukkan inklinasi (kecenderungan) partikel untuk meniru atau mengemulasi perilaku partikel lain yang telah sukses. Hal ini menunjukkan adanya pengaruh tetangga partikel. Setiap iterasi,sekawanan partikel dievaluasi nilai *fitness*-nya dan berdasarkan nilai tersebut, kecepatan dan posisi partikel diperbaruhi [8]. Proses algoritma PSO dapat disajikan dengan diagram alir pada Gambar 1.




**Gambar 1.** Diagram Alir Algoritma PSO

## 4. Differential Evolution Algorithm (DEA)

DEA mempunyai langkah-langkah yang diberikan pada tabel 1 berikut [6].

**Tabel 1** Tabel algoritma DEA

| |
|---|
| *Step 1. Initialization*<br>*Step 2. Evaluation*<br>*Step 3. REPEAT*<br>*Mutation*<br>*Recombination*<br>*Evaluation*<br>*Selection*<br>*UNTIL (stop criteria are met)* |

Suatu hal yang perlu dipahami sebelum membahas konsep DEA adalah bahwa suatu individu yang berisi nilai-nilai real bisa dipandang sebagai suatu vektor.Selanjutnya

menghitung perbedaan antara dua individu sebagai jarak antara dua vektor. DEA menggunakan sejumlah *NP* vektor parameterSebagai suatu populasi pada setiap generasi G. Selama proses pencarian nilai minimum (minimasi), vektor parameter tetap berjumlah NP. Populasi awal dibangkitkan secara acak.

$\underline{x}_{i,G},\ i = 0,1,2,\ldots,NP-1$

Selanjutnya, DEA membangkitkan suatu vektor parameter (individu) baru dengan melibatkan tiga individu sebagi orang tua. Pemilihan orang tua dilakukan dengan probabilitas yang sama untuk setiap individu tanpa memperhatikan nilai fungsi objektifnya. Pembangkitan vektor baru dilakukan dengan menambahkan vektor perbedaan antara dua vektor (orang tua ke-1 dan ke-2) kepada vektor lainnya (orang tua ke-3).Vektor yang dilibatkan dalam pembangkitan individu baru disebut orang tua.Pembangkitan vektor baru ini disebut mutasi *(mutation)*. Pada DEA, suatu bobot *(weight)* diberikan terhadap perbedaan antara dua vektor orang tua. Jika vektor baru yang dihasilkan memberikan nilai fungsi objektif yang lebih kecil daripada suatu vektor lainnya dalam populasi, maka vektor baru ini akan menggantikan vektor tersebut. Vektor yang dijadikan bandingan bisa (tetapi tidak harus) berasal dari ketiga vektor orang tua tersebut. Untuk menjaga jalur perkembangan yang dibuat selama proses minimasi, vektor parameter $\underline{x}_{best,G}$ dievaluasi pada setiap generasi G [9]. Proses mutasi (*mutation*), Rekombinasi (*crossover*), dan seleksi (*selection*) pada DEA akan dijelaskan sebagai berikut.

**a. Mutasi (*Mutation*)**

Pembangkitan vektor baru (mutasi) bisa dilakukan dengan beragam skema.Saat ini sudah banyak skema yang diusulkan oleh para ahli.Pertama kali Rainer Storn dan Kenneth Price [6] mengusulkan sebagai berikut.

Untuk setiap vektor $\underline{x}_{i,G}$ , $i = 0,1,2,\ldots,NP-1$ , suatu vektor baru $\underline{v}$ dibangkitkan dengan rumus berikut ini [17,18].

$$\underline{v}_{i,G+1} = \underline{x}_{r1,G} + F.(\underline{x}_{r2,G} - \underline{x}_{r3,G}) , \tag{8}$$

dengan$r_1, r_2, r_3 \epsilon [0, NP-1]$ adalah integer berbeda dan $F > 0$.

Ketiga bilangan bulat $r_1, r_2, r_3$ harus berbeda satu sama lain dan dipilih secara acak dalam interval $[0, NP-1]$. Ketiga bilangan tersebut menyatakan indeks orang tua dan bisa berbeda untuk setiap proses pembangkitan individu baru. *F* adalah suatu bilangan real dan merupakan konstanta yang mengontrol penguatan *differential variation* $(\underline{x}_{r2,G} - \underline{x}_{r3,G})$. Proses pembentukan individu baru ini disebut *differential mutation* [9].

**b. Rekombinasi (*crossover*)**

Untuk meningkatkan keberagaman *(diversity)* vektor-vektor parameter, maka vektor $\underline{v}$ direkombinasi *(crossover)* dengan suatu vektor sebarang dalam populasi, misal $\underline{x}_{i,G}$. Proses *crossover* ini menghasilkan vektor $\underline{u}$ berikut ini [6].

$$\underline{u}_{i,G+1} = \begin{cases} \underline{v}_{i,G+1} & jika\ rand_j(0,1) \le Cr \ \vee j = k \\ (\underline{x}_{i,G}) & untuk\ \ lainnya \end{cases} , \tag{9}$$

dengan$j = 1,2,3, \ldots Drand_j$ adalah evaluasi ke-$j$ dari generasi G yang diperoleh secara acak antara 0 dan 1, $k\epsilon\{1,2,3,..,D\}$ adalah indeks parameter acak, dan D adalah dimensi fungsi yang dioptimasi.

Selanjutnya, vektor $\underline{u}$ akan menggantikan vektor $\underline{x}_{i,G}$ pada generasi berikutnya jika $\underline{u}$ memberikan nilai lebih kecil untuk fungsi objektif tersebut daripada $\underline{x}_{i,G}$. Tetapi, jika $\underline{u}$ memberikan nilai lebih besar daripada $\underline{x}_{i,G}$, maka $\underline{u}$ tidak menggantikan $\underline{x}_{i,G}$. Dengan kata lain, $\underline{x}_{i,G}$akan tetap muncul pada generasi berikutnya [9].

**c. Seleksi (*Selection*)**

Proses seleksi dilakukan untuk memutuskan apakah individu $\underline{u}_{i,G+1}$akan menjadi anggota populasi pada generasi (G+1).

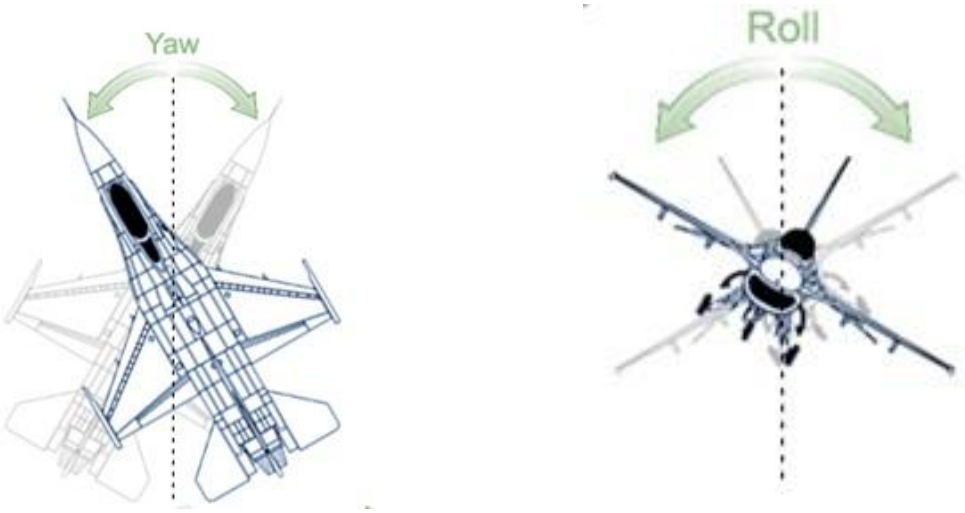
Proses seleksi dirumuskan sebagai berikut.

$$x_{i,G+1} = \begin{cases} \underline{u}_{i,G+1} & ,jika\ f(\underline{u}_{i,G+1}) < f(x_{i,G}) \\ x_{i,G} & ,untuk\ \ lainnya \end{cases}, \quad (10)$$

dengan$f(.)$ adalah fungsi *fitness* (fungsi objektif) [5].

## 5. Studi Kasus

Dalam paper ini akan mengambil studi kasus tentang sistem dinamik gerak lateral pesawat F-16 saat kondisi terbang dengan kecepatan 502 feet/sec, tekanan 300 psf. Gerak lateral pesawat F-16 diilustrasikan pada gambar berikut.




Gambar 2. Ilustrasi gerak lateral pesawat F-16

Sistem dinamik gerak lateral pesawat F-16 saat kondisi terbang yang telah dilinierisasi merupakan sistem LTI dan diilustrasikan dalam persamaan ruang keadaan sebagai berikut [3].

$$\dot{x} = Ax + Bu \quad (12)$$

dengan$x = [\beta \quad \varphi \quad p \quad r \quad \delta_a \quad \delta_r \quad x_w]^T$

$$A = \begin{bmatrix} -0.3220 & 0.0640 & 0.0364 & -0.9917 & 0.0003 & 0.0008 & 0 \\ 0 & 0 & 1 & 0.0037 & 0 & 0 & 0 \\ -30.6492 & 0 & -3.6784 & 0.6646 & -0.7333 & 0.1315 & 0 \\ 8.5396 & 0 & -0.0254 & -0.4764 & -0.0319 & -0.0620 & 0 \\ 0 & 0 & 0 & 0 & -20.2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -20.2 & 0 \\ 0 & 0 & 0 & 57.2958 & 0 & 0 & -1 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 20.2 & 0 \\ 0 & 20.2 \\ 0 & 0 \end{bmatrix} \text{dan } u = \begin{bmatrix} u_a \\ u_r \end{bmatrix},\ u \text{ adalah masukan sistem [3].}$$

dengan kondisi awal sistem sebagai berikut [3].

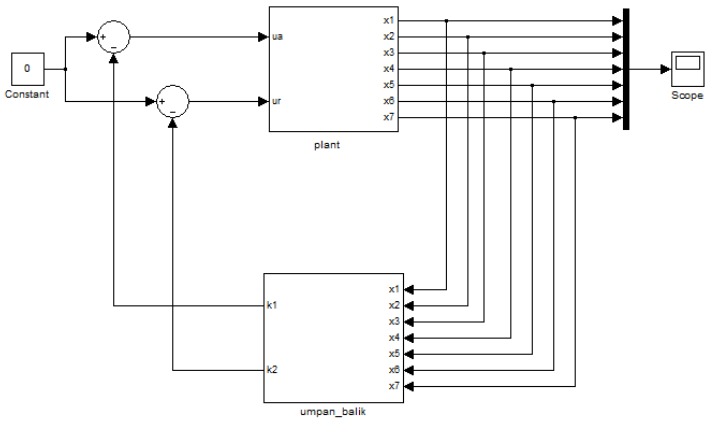
$$x(0) = [\beta(0) \quad \varphi(0) \quad p(0) \quad r(0) \quad \delta_a(0) \quad \delta_r(0) \quad x_w(0)]^T$$

$$= [1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0]^T$$

$x$ : *lateral state*, $\beta$ : *sideslip angle (deg)*, $\varphi$ : *bank angle (deg)*, $p$ : *roll* rate/kecepatan sudut guling *(deg/sec)*, $r$ : *yaw rate*/kecepatan sudut geleng *(deg/sec)*, $\delta_a$ : *aileron actuator (deg)*, $\delta_r$ : *rudder actuator (deg)*, $x_w$ : *washout filter state (deg/sec)*, $u_a$ : *aileron deflection (deg)*, $u_r$ : *rudder deflection (deg)*.

Dalam pendesainan menggunakan algoritma PSO dan DEA dengan metode *invinite horizon* dengan meminimumkan indeks performansi sebagai berikut.

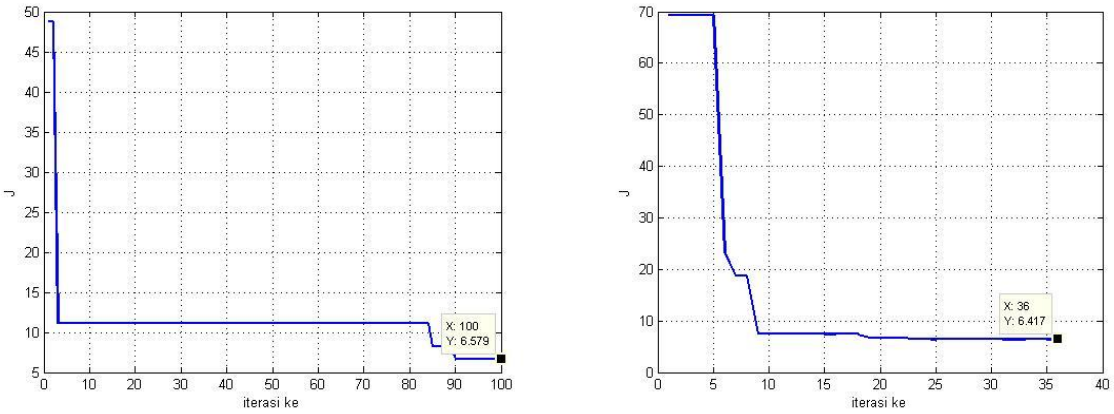
$$J = \frac{1}{2}\int_0^\infty (x^T Q x + u^T R u)\, dt$$




Gambar 3 Diagram Blok Keseluruhan Sistem

Dengan melakukan dua puluh kali *running* diperoleh nilai indeks performansi minimum sebesar 6.579. Setiap kali *running*akan membangkitkan matriks Q dan R yang berbeda.

Matriks pembangkitan pada *running* ke-i tidak akan sama dengan matriks hasil pembangkitan *running* ke-i+1.




(a) (b)

Gambar 4 Nilai indeks performansi minimum setiap iterasi (a)Hasil algoritma PSO

(b) Hasil Algoritma DEA

Gambar 4(a) menunjukkan nilai indeks performansi telah konvergen dengan nilai optimum 6.579 detik dengan konstanta random $r_1 = 0.3276, r_2 = 0.6713$ dan membutuhkan waktu rata-rata kinerja CPU dalam dua puluh kali *running* sebesar 21.14403 detik. Nilai indeks performansi sudah optimum berarti matriks $Q$, $R$, dan umpan balik keadaan $K$ telah diperoleh. Matriks $Q$, $R$, dan $K$ optimum yang dihasilkan oleh algoritma PSO adalah sebagai berikut.

$$Q = \begin{bmatrix} 0.6835 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.6639 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.2372 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.6113 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.1375 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.3772 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 9.5663 \times 10^{-4} \end{bmatrix}$$

$$R = \begin{bmatrix} 0.8554 & 0 \\ 0 & 0.9180 \end{bmatrix}$$

$$K^T = \begin{bmatrix} 0.3230 & 0.0022 \\ -0.7321 & -0.0643 \\ -0.2055 & -0.0156 \\ -0.7125 & -0.1196 \\ 0.5320 & 5.0026 \times 10^{-4} \\ 5.3689 \times 10^{-4} & 0.5813 \\ -3.0305 \times 10^{-4} & -1.4481 \times 10^{-4} \end{bmatrix}$$

dengan menggunakan rumus $u^* = Kx$ diperolehkontrol optimum yang bersesuaian dengan $u_a$ sebagai berikut.

$$u^* = 0.3230\beta - 0.7321\varphi - 0.2055p - 0.7125r + 0.5320\delta_a + 5.3689 \times 10^{-4}\delta_r - 3.0305 \times 10^{-4}x_w$$

dan kontrol optimum yang bersesuaian dengan $u_r$ adalah

$$u^* = 0.0022\beta - 0.0643\varphi - 0.0156p - 0.1196r + 5.0026 \times 10^{-4}\delta_a + 0.5813\delta_r - 1.4481 \times 10^{-4}x_w$$

Gambar 4(b) menunjukkan nilai indeks performansi minimum sebesar 6.417.DEA membutuhkan waktu rata-rata kinerja CPU dalam dua puluh kali *running* sebesar 10.60157 detik. DEA akan dipengaruhi oleh evaluasi ke-$j$ dari genarasi G ($rand_j$) dalam studi kasus ini nilai $rand_j$ dilakukan secara otomatis oleh matlab dan hasil yang diperoleh adalah 0.9119. Matriks pembobotan error $Q$, matriks pembobotan kontrol $R$ yang meminimumkan fungsional objektif (indeks performansi) pada persamaan (2) serta umpan balik optimum $K$ hasil dari algoritma DEA adalah sebagai berikut.

$$Q = \begin{bmatrix} 1.0765 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.7041 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.2399 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.4999 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.3699 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.3407 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 9.5063 \times 10^{-4} \end{bmatrix}$$

$$R = \begin{bmatrix} 0.2009 & 0 \\ 0 & 0.3477 \end{bmatrix}$$

$$K^T = \begin{bmatrix} 0.9830 & -0.0150 \\ -1.6538 & -0.1035 \\ -0.4828 & -0.0231 \\ -1.6320 & -0.2676 \\ 0.6974 & 7.6128 \times 10^{-4} \\ -0.0013 & 0.4075 \\ -0.0011 & -4.1573 \times 10^{-4} \end{bmatrix}$$

dengan menggunakan rumus $u^* = Kx$ diperolehkontrol optimum yang bersesuaian dengan $u_a$ sebagai berikut.

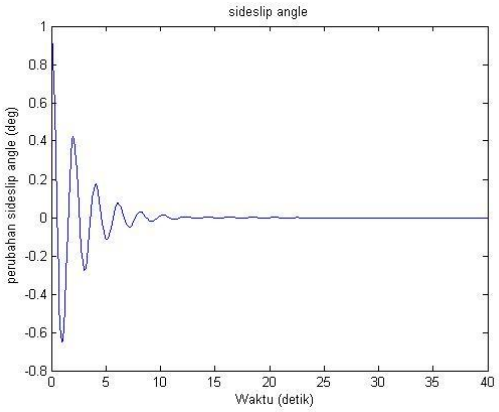
$$u^* = 0.9830\beta - 1.6538\varphi - 0.4828p - 1.6320r + 0.6974\delta_a - 0.0013\delta_r - 0.0011x_w$$

dan kontrol optimum yang bersesuaian dengan $u_r$ adalah

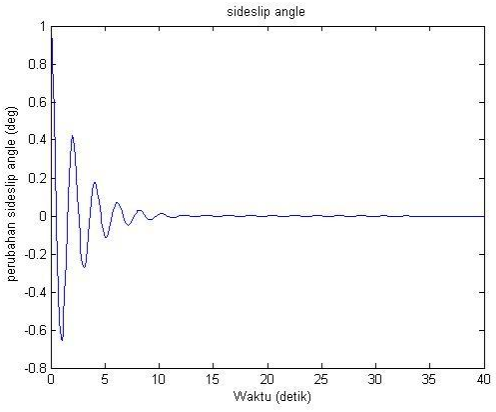
$$u^* = -0.0150\beta - 0.1035\varphi - 0.0231p - 0.2676r + 7.6128 \times 10^{-4}\delta_a + 0.4075\delta_r - 4.1573 \times 10^{-4}x_w$$

Dalam teori kontrol nilai toleransi yang diijinkan sistem untuk mencapai keadaan tunak adalah 5% dan 2% atau secara langsung menunjukkan kestabilan relatif sistem [7]. Dalam studi kasus ini, akan mengambil nilai toleransi sebesar 2% dimana sistem hasil pendesainan algoritma PSO akan dibandingkan dengan hasil DEA.

Respon sistem hasil dari pendesainan algoritma PSO dan DEA diberikan pada gambar berikut.



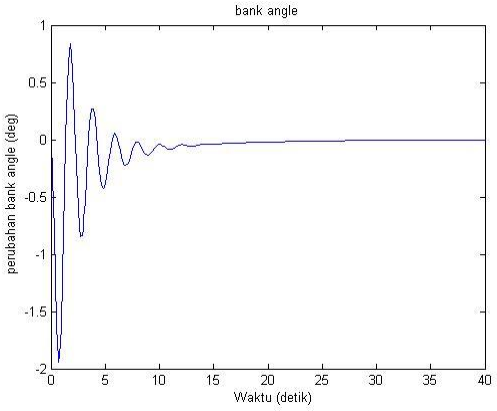

(a) Respon waktu *Sideslip Angle* ($\beta$) Menggunakan PSO



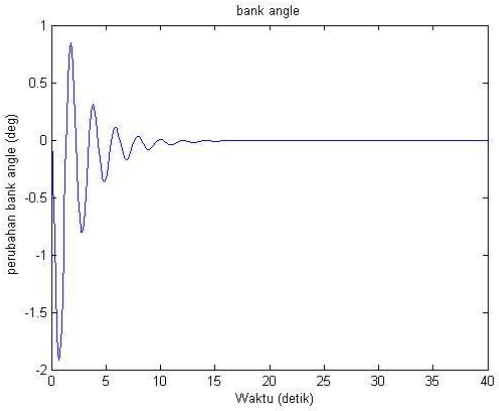

(b) Respon waktu *Sideslip Angle* ($\beta$) Menggunakan DEA

Gambar 5Respon waktu *Sideslip Angle* ($\beta$)

Simulasi *Sideslip Angle* ($\beta$) menggunakan algoritma PSO dan DEA ditunjukkan pada Gambar 5.respon waktu *Sideslip Angle* ($\beta$) hasil pendesainan PSO dan DEA akan mencapai keadaan stabil relatif berturut-turut waktu 9.3725 detik dan 9.3084 detik.



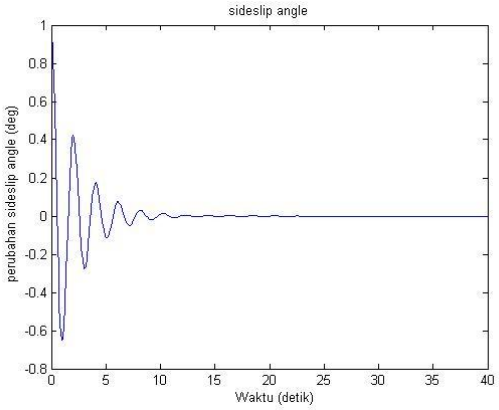

(a) Respon waktu *Bank Angle* ($\varphi$)Menggunakan PSO



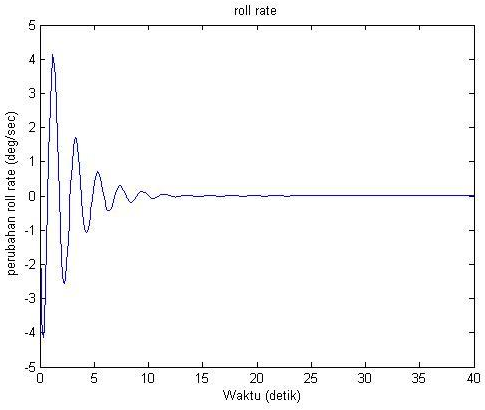

(b) Respon waktu *Bank Angle* ($\varphi$)Menggunakan DEA

Gambar 6Respon waktu *Bank Angle* ($\varphi$)

Simulasi *Bank Angle* ($\varphi$)menggunakan algoritma PSO dan DEA ditunjukkan pada Gambar 6. respon waktu *Bank Angle* ($\varphi$)hasil pendesainan PSO dan DEA akan mencapai keadaan stabil relatif berturut-turut waktu 20.4034 detik dan 13.2145 detik .



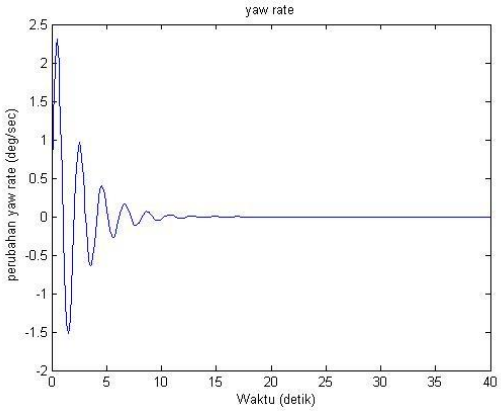

(a) Respon waktu *Roll Rate* ($p$)Menggunakan PSO



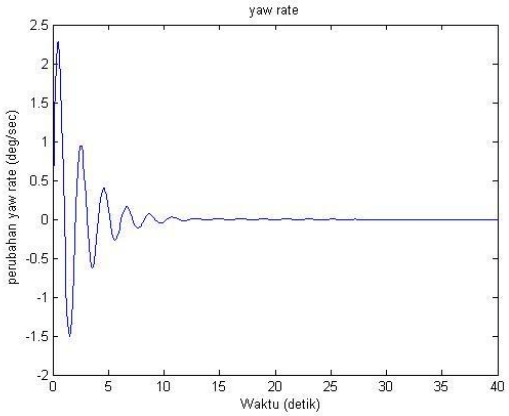

(b) Respon waktu *Roll Rate* ($p$)Menggunakan DEA

Gambar 7Respon waktu *Roll Rate* ($p$)

Simulasi *Roll Rate* ($p$) menggunakan algoritma PSO dan DEA ditunjukkan pada Gambar 7.respon waktu *Roll Rate* ($p$)hasil pendesainan PSO dan DEA akan mencapai keadaan stabil relatif berturut-turut waktu 13.8642detik dan 13.78detik .



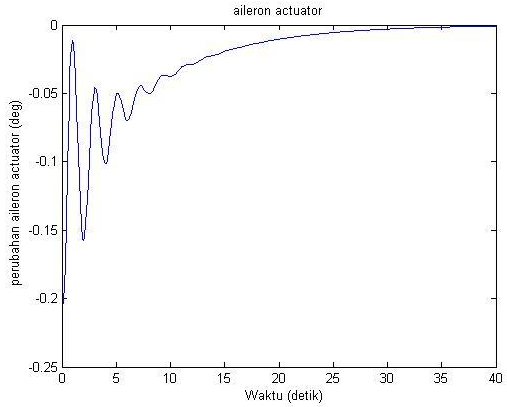

(a) Respon waktu *Yaw Rate* ($r$)Menggunakan PSO



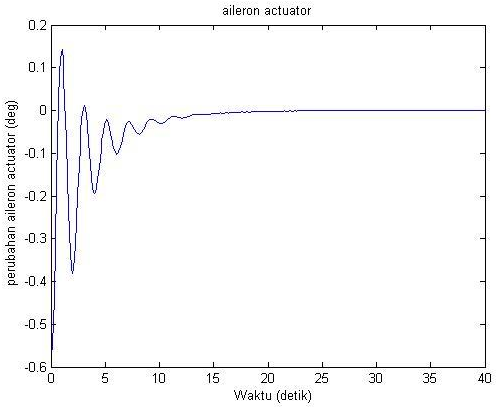

(b) Respon waktu *Yaw Rate* ($r$)Menggunakan DEA

Gambar 8Respon waktu *Yaw Rate* ($r$)

Simulasi *Yaw Rate* ($r$) menggunakan algoritma PSO dan DEA ditunjukkan pada Gambar 8.respon waktu *Yaw Rate* ($r$)hasil pendesainan PSO dan DEA akan mencapai keadaan stabil relatif berturut-turut waktu 11.9767detik dan 11.0432 detik .



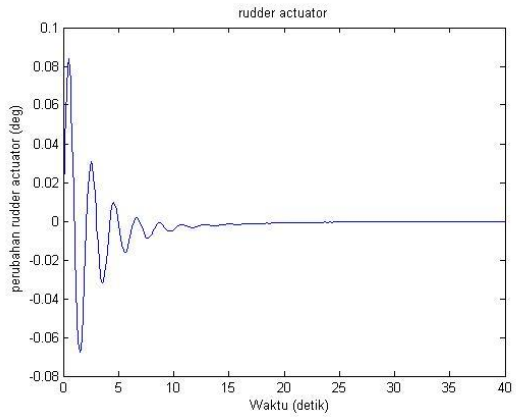

(a) Respon waktu *Alerion Actuator* ($\delta_a$)Menggunakan PSO



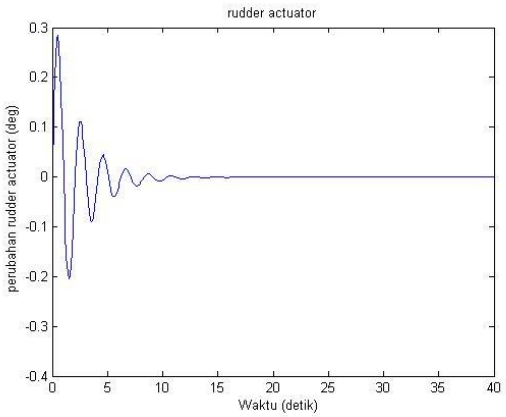

(b) Respon waktu *Alerion Actuator*($\delta_a$)Menggunakan DEA

Gambar 9Respon waktu *Alerion Actuator* ($\delta_a$)

Simulasi *Alerion Actuator* ($\delta_a$)menggunakan algoritma PSO dan DEA ditunjukkan pada Gambar 9.respon waktu *Alerion Actuator* ($\delta_a$)hasil pendesainan PSO dan DEA akan mencapai keadaan stabil relatif berturut-turut waktu 14.804 detik dan 10.855detik .



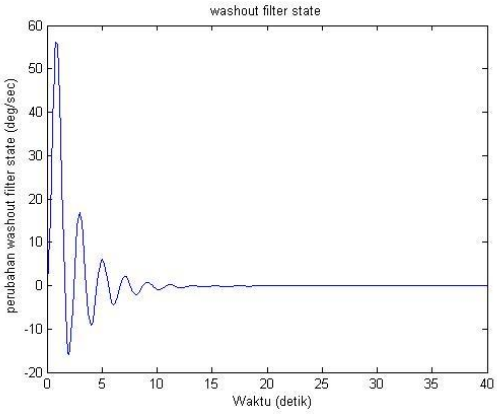

(a) Respon waktu *Rudder Actuator* ($\delta_r$)Menggunakan PSO



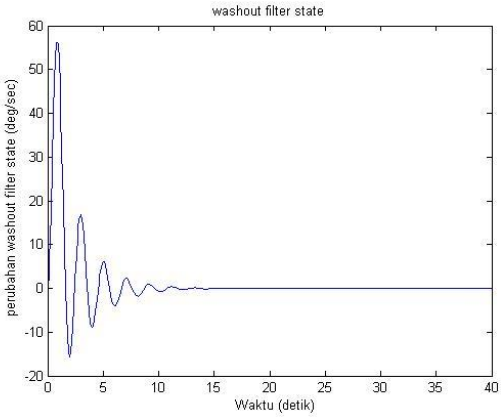

(b) Respon waktu *Rudder Actuator* ($\delta_r$)Menggunakan DEA

Gambar 10Respon waktu *Rudder Actuator* ($\delta_r$)

Simulasi *Rudder Actuator* ($\delta_r$)menggunakan algoritma PSO dan DEA ditunjukkan pada Gambar 10.respon waktu *Rudder Actuator* ($\delta_r$)hasil pendesainan PSO dan DEA akan mencapai keadaan stabil relatif berturut-turut waktu 3.983detik dan 5.998detik .




(a) Respon waktu *Washout Filter State* ($x_w$) Menggunakan PSO




(b) Respon waktu *Washout Filter State* ($x_w$) Menggunakan DEA

Gambar 11Respon waktu *Washout Filter State* ($x_w$)

Simulasi *Rudder Actuator* ($\delta_r$)menggunakan algoritma PSO dan DEA ditunjukkan pada Gambar 11. respon waktu *Rudder Actuator* ($\delta_r$)hasil pendesainan PSO dan DEA akan mencapai keadaan stabil relatif berturut-turut waktu 31.4364 detik dan 18.836 detik .

## 6. Kesimpulan

Algoritma PSO dan DEA merupakan algoritma berbasis komputasi yang digunakan untuk menyelesaikan masalah optimasi.Salah satu contoh aplikasi dari algoritma PSO dan DEA adalah menentukan matriks pembobotan $Q$ dan $R$ dalam masalah LQR.Berdasarkan studi kasus, Algoritma DEA mempunyai kecepatan komputasi yang lebih tinggi dibandingkan dengan PSO. Dalam studi kasus ini, secara umum DEA mempunyai respon waktu yang lebih baik dibanding PSO dimana sistem akan lebih cepat menuju keadaan *steady state* (mantap). Algoritma PSO menghasilkan indeks performansi optimum sebesar 6.579 dengan membutuhkan waktu kinerja rata-rata CPU (*CPU Time*) dalam dua puluh kali *running* sebesar 21.14403 detik dan DEA menghasilkan indeks performansi optimum sebesar 6.417 dengan membutuhkan waktu kinerja rata-rata *CPU Time* dalam dua puluh kali *running* sebesar 10.60157 detik. Perbedaan kecepatan komputasi algoritma PSO dan DEA dipengaruhi oleh parameter-parameter yang dilibatkan dalamnya. Namun, tidak menutup kemungkinan untuk studi kasus yang berbeda dan penggunaan nilai parameter yang berbeda akan menunjukkan pola hasil yang berbeda pula. Aplikasi dari algoritma PSO dan DEA dapat dikembangkan dalam masalah kontrol optimum lainnnya seperti maslah LQR-*tracking*.

## 7. Referensi

[1] Ghoereishi, S. Amir. Arash Ahmadivand. March 2012. *State Feedback Design Aircraft Landing System With Using Differential Evolition Algorithm*. Advances in Computer Science and its Applications, Vol. 1, No 1.pp. 16-20.

[2] Hamidi, J. 2012. *Control System Design Using Particle Swarm Optimization (PSO)*. International Journal of Soft Computing and Engineering (IJSCE).Volume 1.Isssue 6. Pp 116-119

[3] Masten, Michael. K. et all. 1995. *Modern Control Systems*. Institute of Electrical and Electronics Engineers, Inc. USA.

[4] Mobayen S., Mohamady B.,H. Ghorbani, A. Rabii. January 2012. *Optimal Control Design Using Evolutary Algorithms With Application to an Aircraft Landing System.*Journal of Basic and Applied Scientiffic Research. Vol 2.pp 1876-1882

[5] Mobayen.S, A. Rabiei. M. Morabi, and B. Mohammady. November 2011. *Linier Quadratic Optimal Control System Design Using Particle Swarm Optimization Algoritm*. International Journal of The Physical Sciences. Vol. 6(30). Pp. 6958-6966

[6] Nuri Seyman Muhammet, Taspinar Necmi. 2012. *Optimization of Pilot Tones Using Differential Evolution Algorithm in MIMO-OFDM Systems*. Journal Electronics Engineering and Computer Science.Vol 20.No 1. pp 15-23

[7] Ogata, Katsuhiko. 1990. *Teknik Kontrol Automatik (Sistem Pengaturan) jilid I*. Alih Bahasa Oleh Edi Leksono. Penerbit Erlangga : Jakarta

[8] Ratna wati, Dwi Ana. 2011. *Sistem Kendali Cerdas*. Penerbit Graha Ilmu : Yogyakarta

[9] Suyanto. 2008. *Evolutionary Computation : Komputasi Berbasis "Evolusi" dan "Genetika"*. Penerbit Informatika : Bandung

[10] Zeilinski, Karin. Rainer, Laur. 2007. *Stopping Criteria for a Constrained Single-Objective Particle Swarm Optimization Algorithm*. Journal Informatica.Vol. 31. Pp 51-59