TESIS - TE142599

PENGEMBANGAN *ADAPTIVE PARTICLE SWARM OPTIMIZATION* (PSO) DAN APLIKASINYA PADA PERENCANAAN JALUR *MOBILE ROBOT* DENGAN HALANGAN DINAMIS

NOVENDRA SETYAWAN
2215202004

DOSEN PEMBIMBING
Prof. Dr. Ir. Achmad Jazidie, M.Eng.
Ir. Rusdhianto Effendi A.K, M.T

PROGRAM MAGISTER
BIDANG KEAHLIAN TEKNIK SISTEM PENGATURAN
DEPARTEMEN TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI ELEKTRO
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2017

TESIS – TE142599

PENGEMBANGAN *ADAPTIVE PARTICLE SWARM OPTIMIZATION* (PSO) DAN APLIKASINYA PADA PERENCANAAN JALUR *MOBILE ROBOT* DENGAN HALANGAN DINAMIS

NOVENDRA SETYAWAN
2215202004

DOSEN PEMBIMBING
Prof. Dr. Ir. Achmad Jazidie, M.Eng.
Ir. Rusdhianto Effendi A.K, M.T.

PROGRAM MAGISTER
BIDANG KEAHLIAN TEKNIK SISTEM PENGATURAN
DEPARTEMEN TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI ELEKTRO
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2017

LEMBAR PENGESAHAN

Tesis disusun untuk memenuhi salah satu syarat memperoleh gelar
Magister Teknik (M.T.)
di
Institut Teknologi Sepuluh Nopember

oleh:

Novendra Setyawan
NRP. 2215202004

Tanggal Ujian : 8 Juni 2017
Periode Wisuda : September 2017

Disetujui oleh:

1. Prof. Dr. Ir. Achmad Jazidie, M.Eng. (Pembimbing I)
   NIP: 19590219 198610 1 001

2. Ir. Rusdhianto Effendi A.K., M.T. (Pembimbing II)
   NIP: 19570424 198502 1 001

3. Prof. Dr. Ir. Mochammad Nuh, DEA. (Penguji)
   NIP: 19590617 198403 1 002

4. Dr. Trihastuti Agustinah, S.T., M.T. (Penguji)
   NIP: 19680812 199403 2 001

Dekan Fakultas Teknologi Elektro

Dr. Tri Arief Sardjono, S.T., M.T.
NIP. 19700212 199512 1 001

Halaman ini sengaja dikosongkan

PERNYATAAN KEASLIAN TESIS

Dengan ini saya menyatakan bahwa isi keseluruhan Tesis saya dengan judul "PENGEMBANGAN *ADAPTIVE PRTICLE SWARM OPTIMIZATION* (PSO) DAN APLIKASINYA PADA PERENCANAAN JALUR *MOBILE ROBOT* DENGAN HALANGAN DINAMIS" adalah benar-benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diijinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka. Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Surabaya, Maret 2016

Novendra Setyawan

2215202004

Halaman ini sengaja dikosongkan

PENGEMBANGAN *ADAPTIVE PARTICLE SWARM OPTIMIZATION* (PSO) DAN APLIKASINYA PADA PERENCANAAN JALUR *MOBILE ROBOT* DENGAN HALANGAN DINAMIS

Nama mahasiswa : Novendra Setyawan
NRP : 2215202004
Pembimbing : 1. Prof. Dr. Ir. Achmad Jazidie, M.Eng.
2. Ir. Rusdhianto Effendi A.K, M.T.

ABSTRAK

Kunci sukses dari navigasi sebuah *mobile robot* bergantung pada pembangkitan trajektori atau perencanaan jalur. Perencanaan jalur berbasis metode optimasi heuristik dikembangkan untuk menyederhanakan permasalaahan perencanaan jalur menjadi permasalahan optimasi. Salah satu metode optimasi heuristik yang sering digunakan dalam perencanaan jalur adalah *Particle Swarm Optimization* (PSO) karena kesederhanaan pada algoritmanya, mudah diimplementasikan dan memiliki sedikit parameter untuk diatur. Akan tetapi pada permasalahan perencanaan jalur yang kompleks dengan lingkungan dinamis, algoritma dasar PSO tidak dapat menjamin menemukan solusi optimal *(local optimum)* dikarenakan konvergensi prematur yang menyebabkan terjadi tumbukan dengan halangan dan jalur yang lebih panjang.

Pada penelitian ini setelah perilaku pencarian PSO dianalisa, PSO adaptif dikembangkan dengan menggunakan fungsi Gaussian dalam pengaturan nilai parameter pada PSO untuk mempercepat konvergensi dan reinisialisai partikel dilakukan untuk mencegah terjadinya konvergensi prematur.

Simulasi dan perbandingan dengan algoritma *Adaptif Inertia* (AIW) *PSO* dan standard PSO menunjukan algoritma yang diusulkan dapat menemukan solusi optimal lebih cepat dengan konvergensi kurang dari 150 iterasi pada halangan statis dan 200 iterasi pada halangan bergerak. Selain itu algoritma yang diusulkan memiliki 3% panjang lintasan yang lebih pendek, 10% lebih *smooth* dan lebih terjamin terhindar dari tumbukan.

Kata kunci: *Particle Swarm Optimization, Adaptive Parameter PSO, Mobile Robot,* Optimisasi Multi Tujuan

Halaman ini sengaja dikosongkan

MODIFICATED ADAPTIVE PARTICLE SWARM OPTIMIZATION AND ITS IMPLEMENTATION IN MOBILE ROBOT PATH PLANNING

By : Novendra Setyawan
Student Identity Number : 2215202004
Supervisor(s) : 1. Prof. Dr. Ir. Achmad Jazidie, M.Eng.
2. Ir. Rusdhianto Effendi A.K, M.T.

ABSTRACT

The success key in mobile robot navigation depends on trajectory generation or path planning. Path planning based on heuristic optimization method is developed to simplify the path planning issues into optimization problems. One of the heuristic optimization methods used in path planning is the Particle Swarm Optimization (PSO) that is often used because of its simplicity, easy to implement and has few parameters to set. However, in the case of complex path planning with dynamic environments, the PSO basic algorithm can not guarantee finding the optimum solution (local optimum) due to premature convergence that causes collisions, with longer obstacles and paths.

In the proposed method, the Gaussian parameter updating rule use to speed up the convergence by maintaining exploration and exploitation of the particle. Then, particle re-initialization is proposed after analyzing the behavior of PSO algorithm to prevent premature convergence.

Simulation result shows in benchmark test with Adaptive Inertia (AIW) PSO and standard PSO that proposed PSO algorithm can find optimal solution faster than the other algorithm which can convergence in less than 150 iteration in static obstacle and 200 iteration in dynamic obstacle. Proposed PSO algorithm with particle re-initialization can guarantee to find optimal solution with resulting path 3% more shortest, 10% more *smooth* and guaranteed to collision free path.

Key words: Particle Swarm Optimization; Mobile Robot; Path Planning; Multi-Objective Optimization

Halaman ini sengaja dikosongkan

KATA PENGANTAR

Alhamdulillahirabbil'alamin,

Segala puji dan syukur senantiasa kita panjatkan ke hadirat Allah SWT atas segala nikmat, kekuatan, taufik serta hidayah-Nya. Shalawat serta salam semoga tercurah kepada Rasulullah SAW, keluarga sahabat dan para pengikut setianya, Amin. Atas kehendak Allah sajalah, penulis dapat menyelesaikan tesis yang berjudul :

"PENGEMBANGAN *ADAPTIVE PARTICLE SWARM OPTIMIZATION* (PSO) DAN APLIKASINYA PADA PERENCANAAN JALUR *MOBILE ROBOT* DENGAN HALANGAN DINAMIS"

Selain itu ucapan terimakasih penulis sampaikan kepada kedua orang tua atas kepercayaan yang telah diberikan, dukungan, dan doa-doa, kepada para pembimbing yang telah memberikan bimbingan, arahan dan motivasi kepada penulis sehingga dapat menyelesaikan tesis ini, untuk teman teman penulis yang telah memberikan bantuan fisik maupun moril kepada penulis, serta yang terakhir penulis ucapkan banyak terimakasih kepada istri tercinta, yang telah memberikan dukungan dan menjadi motivasi terbesar dalam menyelesaikan tesis ini.

Akhir kata penulis berharap semoga tesis ini dapat menambah literatur dan dapat memberikan manfaat bagi pembaca dan terutama bagi penulis.

.

Surabaya, 12 Juni 2016

Penulis

Halaman ini sengaja dikosongkan

DAFTAR ISI

DAFTAR GAMBAR

Halaman ini sengaja dikosongkan

DAFTAR TABEL

Halaman ini sengaja dikosongkan

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Penelitian mengenai *Mobile robot* telah banyak dilakukan karena aplikasinya sangat luas pada berbagai bidang seperti pada bidang maritim, penanggulangan bencana, industri manufaktur, dan transportasi. Pada aplikasi tersebut robot dituntut agar dapat bernavigasi secara autonomous, sehingga dapat mengurangi peranan manusia dalam melaksanakan pekerjaan tersebut.

Salah satu tantangan agar robot dapat bernavigasi secara autonomus adalah mendapatkan jalur yang cepat, efisien dan aman untuk mencapai tujuan. Hal tersebut adalah permasalahan utama pada *path planning* atau perencanaan jalur. Pada saat ini penelitian mengenai perencanaan jalur terbagi menjadi dua bagian utama yaitu metode klasik dan metode berbasis metaheuristik. Metode metode klasik seperti Cell Decomposition [1], Potential Field [2], and Visibility Graph [3] telah di usulkan pada waktu yang sangat lampau. Metode klasik memiliki kekurangan seperti komputasi yang sangat lama dan rumit [1]

Tidak dapat diragukan bahwa permasalahan *path planning* dapat dilihat sebagai permasalahan optimasi multi-tujuan dengan fungsi tujuan seperti, panjang *path,* waktu tempuh, energi dan dengan batasan menjauh dari halangan. Salah satu alternatif metode penyelesaian optimasi adalah dengan menggunakan metode *heuristic* yang berbasis *artificial intelligent* seperti, *Particle Swarm Optimization* (PSO), Algoritma Genetika (GA), dan *Ant Colony Optimization* (ACO). Pada permasalahan *path planning* statis, metode metode tersebut terbukti dapat menghasilkan jalur yang efektif dan efisien dibandingkan metode klasik. [2]

Terinspirasi dari kejadian yang ada di alam, PSO pertamakali diusulkan pada tahun 1995 dan telah digunakan untuk menyelesaikan berbagai macam permasalahan optimisasi. Penelitian mengenai implementasi menggunakan PSO diantaranya, implementasi PSO untuk menghindari halangan pada permainan sepak bola robot [4], implementasi PSO untuk menghindari halangan pada *home service*

manipulator robot[5], dan hibridisasi dengan menggunakan metode heuristik lain telah dilakukan oleh [6].

Kunci sukses perencanaan jalur yang berbasi metode heuristik adalah berdasarkan permodelan permasalahan perencanaan jalur dengan lingkunganya dan performa algoritma optimisasi. Pada penelitian penelitian yang telah disebutkan diatas perencanaan jalur telah diformulasikan menjadi fungsi tujuan tunggal yang pada umumnya adalah fungsi panjang jalur. Bagaimanapun juga, formulasi yang hanya menggunakan panjang lintasan saja tidak menjamin bahwa jalur yang dibuat aman dan *smooth*. Pada penelitian ini permasalahan perencanaan jalur diformulasikan menggunakan tiga buah fungsi tujuan yakni, panjang lintasan, tingkat resiko tumbukan, dan kriteria *smooth.*

Pada umumnya alasan utama dalam penggunaa PSO pada perencanaan jalur adalah karena kesederhanaan, mudah diimplementasikan dan memiliki sedikit parameter untuk diatur[4] [5]. Bagaimanapun juga, disamping kelebihan tersebut PSO juga memiliki kekurangan yaitu dapat terjebak dalam lokal minimum atau konvergensi prematur pada implementasi perencanaan jalur pada lingkungan yang komplek yang merupakan permasalahan optimisasi multi modal yang memiliki banyak kemungkinan jalur untuk dilalui dan pada lingkungan yang memiliki halangan yang dinamis [4].

Analisa konvergensi PSO secara teoritis telah di lakukan oleh Van den Bergh [7]. PSO dianalisa dalam asusmsi bahwa proses pencarian dilakukan secara deterministik dan dilakukan pada satu buah partikel. Pada analisa tersebut PSO akan konvergen ketika syarat pengaturan parameter terpenuhi. Akan tetapi pembahsan lebih lanjut mengenai penangan konvergensi prematur belum dilakukan.

Pencegahan konvergensi prematur telah dilakukan dengan berbagai cara, salah satunya adalah pengaturan parameter inersia. Pada PSO salah satu parameter yang berperan penting adalah inersia. Nilai parameter inersia yang besar menghasilkan pencarian global (eksplorasi) dan sebaliknya jika nilai inersia kecil akan menghasilkan pencarian lokal(eksploitasi). Ketika pengaturan parameter inersia yang tidak sesuai menyebabkan konvergensi prematur atau konvergensi

yang lambat. Hal tersebut terjadi karena tidak adanya keseimbangan antara pencarian global dan lokal [8]

Seiring perkembanganya modifikasi algoritma PSO telah dilakukan dengan memberikan algoritma perubahan parameter inersia secara dinamis untuk menyeimbangkan antara pencarian global maupun lokal. Algoritma inersia adaptif menjadi alternatif baru dalam PSO, setiap iterasinya proses pencarian termonitoring dan inersia berubah berdasarkan parameter umpanbalik. PSO inersia adaptif (AIWPSO) menggunakan *percentage of success* sebagai parameter umpan balik dan kemudian fungsi linier digunakan untuk pembobotan parameter inersia. Perubahan inersia secara adaptif tersebut membuat konvergensi menjadi lebih cepat dibandingkan dengan inersia konstan, *linear decreasing, linear increasing.* Akan tetapi konvergensi premature dalam permasalahan optimasi kompleks tetap sulit dihindari dengan metode ini. [8]

Selain parameter inersia, parameter lain dalam PSO yang perlu diatur adalah koefisien akselerasi. Parameter ini berfungsi untuk mengatur laju pembelajaran antara komponen kognitif dan sosial pada PSO. Selain diatur secara konstan penelitian mengenai pengaturan parameter ini adalah dirubah berdasarkan waktu iterasi atau *time varying acceleration coefficient*. Pengujian menunjukan dengan menggunakan metode pengaturan tersebut menghasilkan hasil optimasi yang lebih baik dibandingkan dengan PSO yang menggunakan pengaturan parameter inersia menggunakan *linear decreasing.* Hasil yang baik tersebut dicapai pada permasalahan permasalahan multi modal [9].

Oleh karena itu berdasarkan permasalahan diatas, maka pada penelitian ini permasalahan perencanan jalur *mobile robot* diformulasikan dengan menggunakan 3 buah fungsi tujuan agar representasi jalur lebih aman dari halangan dan lebih *smooth*. Pengembangan PSO adaptif dilakukan untuk mengatasi permasalahan konvergensi. Parameter inersia dan koeffisien akselerasi di atur menggunakan fungsi Gaussian untuk mempercepat konvergensi. Kemudian reinisalisasi partikel dilakukan untuk mengetahui apakah solusi yang dihasilkan adalah hasil yang global optimum. Selain itu untuk mempermudah permasalahan optimisasi dengan adanya halangan dinamis prediksi trrajektori halangan dilakukan agar perencanaan jalur dinamis dapat didekati dengan perencanaan jalur yang statis.

1.2 Rumusan Masalah

Rumusan masalah dari penelitian ini adalah sebagai berikut

1. Bagaimana memformulasikan permasalahan perencanaan jalur dalam fungsi tujuan optimasi yang mepertimbangkan panjang jalur, tingkat keaman terhadap tumbukan dan kriteria *smooth*
2. Bagaimana merancang suatu algoritma PSO yang dapat mempercepat konvergensi dan mengatasi permasalahan lokal minimum agar dihasilkan jalur yang optimal dengan panjang lintasan terpendek, aman dari tumbukan dan *smooth.*

1.3 Tujuan

Adapun tujuan penelitian ini adalah sebagai berikut :

1. Merancang model permasalahan optimasi perencanaan jalur yang mepertimbangkan panjang jalur, tingkat keaman terhadap tumbukan dan kriteria *smooth* pada jalur
2. Menghasilkan algoritma PSO yang dapat menangani permasalahan lokal optimum dan konvergensi premature pada implementasi perencanaan jalur mobile robot sehingga dihasilkan jalur yang terpendek, aman dari tumbukan dan *smooth.*.

1.4 Batasan Masalah

Pada penelitian ini terdapat batasan masalah, yaitu:

1. Analisa PSO dilakukan secara deterministic
2. Kecepatan dan percepatan halangan konstan
3. Path yang dihasilkan tidak memperhatikan orientasi dan sistem dinamika robot.
4. Lingkuan robot diketahui secara tepat

1.5 Kontribusi

Kontribusi dari penelitian ini adalah menghasilkan algoritma PSO yang dapat menangani permasalahan lokal optimum dan dapat mempercepat konvergensi

1.6 Metodologi Penelitian

Adapun metodologi yang digunakan dalam penelitian guna tercapainya tujuan penelitian adalah sebagai berikut:

1.6.1 Studi Literatur

Untuk menunjang pengerjaan penelitian ini adapun hal hal yang mendukung adalah konsep PSO, relasi rekursif homogen dan non homogen, dan optimasi multi tujuan.

1.6.2 Pemodelan Permasalan Perencanaan Jalur *Mobile Robot*

Pemodelan dilakukan dengan terlebih dahulu melakukan transformasi posisi dan lingkungan robot dalam koordinat lokal robot dan memformulasikanya menjadi tiga buah fungsi tujuan yaitu panjang lintasan, tingkat resiko tumbukan dan kriteria *smooth.*

1.6.3 Analisa dan Perancangan Sistem

Pada perancangan sistem terlebih dahulu PSO dianalisa dengan asusmsi proses pencarian adalah deterministik. Setelah itu dirancang algoritma PSO adaptif untuk mencegah terjadinya konvergensi premature dan konvergensi lambat. Kemudian merancang algoritma path planning menggunakan PSO adaptif.

1.6.4 Simulasi dan Analisa Performa Algoritma

Performa algoritma PSO dan permodelan permaslahan perencanaan jalur disimulasikan kemudian dibandingkan dengan PSO standart dan *Adaptive Inertia* (AIW) PSO pada beberapa skenario lingkungan robot.

1.6.5 Penarikan Kesimpulan

Kesimpulan diambil berdasarkan data pengujian dari perbandingan metode yang diusulkan dengan PSO standart dan AIW PSO dengan 3 kriteria yaitu jarak minimum yang dicapai dan jarak rata-rata yang dicapai, tingkat keamanan terhadap tumbukan, dan tingkat *smoothness* dari jalur yang dihasilkan

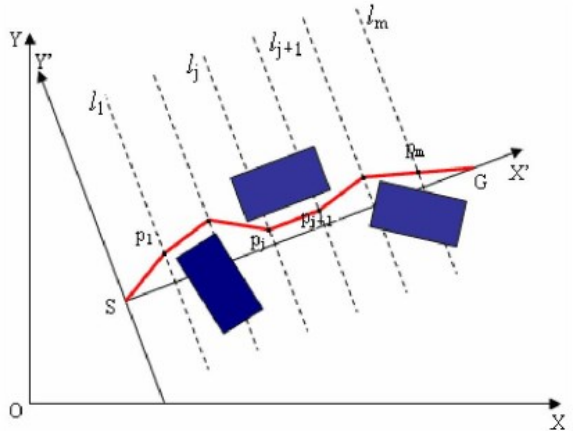
Halaman ini sengaja dikosongkan

BAB 2
KAJIAN PUSTAKA

2.1 Kajian Penelitian Terkait

2.1.1 Obstacle-avoidance Path Planning for Soccer Robots Using Particle Swarm Optimization

Penelitian ini menyajikan implementasi PSO dalam permasalahan perencanaan jalur dinamis yang mempertimbangkan bentuk dari halangan. Semua bentuk halangan direpresentasikan memiliki bentuk lingkaran. Pada mulanya semua posisi halangan ditransformasikan pada koordinat lokal seperti Gambar 2.1



Gambar 2.1 Model Representasi Jalur dengan Transofrmasi Koordinat Lokal [4]

Koordinat lokal dibentuk berdasarkan titik start S dan target G robot. Kemudian garis lurus antara SG dijadikan sebagai sumbu X' dan garis tegak lurus terhadap SG dijadikan sebagai sumbu Y'. Pada garis SG kemudian dibagi menjadi beberapa segmen dimana setiap segmen dibagi secara tetap. Penggunaan metode seperti ini dimaksudkan untuk membuat variabel keputusan optimisasi hanya bergantung pada sumbu Y'.

Melalui transformasi tersebut, pada penelitian ini kemudian membuat dua buah fungsi tujuan. Fungsi yang pertama adalah fungsi panjang jalur yang terbentuk melalui titik titik yang terbentuk dalam koordinat lokal dengan titik X' adalah konstan. Fungsi panjang jalur dijabarkan dalam persamaan berikut

$$f_1 = \sum_{j=0}^{m} \sqrt{\left(x'_{pj} - x'_{pj+1}\right)^2 + \left(y'_{pj} - y'_{pj+1}\right)^2} \quad (2.1)$$

Fungsi yang kedua merepresentasikan fungsi untuk menghindari halangan yang di jabarkan dalam persamaan (2.2)

$$f_2 = \begin{cases} 1 & L_{min} \geq R_k (k = 1,2,3, \dots, n) \\ 0 & else \end{cases} \quad (2.2)$$

dimana R_k adalah jari-jari halangan pada halangan ke-k dan L_{min} adalah jarak terdekat antara titik p_j dengan halangan ke-k yang dijabarkan dalam persamaan (2.3)

$$L_{min} = \min(\sqrt{\left(x'_{pj}(t) - x'_k(t)\right)^2 + \left(y'_{pj}(t) - y'_k(t)\right)^2} \quad (2.3)$$

Kemudian PSO digunakan untuk meminimumkan fungsi tujuan keseluruhan yang dijabarkan dalam persamaan (2.4)

$$f = f_1 \times f_2 \quad (2.4)$$

Dari 50 kali pengujian didapatkan 2 kali terjebak dalam lokal minimum yang mengakibatkan robot gsgsl menghindari halangan. Jika diperhatikan pada fungsi menghindari halangan pada persamaan (2.2) jika jarak antara halangan dengan titik jalur p_j kurang dari jarak aman atau jari jari R_k fungsi diberi bobot 0. Hal tersebut membuat algoritma memutuskan bahwa jalur yang bertumbukan dengan halangan dianggap paling minimum Karena diberi bobot 0.

2.1.2 A Convergence Proof for The Particle Swarm Optimization

Penelitian ini menyajikan Analisa konvergensi PSO secara analitis. Analisa dilakukan dengan mengasumsikan bahwa pergerakan partikel adalah deterministik dan dianalisa pada satu buah partikel saja. Telah kita ketahui sebelumnya bahwa PSO memiliki mekanisme perubahan kecepatan dan posisi partikel yang dijabarkan pada persamaan (2.5) dan (2.6)

$$V(t+1) = wV(t) + c_1 r_1 \left(P - x(t)\right) + c_2 r_2 \left(G - x(t)\right) \quad (2.5)$$

$$x(t+1) = x(t) + V(t+1) \quad (2.6)$$

Substitusi persamaan (2.5) ke persamaan (2.6) akan menghasilkan

$$x(t+1) = (1 - \varphi_1 - \varphi_2)x(t) + wV(t) + \varphi_1 P + \varphi_2 G \quad (2.7)$$

kemudian dari persamaan (2.6) didapatkan bahwa

$$V(t) = x(t) - x(t-1) \tag{2.8}$$

Substitusi persamaan (2.8) ke dalam persamaan (2.7), menghasilkan persamaan relasi rekurensi homogen

$$x(t+1) = (1 + w - \varphi_1 - \varphi_2)x(t) + wx(t-1) + \varphi_1 P + \varphi_2 G \tag{2.9}$$

yang memiliki persamaan karakteristik

$$(1-\lambda)(\omega - (1 + \omega - \emptyset_1 - \emptyset_2)\lambda + \lambda^2) = 0 \tag{2.10}$$

diengan nilai eigen dari persamaan tersebut adalah

$$\lambda_1 = 1 \tag{2.11}$$

$$\lambda_{2,3} = \frac{(1 + \omega - \emptyset_1 - \emptyset_2) \pm \gamma}{2} \tag{2.12}$$

dimana

$$\gamma = \sqrt{(1 + \omega - \emptyset_1 - \emptyset_2)^2 + 4\omega} \tag{2.13}$$

Setelah didapatkan nilai eigen, bentuk solusi persamaan relasi rekursif non-homogen pada persamaan (2.7) adalah sebagai berikut

$$x_t = K_1\lambda_1^t + K_2\lambda_2^t + K_3\lambda_3^t \tag{2.14}$$

dengan $\lambda_1 = 1$ maka

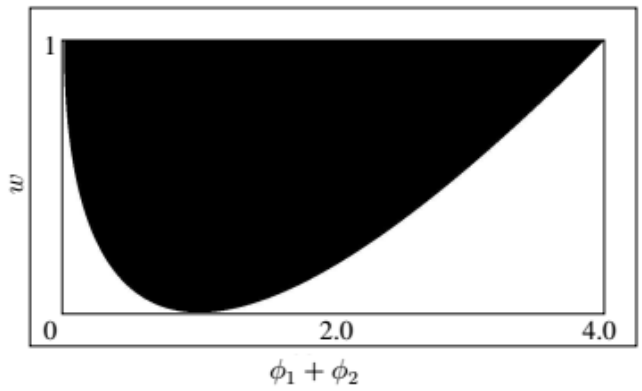
$$x_t = K_1 + K_2\lambda_2^t + K_3\lambda_3^t \tag{2.15}$$

Dari persamaan (2.15) didapatkan bahwa PSO akan konvergen jika dan hanya jika

$$\max(\|\lambda_2\|, \|\lambda_3\|) < 1 \tag{2.16}$$

Ketika syarat tersebut terpenuhi maka

$$\begin{gathered} \lim_{t\to\infty} x_t = K_1 + K_2 \lim_{t\to\infty} \lambda_2^t + K_3 \lim_{t\to\infty} \lambda_3^t \\ \lim_{t\to\infty} x_t = K_1 + 0 + 0 = K_1 \end{gathered} \tag{2.17}$$



Gambar 2.2 Kurva area parameter konvergensi PSO [7]

dimana

$$K_1 = \frac{\emptyset_1 P + \emptyset_2 G}{\emptyset_1 + \emptyset_2} \qquad (2.18)$$

ketika kondisi inisial $x(0)$ dan $x(1)$ diketahui.

Pada penelitian ini syarat pada persamaan (2.16) dipenuhi jika dan hanya jika

$$0 < w \leq 1 \qquad (2.19)$$

$$0 < c_1 + c_2 \leq 4 \qquad (2.20)$$

Syarat tersebut adalah syarat pengaturan parameter PSO agar konvergen untuk mendapatkan solusi dari permasalahan optimisasi yang dapat ditunjukan pada Gambar 2.2

2.1.3 A Novel Particle Swarm Optimization Algorithm with Adaptive Inertia Weight

Parameter inertia pertamakali diperkenalkan oleh Shi dan Eberhart [10] untuk mengatur konvergensi PSO akibat kecepatan partikel pada itersai sebelumnya. Berawal dari itu banyak peneliti yang mengusulkan beberapa metode untuk mengatur parameter tersebut agar diperoleh hasil optimisasi yang effisien dan effektif. Pada penelitian ini dilakukan peninjauan ulang dari beberapa pengaturan parameter inertia diantaranya

2.1.3.1 Random

$$w = 0.5 + \frac{rand()}{2} \qquad (2.21)$$

2.1.3.2 Time-Variying

Linear decreasing

$$w = \frac{iter_{max} - iter}{iter_{max}}(w_{max} - w_{min}) + w_{min} \tag{2.22}$$

Non-linear decreasing

$$w = \frac{(iter_{max} - iter)^n}{(iter_{max})^n}(w_{max} - w_{min}) + w_{min} \tag{2.23}$$

Dimana

$iter_{max}$: jumlah itersai maksimum

$iter$: nilai iterasi saat ini

w_{max} : nilai inersia maksimum yang diinginkan

w_{min} : nilai inersia minimum yang diinginkan

n : derajat nonlinearitas yang diinginkan

2.1.3.3 Adaptive Inertia

Selain *random* dan *time varying,* metode lain yang digunakan dalam pengaturan parameter inersia dalah metode adaptive. Metode yang pertama adalah metode yang perubahan parameter inersia bergantung pada perubahan 2 buah factor yang diberi nama *speed factor* dan *aggregation factor*, dimana perubahan inersia dilakukan dengan persamaan

$$w = w_{initial} - \alpha\big(1 - h(t)\big) + \beta s \tag{2.24}$$

dimana

$$h = \left|\frac{\min(f(P)_{t-1}, f(P)_t)}{\max(f(P)_{t-1}, f(P)_t)}\right| \tag{2.25}$$

adalah *speed factor,* $f(P)_t$ adalah nilai fungsi terbaik saat iterasi t dan $t-1$ dan

$$h = \left|\frac{\min\big(f(G)_t, \bar{f}\big)}{\max\big(f(G)_t, \bar{f}\big)}\right| \tag{2.26}$$

adalah *aggregation factor.*

Metode adaptasi yang kedua adalah menggunakan rasio perbandingan antara *global position* dan rata rata *personal best position.*

$$w = 1.1 - \frac{G}{\bar{P}} \tag{2.27}$$

Selain melakukan *review* terhadap beberapa metode diatas pada penelitian ini mengusulkan metode adaptasi parameter inersia berdasarkan parameter umpan balik *percentage of success* atau presentasi dari kesuksesan disetiap iterasi. Metode tersebut diusulkan karena metode adaptasi sebelumnya menggunakan parameter umpan balik yang tidak sesuai dengan kondisi yang terjadi pada proses pencarian PSO. Parameter umpan balik *percentage of success* dijabarkan dalam persamaan (2.28)

$$PS = \frac{\sum_{i=0}^{n} SC_i}{n} \tag{2.28}$$

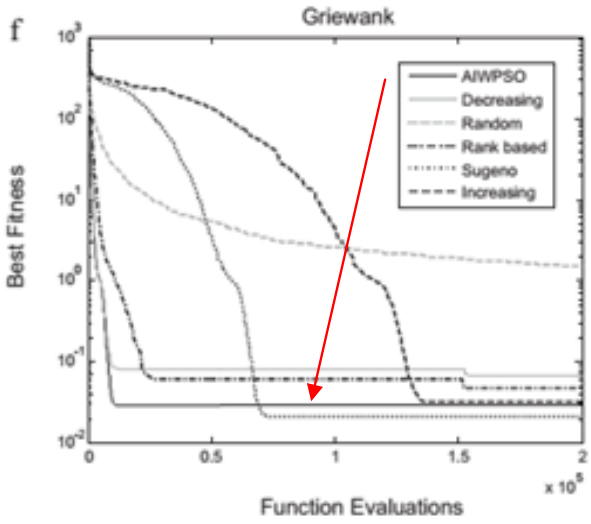
dimana

$$SC = \begin{cases} 1 & f\big(X(t)\big) < f\big(P(t-1)\big) \\ 0 & f\big(X(t)\big) \geq f\big(P(t-1)\big) \end{cases} \tag{2.29}$$

Kemudian parameter inersia diperbaharui dengan menggunakan fungsi linier berikut

$$w = (w_{max} - w_{min})PS(t) + w_{min} \tag{2.30}$$

Dari beberapa kali pengujian dengan menggunakan fungsi uji optimisasi maka didapatkan bahwa dengan menggunkana parameter tersebut membuat konvergensi lebih cepat dibanding dengan metode pengaturan inersia yang lainya seperti yang ditunjukan pada Gambar 2.3. Meskipun lebih cepat konvergensinya namun kemungkinan konvergensi premature tetap saja bisa terjadi seperti yang ditunjukan pada Gambar 2.3.



Gambar 2.3 Perbandingan nilai fungsi tujuan dari berbagai metode pengaturan parameter inersia [11]

2.1.4 Self-Hierarcical Particle Swarm Optimizer with Time-Varying Acceleration Coefficients

Selain konstan, awal mula penelitian mengenai pengaturan koefisien akselerasi pada PSO adalah parameter tersebut diberikan secara *time varying* yang terinspirasi dari pengaturan parameter inersia secara *linear decreasing*. Kedua parameter akselerasi di atur menggunakan persamaan berikut

$$c_1 = \left(c_{1_{max}} - c_{1_{min}}\right)\frac{iter}{iter_{max}} + c_{1_{min}}$$
$$c_2 = \left(c_{2_{min}} - c_{2_{max}}\right)\frac{iter}{iter_{max}} + c_{2_{max}} \quad (2.31)$$

Alasan menggunakan mekanisme tersebut adalah untuk menyeimbangkan antara social komponen dan kognitif komponen. Pada awal iterasi nilai c_1 lebih besar dari c_2 agar partikel tersebar berdasarkan kemampuan kognitifnya dan pada akhir iterasi c_2 lebih besar dari c_1 agar terjadi konvergensi yang lebih cepat pada komponen sosialnya. Percobaan secara empiris dengan menggunakan 5 fungsi uji optimisasi menunjukan bahwa metode yang diusulkan memiliki hasil yang lebih baik dibandingkan dengan metode pengaturan pearameter inersia secara *linear decreasing* seperti yang ditunjukan pada Tabel 2.1.

Tabel 2.1 Rata rata nilai dan standar deviasi dari nilai optimal pada 50 kali trial

Function	Dimension	$Iter_{max}$	Average (Standar Deviation)		
			PSO-TVIW	PSO-RandW	PSO-TVAC
f_1	10	1000	**0.01**	**0.01**	**0.01**
	20	2000	**0.01**	**0.01**	**0.01**
	30	3000	**0.01**	**0.01**	**0.01**
f_2	10	3000	27.11 (58.312)	**2.102 (3.218)**	9.946 (32.127)
	20	4000	51.56 (119.79)	28.1788 (73.072)	**17.944 (46.296)**
	30	5000	63.35 (71.210)	35.277 (55.751)	**28.97 (51.638)**
f_3	10	3000	**2.069 (1.152)**	4.63 (2.366)	2.268 (1.333)
	20	4000	**11.74 (3.673)**	26.293 (8.176)	15.323 (5.585)
	30	5000	**29.35 (6.578)**	69.7266 (20.700)	36.236 (8.133)
f_4	10	3000	0.0675 (0.029)	0.0661 (0.030)	**0.05454 (0.025)**
	20	4000	0.0288 (0.023)	**0.0272 (0.025)**	0.0293 (0.027)
	30	5000	**0.0167 (0.013)**	0.0175 (0.004)	0.0191 (0.015)
f_6	2	1000	0.0039 (0.0019)	**0.0029 (0.004)**	0.0039 (0.0019)

2.2 Dasar Teori

2.2.1 Standard Particle Swarm Optimization (SPSO)

Particle Swarm Optimization (PSO) adalah teknik optimasi berbasis populasi yang dikembangkan oleh James Kennedy dan Russ Eberhart pada tahun 1995. Teknik ini terinspirasi oleh tingkah laku sosial pada kawanan burung yang terbang berduyun-duyun (*bird flocking*) atau gerombolan ikan yang berenang berkelompok (*fish schooling*). Kawanan burung bangau, dalam jumlah sangat banyak, bisa terbang membentuk formasi tertentu tanpa bertabrakan satu sama lain. Gerombolan ikan yang berjumlah ribuan bisa bergerak sangat cepat tanpa tabrakan meskipun jarak antar ikan begitu dekat. Burung maupun ikan memiliki kecerdasan

yang luar biasa sehingga bisa menjaga jarak tetap stabil dengan mengatur kecepatan terbang atau berenangnya.

PSO dimulai dengan suatu populasi yang terdiri dari sejumlah individu (yang menyatakan solusi) yang dibangkitkan secara acak dan selanjutnya melakukan pencarian solusi optimum melalui perbaikan individu untuk sejumlah generasi tertentu. Pada PSO, posisi dan kecepatan terbang partikel di–*update* pada setiap iterasi sehingga pertikel tersebut bisa menghasilkan solusi baru yang lebih baik.

Pada PSO, solusi-solusi (partikel) yang potensial, "terbang" di dalam ruang masalah mengikuti partikel-partikel yang optimum saat ini (*current optimum particles*). Dengan konsep ini, PSO lebih mudah diimplementasikan dan parameter yang harus disetel hanya sedikit. PSO telah berhasil diaplikasikan pada banyak area: optimasi fungsi, pelatihan *artificial neural network* (ANN), *fuzzy system control*, dan area-area lainnya di mana GA dapat diaplikasikan.

A. Konsep Dasar

PSO dimulai dengan sekumpulan partikel (solusi) yang dibangkitkan secara acak. Setiap partikel kemudian dievaluasi kualitasnya menggunakan fungsi *fitness*. Selanjutnya. Partikel-partikel akan terbang mengikuti partikel yang optimum. Pada setiap generasi, setiap partikel di–*update* mengikuti dua nilai "terbaik". Yang pertama adalah *fitness* terbaik yang dicapai oleh satu partikel saat ini. Nilai *fitness* ini dilambangkan dengan *Pbest* dan disimpan di *memory.* Sedangkan nilai "terbaik" yang ke dua adalah *fitness* terbaik yang dicapai oleh semua partikel dalam topology ketetanggaan. Indeks *Gbest* digunakan untuk menunjuk partikel dengan *fitness* terbaik tersebut.

Jika kita menggunakan topology ketetanggaan yang berupa *ring topology*, maka cara ini disebut sebagai PSO versi global. Tetapi, jika topologi ketetanggaannya berupa *star topology* maka cara ini di sebut PSO versi lokal. Setelah menemukan dua nilai "terbaik", suatu partikel *i* pada posisi Xi meng-*update* vektor *velocity* dan kemudian meng-*update* posisinya menggunakan persamaan:

$$v_{id} = \omega_{id} * v_{id} + c_1 * r * (P_{id} - x_{id}) + c_2 * r * (G_d - x_{id}) \qquad (2.32)$$

$$x_{id} = x_{id} + v_{id} \qquad (2.33)$$

di mana

ω =inersia pada partikel ke-*i* (konstan)

i = partikel ke – *i*;

d = dimensi ke – *d*;

c_1= laju belajar (*learning rates*) untuk komponen ***cognition*** (kecerdasan individu) ;

c_2= laju belajar untuk komponen ***social*** (hubungan sosial antarindividu);

P= vektor nilai *fitness* terbaik yang dihasilkan sejauh ini;

g=indeks dari partikel dengan *fitness* terbaik di dalam topologi ketetanggaan

r= bilangan acak (random) dalam interval [0,1].

Velocity partikel pada setiap dimensi dibatasi pada statu *velocity* maksimum *Vmax*. Jika percepatan akan mengakibatkan *velocity* pada suatu dimensi melebihi *Vmax*, maka *velocity* pada dimensi tersebut dianggap sama dengan *Vmax*. Nilai batas *Vmax* ditentukan oleh *user*.

B. Parameter

PSO memiliki dua komponen penting: representasi solusi dan fungsi *fitness*. Setiap partikel yang merepresentasikan satu solusi dapat berupa bilangan real. Sebagai contoh, untuk memaksimasi fungsi $f(x) = x_1^3 + x_2^4 + x_3^2$ posisi partikel *i* bisa berupa $X_i = <x_1,x_2,x_3>$ .Sedangkan fungsi *fitness*-nya adalah *f (x)* itu sendiri. Setelah mendefinisikan ke dua komponen tersebut, maka kita bisa menggunakan algoritma PSO di atas untuk mencari nilai maksimum dari fungsi tersebut, Pencarian dilakukan secara iteratif sampai sejumlah iterasi tertentu atau tingkat kesalahan tertentu, yang diinginkan *user*, sudah dicapai. PSO memiliki beberapa parameter untuk diatur-atur (disetel), yaitu :

1. **Jumlah partikel**. Biasanya antara 20 sampai 40. Tetapi, untuk sebagian besar masalah, 10 partikel sudah cukup besar untuk mendapatkan hasil yang bagus. Untuk masalah khusus yang sangat sulit, kita bisa saja menggunakan 100 partikel atau lebih. Pada, Carlisle menyatakan bahwa partikel tidak terlalu berpengaruh terhadap solusi optimum yang dihasilkan PSO, tetapi berpengaruh terhadap kecepatan proses. Jumlah partikel yang terlalu kecil bisa terjebak pada optimum lokal meskipun waktu prosesnya sangat cepat.

Sebaliknya, jumlah partikel yang besar jarang terjebak pada optimum lokal tetapi waktu prosesnya lebih lama. Carlisle menyarankan jumlah partikel sekitar 30, cukup kecil untuk efisiensi waktu dan sudah cukup besar untuk menghasilkan solusi yang baik (mendekati optimum global).

2. **Dimensi partikel**. Hal ini bergantung pada masalah yang akan dioptimasi.
3. **Rentang nilai dari partikel**. Hal ini juga bergantung pada masalah yang akan dioptimasi.
4. ***Vmax***. Variabel ini menentukan perubahan maksimum yang bisa dilakukan oleh suatu partikel dalam satu iterasi. Biasanya Vmax diset sama dengan rentang nilai partikel. Misalkan, suatu partikel *i* yang berada pada posisi $X_i = \langle x_1, x_2, x_3 \rangle$ dengan x_1 berada dalam rentang [-10,10], x_2 dalam rentang [-5,5], dan x_3 dalam rentang [-3,3]. Untuk partikel tersebut, kita bisa menggunakan Vmax untuk x_1 sebesar 20 (didapat dari 10- (-10), Vmax untuk x_2 sebesar 10, dan Vmax x_3 sebesar 6, Carlisle menemukan bahwa untuk permasalahan dengan batasan $[X_{min}, X_{max}]$, ketika partikel mencapai X_{max} maka ubah *velocity*nya menjadi 0.
5. ***Synchronous*** atau ***asynchronous update.*** Pada *Synchronous update*, semua partikel digerakkan secara paralel kemudian partikel terbaik di dalam topologi ketetanggaan dipilih dan iterasi berikutnya dijalakan. Sedangkan pada *asynchronous update*, partikel terbaik di dalam topologi ketetanggaan dipilih lebih dulu, kemudian partikel terbaik tersebut diperhitungkan untuk menggerakkan semua partikel lainnya. Carlisle menemukan bahwa *asynchronous update* lebih efisien dibandingkan *Synchronous update*.

2.2.2 Relasi Rekrusif Linier

Relasi Rekursif adalah sebuah fungsi deret (a_n) yang mana nilai nilai a_n bergantung pada nilai yang sebelumnya $(a_{n-1}, a_{n-2}, \dots, a_1, a_0)$. Berdasarkan bentuk fungsinya relasi rekursif dibedakan menjadi bentuk liner dan non linier serta dalam bentuk homogen dan non homogen. Relasi rekursif biasanya digunakan dalam menyelesaikan masalah *counting.*

A. *Relasi Rekursif Linier Homogen*

Relasi rekurensi jenis ini merupakan relasi rekurensi yang mengekspresikan suku - suku barisan sebagai kombinasi linier suku-suku sebelumnya. Sebuah persamaan relasi rekursif linier dengan derajat k serta memiliki koefisien konstan dinyatakan dalam bentuk persamaan sebagai berikut:

$$a_n = c_1 a_{n-1} + c_2 a_{n-2} + \cdots + c_k a_{n-k} \quad (2.34)$$

Bentuk tersebut linier jika tidak ada perkalian atau perpangkatan dari a_j. Kemudian yang dimaksud koeffisien konstan adalah ketika $c_1, c_2, \ldots, c_k$ tidak bergantung pada n. Sedangkan degree atau orde k terjadi jika a_n diekspresikan dalam k suku sebelumnya.

- **Solusi Relasi Rekursif Linier Homogen**

Bentuk dasar penyelesaian dari sebuah persamaan relasi rekursif linier adalah $a_n = r^n$, dimana r adalah sebuah konstatnta. Penyelesaian r^n menjadi solusi relasi rekursif persamaan (2.34) jika dan hanya jika

$$r^n = c_1 r^{n-1} + c_2 r^{n-2} + \cdots + c_k r^{n-k} \; ( ) \quad (2.35)$$

Ketika persamaan (2.35) dibagi dengan r^{n-k} pada bagian kiri dan kanan menjadi:

$$r^k - c_1 r^{k-1} - c_2 r^{k-2} - \cdots - c_{k-1} r - c_k = 0 \quad (2.36)$$

Persamaan tersebut adalah persamaan karakteristik dari sebuah relasi rekursif linier homogen. Penyelesaian dari persamaan karakteristik tersebut disebut akar akar karakteristik yang merupakan solusi eksplisit dari sebuah relasi rekursif linier homogen.

B. Relasi Rekursif Linier Non-Homogen

Bentuk umum dari relasi rekursif non-homogen adalah sebagai berikut:

$$a_n = a_n = c_1 a_{n-1} + c_2 a_{n-2} + \cdots + c_k a_{n-k} + F(n) \quad (2.37)$$

Bentuk umum tersebut bersifat non-homogen jika dan hanya jika $F(n)$ adalah fungsi yang tidak sama dengan nol yang nilanya bergantung pada n. Relasi rekurensi

$$a_n = a_n = c_1 a_{n-1} + c_2 a_{n-2} + \cdots + c_k a_{n-k} \quad (2.38)$$

disebut relasi yang berasosiasi dengan relasi rekursif homogen.

- **Solusi Relasi Rekursif Linier Non-Homogen**

Bentuk umum solusi dari relasi rekursif linier non-homogen terdiri dari dua bagian yaitu solusi homogen dan solusi khusus. Solusi homogeny diperoleh dari persamaan relasi yang berasosiasi dengan rekursif linier homogen. Sedangkan solusi khususnya adalah solusi penyelesaian yang mungkin dari fungsi $F(n)$

Halaman ini sengaja dikosongkan

BAB 3
PERANCANGAN SISTEM

Bab ini membahas tahapan-tahapan yang dilakukan dalam proses perancangan sistem. Proses perancangan yang dilakukan meliputi proses perancangan algoritma PSO adaptif dan pemodelan permasalahan perencanaan jalur mobile robot.

3.1 Pemodelan Perencanaan Jalur Mobile Robot

3.1.1 Representasi Jalur Dalam Lingkungan Robot

Pada bab ini pembahasan tentang bagaimana memodelkan jalur dalam lingkungan robot serta merumuskan fungsi tujuan optimasi. pada permodelan jalur dalam lingkungan robot metode dari [4] diadopsi karena sederhana. Semua obstacle direpresentasikan dalam bentuk lingkaran untuk memudahkan dalam formulasi fungsi tujuan. Selain itu semua posisi dan titik titik jalur ditransormasikan dalam koordinat local untuk memperkecil jumlah variable keputusan.

Pada koordinat global O-XY, Sdan T merepresentasikan posisi start robot dan target robot. Garis ST direpresentasikan sebagai X' *axis* untuk membuat koordinat lokal S-$X'Y'$, kemudian garis tersebut dibagi menjadi $n+1$ segmen. Pada setiap segmen dibuat garis bayangan yang tegak lurus sejumlah n sehingga kita dapatkan satu set garis parallel $(l_1, l_2, \dots, l_n)$ seperti yang ditunjukan oleh Gambar 3.1. Dengan memberikan titik titik acak pada garis vertikal $(l_1, l_2, \dots, l_n)$, kita dapat membentuk jalur robot secara keseluruhan $(pl_1, pl_2, \dots, pl_n)$. Kemudian untuk mentransormasikan posisi pada koordinat global O-XY dalam koordinat lokal S-$X'Y'$ dilakukan dengan persamaan transformasi berikut

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} x_s \\ y_s \end{bmatrix} \tag{3.1}$$

dimana:

(x_s, y_s) : posisi start S dalam koordinat global.

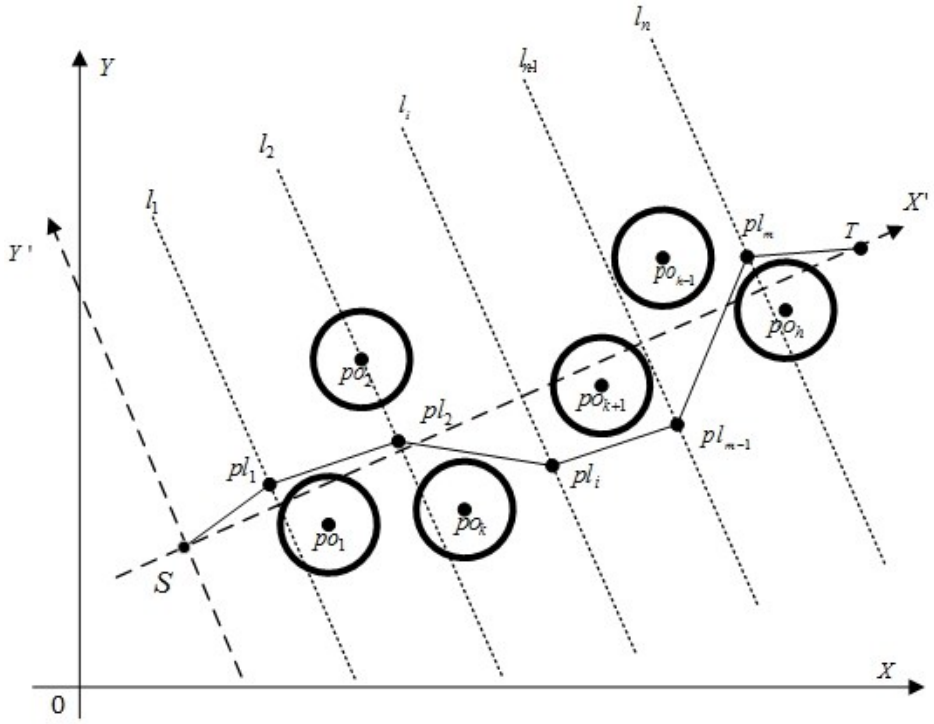
(x', y') : posisi titik (x, y) dalam koordinat lokal

θ : adalah sudut antara X-axis dan garis ST

Melalui transmormasi tersebut permasalahan perencanaan jalur mobile robot menjadi permasalahan optimasi, dimana variable yang dioptimasi adlah sebagai berikut

$$PL = (S, pl_1, pl_2, \dots, pl_n, T) \tag{3.2}$$

dimana setiap titik pl_n tidak tertutupi oleh halangan, dan setiap garis diantara pasang titik (S- pl_1,pl_1- pl_2,…,pl_{m-1}- pl_m, pl_m- T) dapat memenuhi batasan tidak bertumbukan dengan halangan.



Gambar 3.1. Representasi path dalam lingkungan robot

3.1.2 Fungsi Tujuan

Permasalahan utama pada permasalahan perencanaan jalur mobile robot adalah bagaimana membangkitkan jalur yang terbebas dari tumbukan. Pada tesis ini, terdapat tiga fungsi tujuan yang digunakan untuk merepresentasikan permasalahan tersebut, fungsi tujuan tersebut antara lain, panjang jalur, tingkat resiko tumbukan, dan tingkat *smooth* pada jalur.

3.1.2.1 Panjang Jalur

Fungsi tujuan yang pertama adalah panjang jalur, pada sub bab sebelumnya sudah dijelaskan bagaimana mereperesentasikan jalur dalam

lingkukngan robot. Jalur robot dibentuk melalui titik titik $(S, pl_1, pl_2, \dots, pl_n, T)$, sehingga panjang jalur PL didapatkan melalui persamaan

$$L(PL) = \sum_{j=0}^{n} d(pl_j, pl_{j+1}) \tag{3.3}$$

dimana $d(pl_j, pl_{j+1})$ merepresentasikan panjang antara pl_j dan pl_{j+1}.

$$d(pl_j, pl_{j+1}) = \sqrt{\left(x'_{ph_j} - x'_{ph_j+1}\right)^2 + \left(y'_{ph_j} - y'_{ph_j+1}\right)^2} \tag{3.4}$$

Pada koordinat lokal, garis ST telah dibagi menjadi $n+1$ segmen, sehingg panjang jalur dari setiap titik pada persamaan (3.4) dapat dihitung melalui persamaan

$$d(pl_j, pl_{j+1}) = \sqrt{\left(\frac{d(pl_0, pl_{n+1})}{n+1}\right)^2 + \left(y'_{ph_j} - y'_{ph_j+1}\right)^2} \tag{3.5}$$

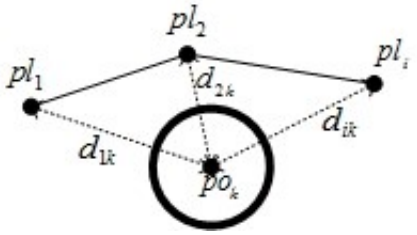
dari persamaan (3.5) panjang jalur PL keseluruhan dapat dihitung dengan persamaan

$$J_{dis} = \sum_{j=0}^{m} \sqrt{\left(\frac{d(pl_0, pl_{n+1})}{n+1}\right)^2 + \left(y'_{ph_j} - y'_{ph_j+1}\right)^2} \tag{3.6}$$

dimana $d(pl_0, pl_{n+1})$ adalah panjang antara start dan target.

3.1.2.2 Tingkat Resiko Tumbukan

Fungsi tujuan selanjutnya adalah tingkat resiko tumbukan dengan halangan. Pada fungsi tujuan ini tingkat resiko tumbukan direpresentasikan dengan jarak setiap titik jalur dengan titik pusat halangan seperti yang ditunjukan oleh Gambar 3.2. pada gambar tersebut $(pl_1, pl_2, \dots, pl_i)$ merupakan titik titik jalu dan po_k adalah titik pusat halangan ke-k.



Gambar 3.2 Representasi jarak halangan dengan titik jalur

Kemudian fungsi tingkat resiko tumbukan dengan halangan direpersentasikan dalam persamaan (3.7).

$$J_{risk} = \sum_{j=0}^{n}\sum_{k=0}^{m}\left(\frac{sign(r - d_{jk}) + 1}{2}\right)\left((r - d_{jk})\alpha\right) + \left(\frac{sign(d_{jk} - r) + 1}{2}\right)0 \tag{3.7}$$

dimana;

r : adalah jarak aman antara jalur dengan halangan

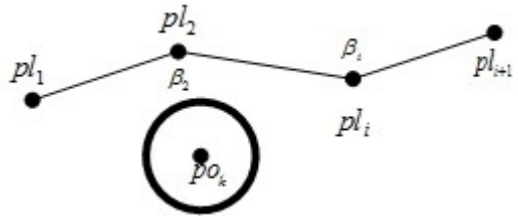
d_{ik} : adalah jarak antar titik jalur ke-i dengan halangan ke-k

α : adalah bobot nilai konstan yang lebih besar dari 0.

Pada fungsi tujuan tersebut, ketika jarak d_{ik} bertumbukan dengan halangan akan diberikan bobot besar sesuai dengan jaraknya dengan titik pusat halangan. Selanjutnya semakin jauh dengan jarak aman r juga akan diberikan bobot yang besar, sehingga dari fungsi tujuan tersebut diharapkan jalur tetap berada di jarak aman.

3.1.2.3 Kriteria smooth jalur

Selain dua fungsi tujuan yang telah dijabarkan pada tesis ini ditambahkan satu fungsi tujuan yaitu kriteria $smooth$ pada jalur. Fungsi kriteria $smooth$ diberikan dengan tujuan agar jalur yang terbentuk tidak memiliki sudut sdut lengkung yang tajam atau sudut lengkung yang kurang dari 90^0. Oleh karena itu, kriteria $smooth$ didapatkan melalui sudut sudut yang dibentuk oleh masing masing garis pada titik titik jalur, seperti yang ditunjukan pada Gambar 3.3.



Gambar 3.3 representasi jalur dan sudut ditiap titik jalur

Kriteria *smooth* direpresentasikan dalam persamaan

$$J_{smooth} = \sum_{j=0}^{m} (\vartheta - \beta_j) \tag{3.8}$$

dimana

β_j : sudut antara titik jalur $(pl_{i-1}, pl_i, pl_{i+1})$

ϑ : sudut yang diinginkan

Pada pemodelan fungsi tujuan didapatkan bahwa permasalahan perencanaan jalur mobile robot merupakan permaslahan multi tujuan. Untuk mempermudah dalam proses optimasi maka fungsi tujuan pada persamaan (3.6),(3.7), dan (3.8) dibuat dalam bentuk penjumlahan berbobot berikut

$$J = \gamma_1 J_{dis} + \gamma_2 J_{risk} + \gamma_3 J_{smooth} \tag{3.9}$$

dimana $\gamma_{1,2,3}$ adalah konstanta yang memiliki nilai antara [0,1].

3.2 Perancangan Algoritma Adaptive Parameter PSO

Pada sub bab ini akan dibahas tentang perancangan algoritma adaptif Gaussian PSO. Melalui Analisa konvergensi PSO didapatkan bagaimana mengatur parameter parameter yang ada dalam PSO.

3.2.1 Analisa Prilaku Kecepatan PSO

Pada sub bab ini Analisa konvergensi dilakukan untuk mengetahui karakteristik PSO. Analisa konvergensi dilakukan dengan menganalisa kecepatan partikel dengan asumsi bahwa pergerakan partikel adalah deterministik dan dalam waktu diskrit seperti yang dilakukan oleh [7]. Substitusi persamaan posisi pada persamaan (2.33) kedalam persamaan kecepatan (2.32) didapatkan persaman relasi rekurensi non-homogen (Lampiran 1)

$$V_i(t+1) = (1 + w - \varphi_1 - \varphi_2)V_i(t) - wV_i(t-1) \tag{3.10}$$

dimana $\varphi_1 = c_1 r_1$ dan $\varphi_2 = c_2 r_2$. Dari persamaan (3.10) kita dapat membentuk persamaan linear

$$\begin{bmatrix} V_i(t+1) \\ V_i(t) \end{bmatrix} = \begin{bmatrix} (1 + w - \varphi_1 - \varphi_2) & -w \\ 1 & 0 \end{bmatrix} \begin{bmatrix} V_i(t) \\ V_i(t-1) \end{bmatrix} \tag{3.11}$$

persamaan karakteristik dari sistem tersebut adalah sebagai berikut

$$\lambda_{1,2} = \frac{(1 + w - \varphi_1 - \varphi_2) \pm \gamma}{2} \tag{3.12}$$

diamana $\gamma = \sqrt{(1 + w - \varphi_1 - \varphi_2)^2 - 4w}$. Dari persamaan (3.12), solusi dari persamaan relasi rekursif non-homogen (3.10) adalah

$$v_i(t) = P_1\lambda_1^t + P_2\lambda_2^t. \tag{3.13}$$

Pada persamaan (3.13), trajektori partikel akan konvergen jika dan hanya jika syarat $max\{\|\lambda_1\|, \|\lambda_2\|\} < 1$ terpenuhi, sehingga dengan kondisi terseebut didapatkan bahwa

$$\lim_{t\to\infty} v_i(t) = 0 \tag{3.14}$$

Dari hasil tersebut dapat disimpulkan bahwa ketika syarat konvergensi terpenuhi PSO akan konvergen dengan kecepatan partikel menuju ke nol. Konvergensi premature terjadi ketika partikel konvergen dan kecepatan partikel mendekati nol, namun solusi global belum diketemukan.

3.2.2 Re-inisialisasi Partikel

Pada subab sebelumnya kita telah menganalisa kecepatan partikel dari PSO. Berdasarkan persamaan (3.14), kecepatan partikel akan menuju ke nol saat partikel konvergen meskipun solusi global belum diketemukan. Untuk mencegah terjadinya kondisi tersebut, reinisialisasi partikel dilakukan ketika rata rata kecepatan dari semua partikel mendekati nol.

Proses reinisialisai tersbut dapat dijabarkan melalui pernyataan berikut

$$x_{id}(t) = \begin{cases} G_d(t) + (r_3 - r_4)r & |\overline{V_{id}}| < \gamma \\ x_{id}(t) & else \end{cases} \tag{3.15}$$

dimana :

$G_d(t)$: indeks posisi partikel terbaik dalam *swarm*

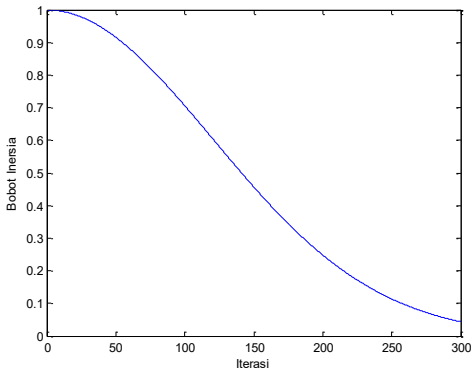
r_3, r_4 : *random number* dengan distribusi normal [0,1]

r : adalah radius penebaran partikel

Melalui reinisialisasi partikel diharapkan algoritma dapat keluar dari lokal minimum yang mengakibatkan konvergensi premature.

3.2.3 Parameter Inersia

Fungsi utama dari parameter inersia adalah untuk menjaga keseimbangan antara eksplorasi dan eksploitasi. Nilai inertia yang relatif besar akan cenderung membuat pola pencarian lebih eksploratif dan sebaliknya. Salah satu teknik untuk mencegah terjadinya konvergensi premature adalah dengan mengatur parameter inersia secara adaptif sesuai dengan dinamika yang terjadi pada saat pencarian. Pada tesis ini parameter umpan balik *Percentage of Successful* (PS) pada persaman (2.28) di adopsi karena parameter tersebut menggambarkan apa yang terjadi pada proses pencarian pada algoritma PSO.



Gambar 3.4 Bobot arameter inersia dengan menggunakan fungsi gausssian

Kemudian, parameter inersia diadaptasi dengan menggunakan fungsi Gaussian berikut

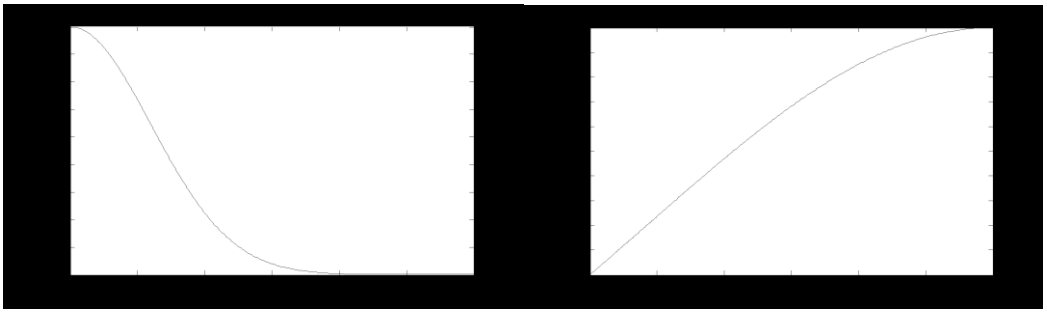
$$w = \left(\exp\left(-\frac{1}{2}\left(\frac{it - maxit}{0.5\, maxit}\right)^2\right)\right) PS \tag{3.16}$$

Dengan menggunakan persamaan (3.16), nilai inersia akan terdistribusi seperti yang ditunjukan oleh Gambar 3.4. Pemberian bobot dengan distribusi tersebut bertujuan agar pola pencarian lebih ekspolratif.

3.2.4 Koeffisien Akselerasi (C_1 & C_2)

Terdapat tiga component dalam persamaan kecepatan PSO yaitu, komponen kecepatan waktu yang lalu, komponen kognitif $\left(P - x(t)\right)$ dan komponen sosial $\left(G - x(t)\right)$. Koeffisien akselerasi C_1 dan C_2 berfungsi mengatur keseimbangan antara komponen kognitif dan komponen social. Jika nilai C_1 lebih

besar dibandingkan dengan C_2 atau komponen kognitif lebih besar dibanding dengan komponen social maka partikel akan mengeksplorasi dalam ruang pencarian. Sebaliknya, jika nilai C_2 lebih besar dibandingkan dengan C_1 atau komponen sosial lebih besar dibanding dengan komponen kognitif maka akan mempercepat konvergensi. Berdasarkan pernyataan tersebut maka untuk menyeimbangkannya mekanisme pemberian kedua bobot tersbut menggunakan fungsi Gaussian berikut ini



Gambar 3.5 (a) nilai C_1 pada setiap iterasi (b) nilai C_2 pada setiap iterasi

$$c_1 = \left(c_{1_{max}} - c_{1_{min}}\right) \exp\left(-0.5\left(\frac{t}{0.2t_{max}}\right)^2\right) + c_{1_{min}}$$
$$c_2 = \left(c_{2_{max}} - c_{2_{min}}\right) \exp\left(-0.5\left(\frac{t - t_{max}}{0.9t_{max}}\right)^2\right) + c_{2_{min}} \quad (3.17)$$

Dimana :

t : waktu iterasi

t_{max} : waktu iterasi maksimal

$0 \leq c_{1_{min}}, c_{2_{min}} \leq c_{1_{max}}, c_{2_{max}} \leq 2$

3.3 Implementasi Algoritma PSO Pada Perencanaan Jalur Mobile Robot

3.3.1 Representasi Partikel

Pada sub bab 3.1.1, telah dijabarkan bagaimana merepresentasikan jalur PL dalam lingkungan robot dengan titik titik $(pl_1, pl_2, \dots, pl_m)$. Pada titik titik tersebut, sejak garis bayangan $(l_1, l_2, \dots, l_n)$ diberikan untuk membagi sumbu X' dalam koordinat lokal dalam beberapa segmen secara konstan, maka titik titik jalur $(pl_1, pl_2, \dots, pl_m)$ hanya bergantung pada titik titik pada garis $(l_1, l_2, \dots, l_n)$ pada

sumbu Y' koordinat lokal. Sehingga untuk mendapatkan jalur yang optimal, variabel keputusan $\left(y'_{pl_1}, y'_{pl_2}, \dots, y'_{pl_m}\right)$ dijadikan sebagai partikel.

$$x_d = (x_1, x_2, x_3, \dots, x_d) = \left(y'_{pl_1}, y'_{pl_2}, \dots, y'_{pl_m}\right) \tag{3.18}$$

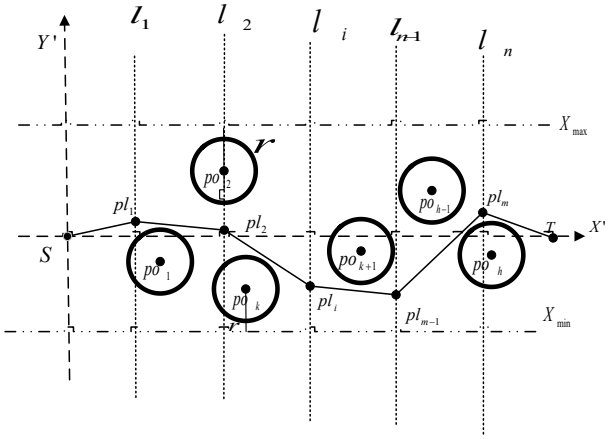
Setelah semua posisi di transformasikan dalam koordinat lokal dan variabel keputusan bergantung pada sumbu Y', untuk membuat pencarian effektif dan efisien ruang pencarian partikel dibatasi oleh batas atas dan batas bawah. Dimana batas atas adalah posisi halangan tertinggi pada sumbu Y' dan batas bawah adalah posisi halangan terendah halangan pada sumbu Y'.

$$X_{min} \leq X_{id} \leq X_{max} \tag{3.19}$$

dimana

X_{min} : $\min(y'_{o_1}, y'_{o_2}, \dots, y'_{o_k}) - r$

X_{max} : $\max(y'_{o_1}, y'_{o_2}, \dots, y'_{o_k}) + r$



Gambar 3.6 Batasan ruang pencarian

3.3.2 Pemilihan Personal best position (P_{id}) dan Global best position (G_d)

Personal best position (P_{id}) adalah indeks dari posisi partikel yang menghasilkan nilai fungsi tujuan terbaik dan Global best position (G_d) adalah indeks dari *swarm* partikel yang menghasilkan nilai fungsi tujuan terbaik. Sejak permasalahan perencanaan jalur diformulasikan menjadi persoalaan minimisasi fungsi tujuan persamaan (3.9), sehingga Personal best position (P_{id}) dan global best position (G_d) di perbaharui dengan memilih partikel yang mendominasi atau yang

menghasilkan nilai fungsi tujuan yang kecil. Mekanisme pembaharuan personal best position (P_{id}) dan global best position (G_d) tersebut dideskripsikan menjadi

$$P_{id}(t) = \begin{cases} X_{id}(t) & J\big(X_{id}(t)\big) < J\big(P_{id}(t-1)\big) \\ P_{id}(t-1) & J\big(X_{id}(t)\big) \geq J\big(P_{id}(t-1)\big) \end{cases} \quad (3.20)$$

dan global best position (G_d) didapatkan dari

$$G_d(t) = \min\Big(J\big(P_1(t)\big), J\big(P_2(t)\big), \dots, J\big(P_d(t)\big)\Big) \quad (3.21)$$

Secara keselruhan algoritma PSO dijabarkan dalam *pseudocode* pada Gambar 3.7.

Start
Transformasi posisi dalam koordinat global ke dalam koordinat S-X'Y';
Membuat n segment $(l_1, l_2, \dots, l_n)$ *pada X'-axis;*
For i=1 hingga n
Inisialisasi partikel sebagai titik y'pada $(l_1, l_2, \dots, l_n)$ *;*
$P_i = x_i$ *;*
End
While (kondisi iterasi belum terpenuhi)
SC=0;
For i=1 hingga n partikel
For d=1 hingga d dimens
Hitung kecepatan partikel dengan persamaan (2.22)*;*
Hitung posisi partikel dengan persamaan (2.23)*;*
Evaluasi partikel dengan menggunakan fungsi tujuan persamaan (3.9)*;*
End
End
Perbaharui P_{id} *and* G_d *menggunakan persamaan* (3.20) *dan* (3.21)*;*
Perbaharui Parameter inersia menggunakan persamaan (3.16)*;*
Perbaharui Parameter akselerasi menggunakan persamaan (3.17)*;*
Evaluasi kondisi reinisialisasi partikel menggunakan persamaan (3.15)*;*
End
$\left(y'_1, y'_2, \dots, y'_n\right) = G_d$ *;*
End

Gambar 3.7 *Pseudocode dari algoritma PSO yang diusulkan*

3.4 Prediksi Trajektori Halangan Dalam Perencanaan Jalur Dinamis

Adanya halangan yang bergerak menyebabkan persoalan perencanaan jalur menjadi persoalan optimisasi dinamis. Perencanaan jalur dengan pendekatan optimasi dinamis memiliki tingkat kesulitan yang lebih tinggi karena menuntut algoritma optimasi memiliki kemampuan adaptasi terhadap perubahan kondisi ruang pencarian. Oleh karena itu dalam tesis ini dilakukan prediksi halangan agar dapat menghitung potensi terjadinya tumbukan antara halangan yang bergerak dan robot pada jalur yang telah direncanakan.

3.4.1.1 Model Halangan

Sistem visi ataupun sensor dari robot mendapatkan informasi posisi robot setiap waktu sampling. Karena informasi mengenai halangan yang didapatkan hanya posisi sedangkan dinamika halangan tidak bias diketahui secara akurat maka pergerakan halangan didekati dengan persamaan polynomial dengan orde 2, seperti pada persamaan

$$\begin{aligned} x(t) &= a_0 + a_1 t + a_2 t^2 \\ y(t) &= b_0 + b_1 t + b_2 t^2 \end{aligned} \tag{3.22}$$

dimana $a_{0,1,2}$ dan $b_{0,1,2}$ adalah parameter yang dicari dari tiga kali sampling data pengukuran posisi.

$$\begin{aligned} y_o(t) &= a_0 + a_1 t + a_2 t^2 \\ y_o(t - ts) &= a_0 + a_1 (t - ts) + a_2 (t - ts)^2 \\ y_o(t - 2ts) &= a_0 + a_1 (t - 2ts) + a_2 (t - 2ts)^2 \end{aligned} \tag{3.23}$$

Persamaan (3.23) dapat dibentuk menjadi persamaan (3.24)

$$\begin{bmatrix} y_o(t) \\ y_o(t - ts) \\ y_o(t - 2ts) \end{bmatrix} = \begin{bmatrix} 1 & t & t \\ 1 & (t - ts) & (t - ts)^2 \\ 1 & (t - 2ts)^2 & (t - 2ts)^2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} \tag{3.24}$$

dimana $\begin{bmatrix} 1 & t & t \\ 1 & (t - ts) & (t - ts)^2 \\ 1 & (t - 2ts)^2 & (t - 2ts)^2 \end{bmatrix} = T\big(t, (t - ts), (t - 2ts)\big)$ adalah bukan matrik singular maka parameter $a_{0,1,2}$ dan $b_{0,1,2}$ dicari dengan persamaan (3.25)

$$\begin{aligned} A &= T^{-1}\big(t, (t - ts), (t - 2ts)\big) Y\big(t, (t - ts), (t - 2ts)\big) \\ B &= T^{-1}\big(t, (t - ts), (t - 2ts)\big) X\big(t, (t - ts), (t - 2ts)\big) \end{aligned} \tag{3.25}$$

3.5 Perencanan Jalur Ulang

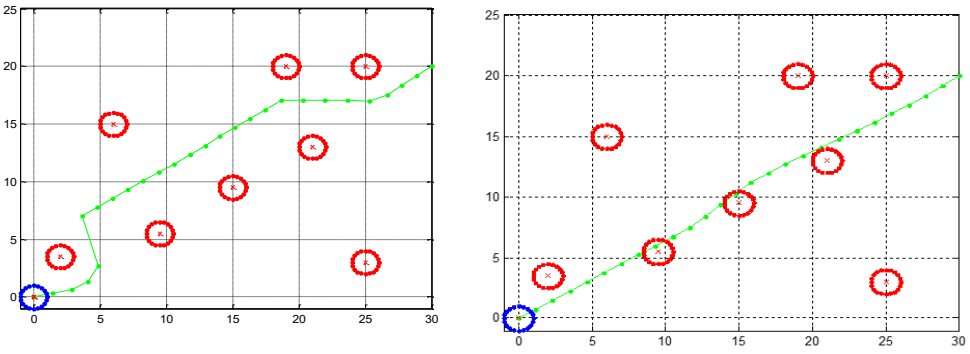
Setelah trejektori diprediksi selanjutnya adalah melakukan perencanaan jalur ulang ketika trayektori dari halangan berpotensi terjadi tumbukan terhadap mobile robot. Untuk mengetahui apakah trajektori halangan bertumbukan atau tidak dilakukan dengan bebrapa langkah berikut:

1. Sampling 3 posisi terakhir dari halangan sehingga didapatkan parameter $(a_0, a_1 a_2)(b_0, b_1, b_2)$.
2. Prediksi trayektori halangan dari t hingga $t + 6ts$ dengan persamaan (3.22).
3. Periksa ke 6 posisi dari prediksi halangan tersebut apakah kurang dari radius aman terhadap jalur yang dibentuk.
4. Jika kurang dari radius aman maka periksa apakah waktu yang dibutuhkan mobile robot terhadap titik tersebut sama dengan $6ts$ jika iya lakukan perencanaan ulang dengan menggunakan algoritma PSO pada gambar Gambar 3.7.

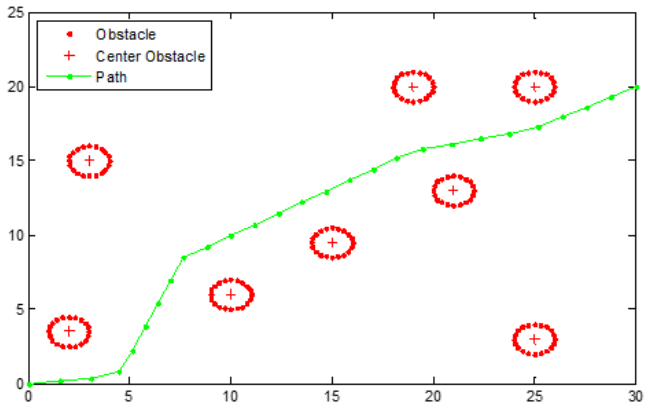
BAB 4
HASIL DAN PEMBAHASAN

4.1 Pengaturan Bobot Fungsi Tujuan

Pada persamaan (3.9), fungsi panjang lintasan, tingkat resiko tumbukan dan fungsi kriteria *smooth* dijadikan satu fungsi tujuan dengan pembobot. Karena tujuan utama dari perencanaan jalur adalah mencari jalur terpendek dan menghindari halangan, sehingga kedua fungsi tersebut diberi bobot penuh yaitu 1. Selanjutnya fungsi keriteria *smooth* bertujuan agar jalur yang dibentuk lurus dan sedikit berbelok yang merupakan bukan tujuan utama dari perencanan jalur. Berdasarkan alasan tersebut bobot γ_3 ditala untuk mendapatka representasi jalur yang sedikit berbelok dan tanpa bertumbukan. Penalaan bobot dilakukan pada halangan statis dilakukan dengan memberikan bobot 0 hingga 1 dengan skala 0.1. Pada Gambar 4.1 (a) menunjukan reperesentasi jalur tanpa menggunakan fungsi keriteria *smooth* atau bobot bernilai 0 dan gambar (b) saat diberi bobot 1. Hasil terbaik diperoleh ketika bobot bernilai 0.2 dengan panjang jalur dan resiko tumbukan terkecil yaitu 37.824 m 0.**0**122 yang ditunjukan pada gambar Gambar 4.2.



Gambar 4.1 Perbandingan Reperesentasi Jalur dengan bobot γ_3 0 dan 1



Gambar 4.2 Reperesentasi Jalur dengan bobot $\gamma_3 = 0.2$

4.2 Perencanan Jalur Pada Halangan Statis

Pengujian pertama kali dilakukan pada skenario lingkungan yang statis dengan jumlah halangan 8 buah halangan berbentuk lingkaran. Pengujian ini dilakukan dengan membandingkan algoritma PSO yang diusulkan dengan *Adaptive Inertia* (AIW) PSO dan Standard PSO untuk mengetahui performa algoritma PSO yang diusulkan. Agar pengujian seimbang dilakukan dengan parameter yang sama seperti yang ditunjukan Tabel 4.1.

Tabel 4.1 Parameter Parameter Pada Pengujian Statis

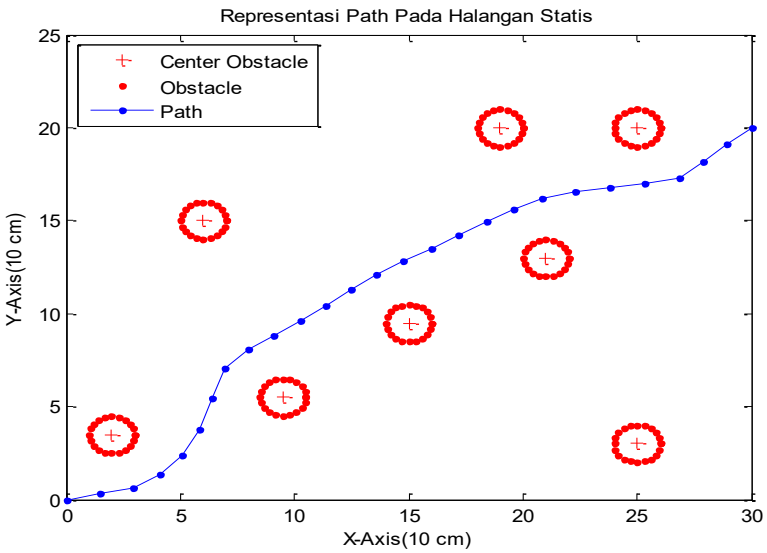
No	Parameter	PSO yang diusulkan	AIW-PSO	SPSO
1	Jumlah Partikel	25	25	25
2	Dimensi Jalur	25	25	25
3	Waktu Iterasi	300	300	300
4	w_{max}	1	1	0.85(konstan)
5	w_{min}	0.1	0.1	0.85(konstan)
6	$c_{1_{max}}$	2	2(konstan)	2(konstan)
7	$c_{1_{min}}$	0.2	2(konstan)	2(konstan)
8	$c_{2_{max}}$	2	2(konstan)	2(konstan)
9	$c_{2_{min}}$	1	2(konstan)	2(konstan)
10	γ	10^{-3}	-	-

Pada pengujian ini robot start pada posisi (0,0) dan target berada pada (30,20). Posisi kedelapan buah halangan tersebut ditunjukan pada Tabel 4.2.

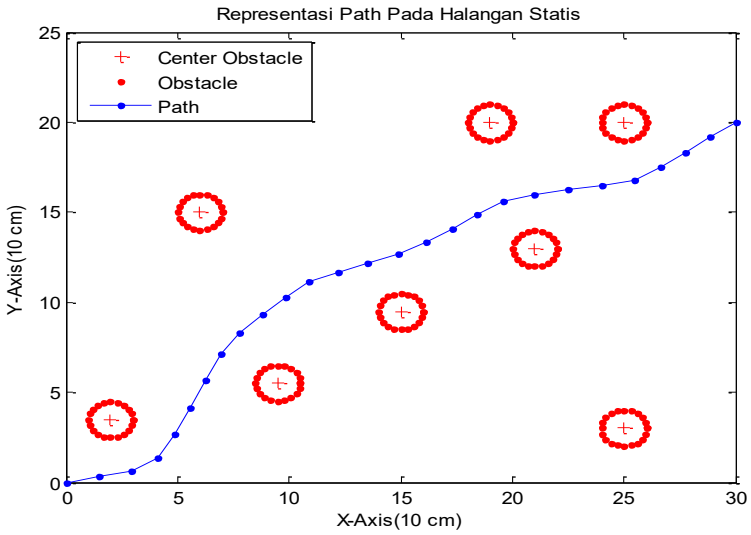
Tabel 4.2 Posisi Halangan Pada Pengujian Halangan Statis

Halangan	Posisi	
	X	*Y*
Halangan 1	2	3.5
Halangan 2	9.5	5.5
Halangan 3	15	9.5
Halangan 4	19	20
Halangan 5	21	13
Halangan 6	25	3
Halangan 7	25	20
Halangan 8	6	15

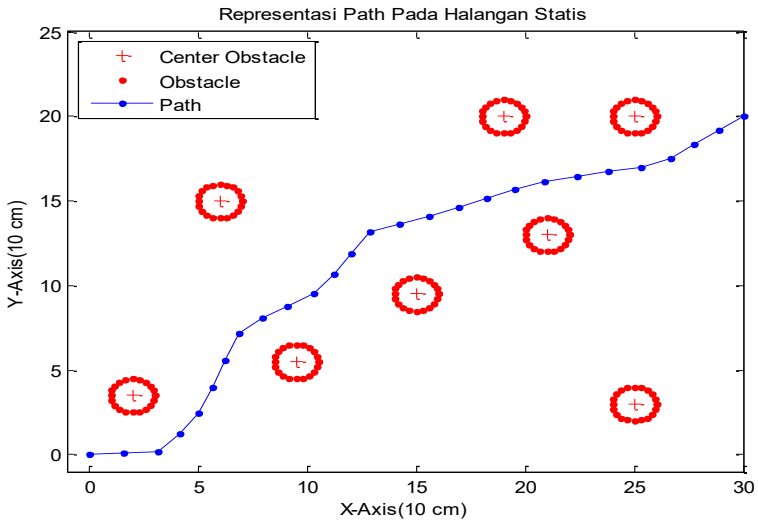
Pada pengujian perbandingan dengan algoritma PSO lain, hasil statistic dengan 20 kali pengujian menujukan bahwa algoritma yang diusulkan menghasilkan panjang jalur rata rata 37.913 m dan rata rata error sudut adalah 85.831^0 dengan sudut yang diinginkan adalah 180^0 pada setiap titik jalur. Hasil tersbut menunjukan bahwa jalur yang dihasilkan dari algoritma PSO yang diusulkan 3% lebih pendek.



Gambar 4.3 Representasi jalur yang dihasilkan oleh algoritma PSO yang diusulkan



Gambar 4.4 Representasi jalur yang dihasilkan oleh algoritma AIW-PSO



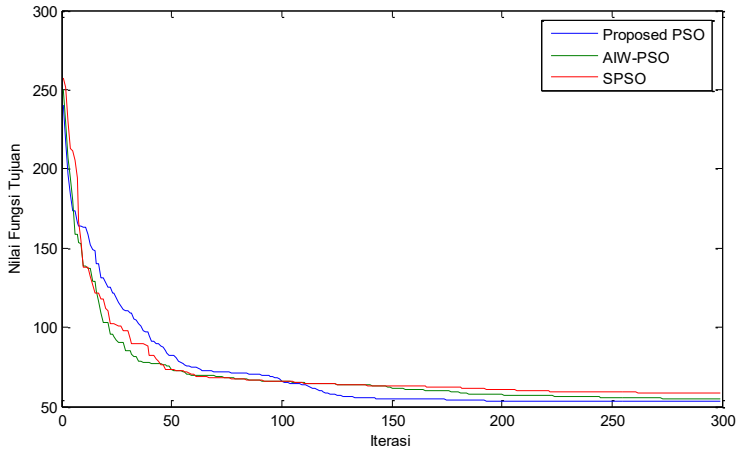
Gambar 4.5 Representasi jalur yang dihasilkan oleh algoritma SPSO

Pada Tabel 4.3 menunjukan bahwa PSO yang diusulkan juga memiliki nilai tingkat resiko tumbukan rata rata yang lebih kecil dari yang lainya yaitu 0.229 meskipun pada jalur yang terburuk pun tetap menghasilkan jalur dengan tingkat resiko tumbukan terkecil yaitu 0.880 seperti yang ditunjukan oleh Gambar 4.3, representasi jalur aman dari halangan dan *smooth*.

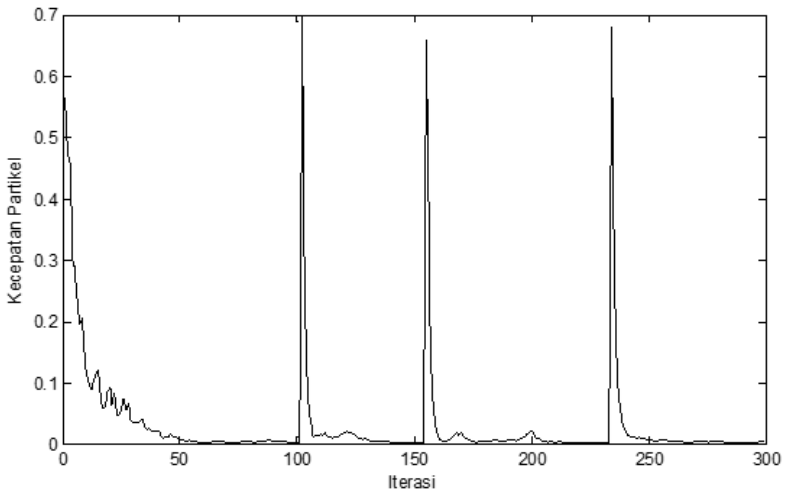
Pada pengujian dengan halangan statis kecepatan konvergensi dari algoritma yang diusulkan lebih cepat dibandingkan dengan algoritma lain, seperti yang ditunjukan Gambar 4.6 solusi optimal diperoleh kurang dari 150 iterasi. Pada iterasi awal, konfergensi pertama kali terjadi pada 100 iterasi pertama dan

kemudian reinisialisasi partikel pada iterasi selanjutnya mengakibatkan solusi optimal ditemukan disekitar konvergensi pertama tersebut.

Proses reinisialisasi pertikel dapat terlihat pada Gambar 4.6, pada 100 iterasi pertama terjadi konvergensi pertama kemudian algoritma melakukan reinisialisasi partikel yang menghasilkan solusi baru yang dapat meminimumkan fungsi tujuan. Reinisialisali selanjutnya tidak menyebabkan algoritma mendapatkan solusi baru lagi sehingga solusi dari reinisalisasi yang pertama merupakan solusi yang optimal.



Gambar 4.6 Perbandingan Nilai Fungsi Tujuan Persamaan (3.9) Setiap Iterasi



Gambar 4.7 Kecepatan partikel PSO pada setiap Iterasi

Tabel 4.3 Hasil statistik dari 20 kali pengujian pada halangan statis

Kriteria	Jumlah	PSO Usulan	AIW-PSO	S-PSO
Panjang Jalur (m)	Min	37.643	**37.610**	37.622
	Max	**38.691**	42.020	47.499
	Mean	**37.913**	38.3465	39.991
Tingkat Resiko Tumbukan	Min	**0**	**0**	**0**
	Max	**0.880**	2.076	6.829
	Mean	**0.229**	0.536	0.509
Kriteria *Smooth* (error sudut dalam derajat)	Min	74.408	**67.192**	73.912
	Max	**114.056**	156.049	194.618
	Mean	85.831	**84.975**	109.5744
Nilai Fungsi Tujuan	Min	52.588	**51.122**	52.689
	Max	**61.514**	75.285	86.473
	Mean	**55.309**	55.878	62.415
Standar Deviasi Jalur	Jumlah	**0.121**	0.196	0.357
Kecepatan Konvergensi	Mean	**152.105**	168.421	186.842
Tumbukan	Jumlah	**0**	0	0

4.3 Perencanan Jalur Pada Halangan Dinamis

Pengujian pada halangan dinamis dilakukan dengan membuat dua skenario lingkungan. Skenario lingkungan yang pertama terdiri dari 8 buah halangan berbentuk lingkaran dimana 1 halangan bergerak, Sedangkan pada skenario yang kedua terdapat 2 halangan yang bergerak untuk memotong jalur yang dilalui oleh mobile robot.

4.3.1 Pengujian dengan 1 halangan bergerak

Pada pengujian ini dilakukan dengan posisi halangan yang menyerupai pada skenario pengujian lingkungan halangan statis, dengan parameter halangan ditunjukan pada Tabel 4.4.

Tabel 4.4 Parameter halangan pada pengujian 1 halangan bergerak

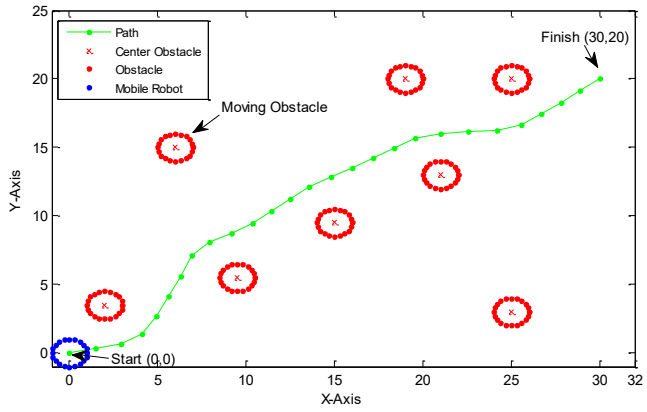
Halangan	Posisi Inisial		Kecepatan		Percepatan	
	X	Y	V_x	V_y	a_x	a_y
Halangan 1	2	3.5	0	0	0	0
Halangan 2	9.5	5.5	0	0	0	0
Halangan 3	15	9.5	0	0	0	0
Halangan 4	19	20	0	0	0	0
Halangan 5	21	13	0	0	0	0
Halangan 6	25	3	0	0	0	0
Halangan 7	25	20	0	0	0	0
Halangan 8	6	15	0.5	-0.8	0.1	-0.1

Pada pengujian ini untuk membentuk jalur awal robot parameter PSO yang digunakan sama dengan pada pengujian parameter statis. Selanjutnya untuk parameter replanning menggunakan parameter sperti yang ditunjukan Tabel 4.5.

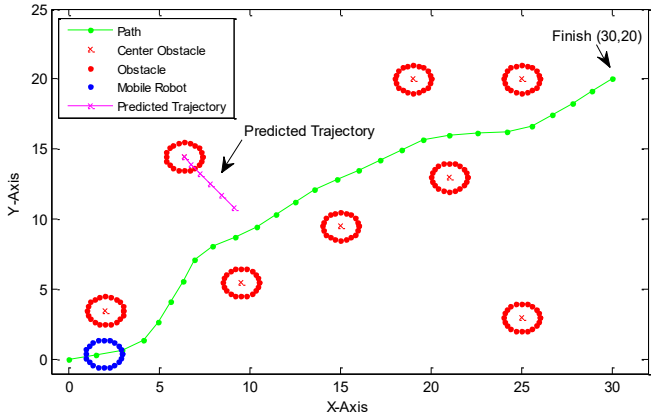
Gambar 4.8 merupakan jalur awal yang dibentuk oleh algoritma PSO yang diusulkan yang sama seperti hasil pengujian pada halangan statis. Kemudian pada detik ke 0.6, setelah mendapatkan data dari 3 kali sampling posisi algoritma perdiksi mengetahui trayektori yang dilalui halangan. Setelah prediksi mengetahui bahwa pada koordinat (11,8.4) akan terjadi tumbukan, pada detik ke 3 algoritma memperbaharui jalur awal yang dihasilkan. Melalui jalur baru tersebut robot dapat menghindari halangan yang bergerak sperti yang ditunjukan pada Gambar 4.11, yang merupakan pergerakan robot dan halangan dari detik ke 4.2 hingga 7.6.

Tabel 4.5 Parameter PSO yang Diusulkan Untuk Perencanaan Jalur Ulang

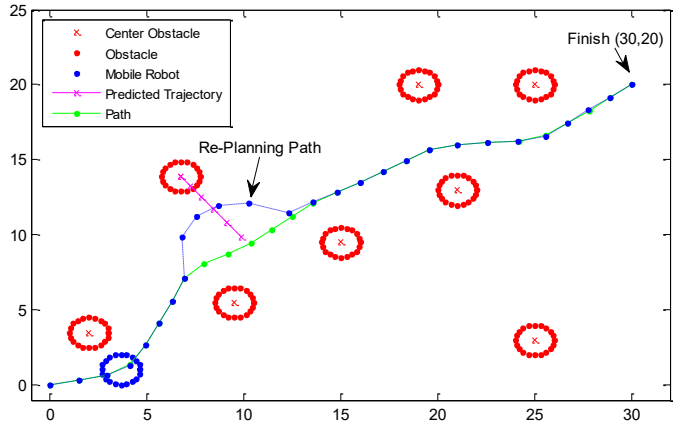
No	Parameter	PSO yang diusulkan
1	Jumlah Partikel	20
2	Dimensi Jalur	25
3	Waktu Iterasi	50
4	w_{max}	1
5	w_{min}	0.1
6	$c_{1_{max}}$	2
7	$c_{1_{min}}$	0.2
8	$c_{2_{max}}$	2
9	$c_{2_{min}}$	1
10	γ	10^{-3}



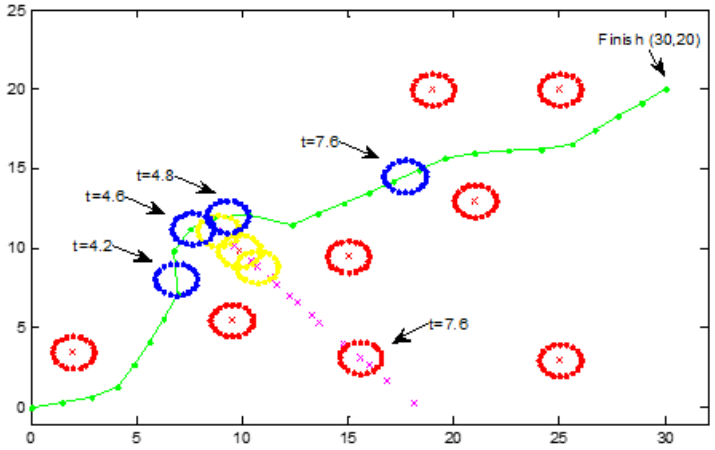
Gambar 4.8 Hasil Jalur Awal yang Dihasilkan (t=0)



Gambar 4.9 Hasil Prediksi Trayektori dari Halangan pada $t = 0.6s$



Gambar 4.10 Hasil Perencanaan Ulang Jalur Menggunakan PSO pada $t = 3s$



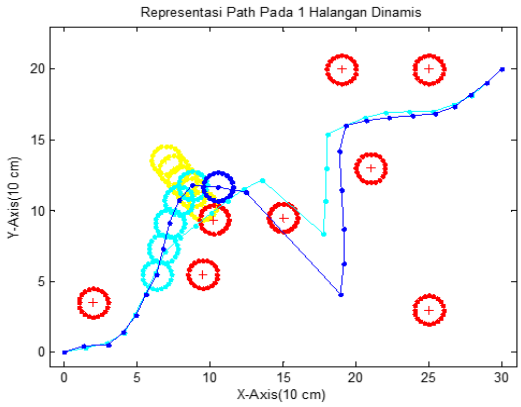
Gambar 4.11 Pergerakan *Mobile Robot* dan Halangan saat $t = 4.2s$ hingga $7.6s$

Tabel 4.6 Hasil statistik dari 20 kali pengujian pada 1 halangan bergerak

Kriteria	Jumlah	PSO Usulan	AIW-PSO	S-PSO
Panjang Jalur (m)	Min	**38.897**	39.139	39.201
	Max	**40.867**	52.074	53.714
	Mean	**39.704**	41.061	41.496
Tingkat Resiko Tumbukan	Min	**0**	0.126	0
	Max	**2.3044**	7.026	11.713
	Mean	**0.683**	1.624	3.560
Kriteria *Smooth* (error sudut dalam derajat)	Min	**87.051**	93.284	128.608
	Max	**145.236**	217.720	232.825
	Mean	**109.639**	132.010	164.727
Nilai Fungsi Tujuan	Min	**56.311**	58.451	68.623
	Max	**68.986**	95.976	97.560
	Mean	**62.316**	69.087	78.002
Standar Deviasi Jalur	Jumlah	**0.228**	0.467	0.478
Kecepatan Konvergensi	Mean	**59.73684**	74.21053	78.15789
Tumbukan	Jumlah	**0**	1	1

Pada Tabel 4.6, menunjukan bahwa PSO yang diusulkan menghasilkan jalur yang lebih optimal disbanding dengan algoritma PSO yang lainya. Pada 20 kali pengujian menunjukan bahwa PSO yang diusulkan menghasilkan jalur yang lebih pendek 3.3% dibanding dengan algoritma PSO lain dengan panjang rata rata 39.704 m dan memiliki jalur 80% lebih aman dengan tingkat resiko tumbukan rata rata hanya 0.683. Pada hasil pengujian tersebut, AIW-PSO dan S-PSO mengalami

tumbukan sebanyak masing masing 1 kali. Gambar 4.12 menunjukan representasi jalur dari kedua algoritma tersebut. Kedua algoritma tersebut menghasilkan jalur yang lokal minimum setelah melewati halangan bergerak.



Gambar 4.12 Reperesentasi jalur algoritma AIW-PSO yang mengalami tumbukan

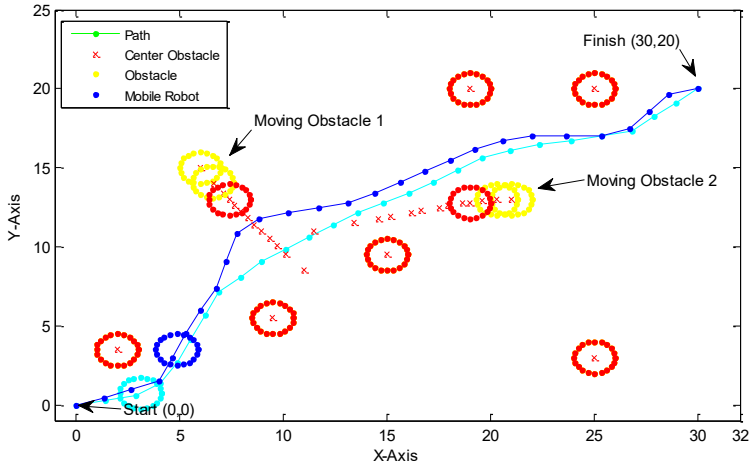
4.3.2 Pengujian dengan 2 halangan bergerak

Pengujian selanjutnya dilakukan dengan scenario lingkungan yang memiliki 2 halangan bergerak yang masing masing halangan akan memotong jalur pergerakan dari *mobile robot.* Pada pengujian ini kecepatan dan percepatan tetap konstan seperti yang ditunjukan pada Tabel 4.7.

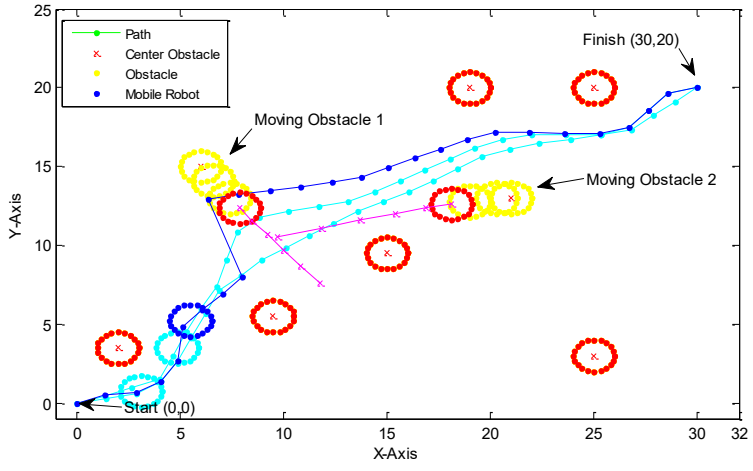
Tabel 4.7 Parameter halangan pada pengujian 2 halangan bergerak

Halangan	Posisi Inisial		Kecepatan(m/s)		Percepatan (m/s^2)	
	X	Y	V_x	V_y	a_x	a_y
Halangan 1	2	3.5	0	0	0	0
Halangan 2	9.5	5.5	0	0	0	0
Halangan 3	15	9.5	0	0	0	0
Halangan 4	19	20	0	0	0	0
Halangan 5	21	13	0	0	0	0
Halangan 6	25	3	0	0	0	0
Halangan 7	25	20	-0.4	0.1	-0.3	-0.1
Halangan 8	6	15	0.5	-0.8	0.1	-0.1

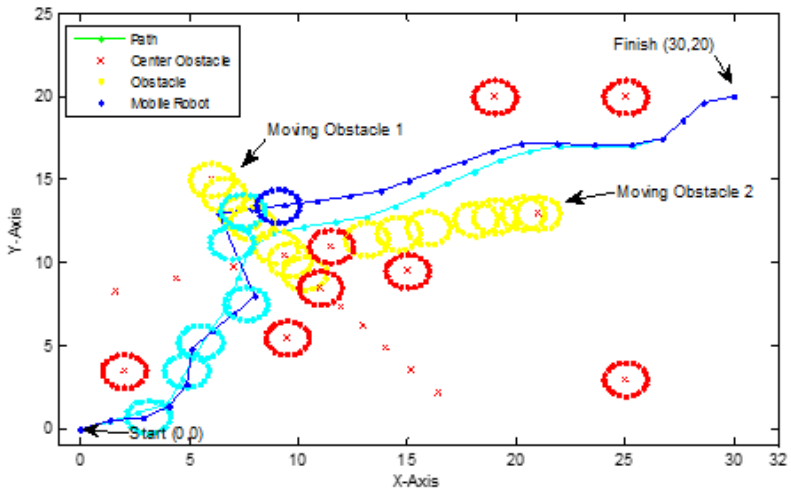
Pada detik ke 0.8 setelah algoritma prediksi mengetahui bahwa trajektori halangan pertama berpotensi terjadi tumbukan, kemudian melakukan *replanning* jalur seperti yang ditunjukan Gambar 4.13. Jalur yang terbentuk setelah melakukan *replanning* yang pertama ternyata tidak menjamin terbebas dari tumbukan dari pergerakan halangan kedua, sehingga dilakukan perbaikan jalur yang kedua seperti yang ditunjukan Gambar 4.14. Adanya dua halangan yang bergerak membuat representasi jalur memiliki tikungan tajam yang kurang baik pada pergerakan mobile robot.



Gambar 4.13 Representasi jalur saat *replanning* yang pertama pada detik ke 2

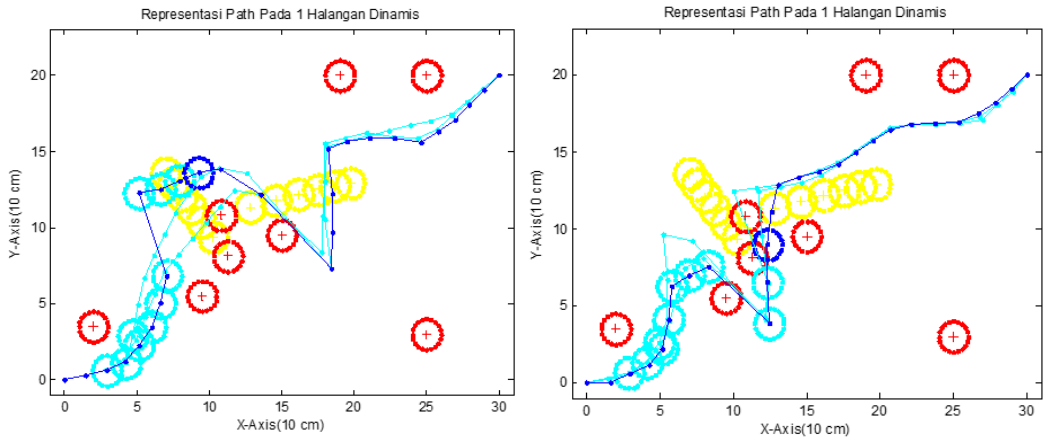


Gambar 4.14 Representasi jalur saat *replanning* yang kedua pada detik ke 3.2



Gambar 4.15 Pergerakan mobile robot dan halangan dari detik ke 0.2-6

Pada 20 kali pengujian PSO yang diusulkan menghasilkan jalur yang 5% lebih pendek dan lebih aman dengan tidak pernah terjadi tumbukan. Sama seperti sebelumnya standard PSO dan AIW-PSO mengalami tumbukan masing masing 1 kali karena terperangkap dalam nilai minimum lokal seperti yang ditunjukan pada Gambar 4.16. Algoritma AIW-PSO mengalami minimum lokal setelah melewati halangan bergerak sehinggga bertumbukan dengan halangan statis.

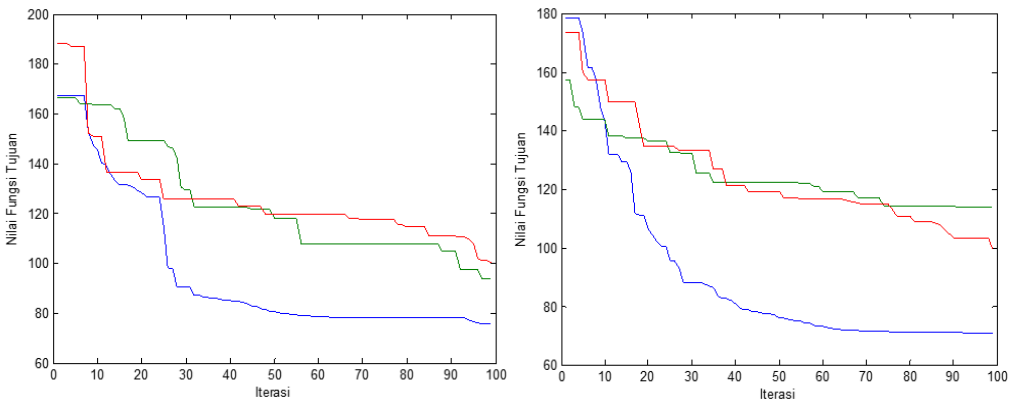


Gambar 4.16 Reperesentasi jalur algoritma AIW-PSO dan S-PSO yang mengalami tumbukan

Tabel 4.8 Hasil statistic dari 20 kali pengujian pada 2 halangan dinamis

Kriteria	Jumlah	PSO Usulan	AIW-PSO	S-PSO
Panjang Jalur	Min	42.659	**42.347**	42.716
	Max	**47.243**	54.279	51.027
	Mean	**44.040**	46.399	45.325
Tingkat Resiko Tumbukan	Min	**0**	0.039	2.020
	Max	**17.600**	30.711	30.648
	Mean	**4.923**	8.471	12.211
Kriteria *Smooth* (error sudut dalam derajat)	Min	**102.745**	150.441	160.161
	Max	**163.453**	254.834	312.099
	Mean	**129.794**	199.775	215.868
Nilai Fungsi Tujuan	Min	**64.256**	78.676	81.527
	Max	**92.217**	114.013	136.764
	Mean	**75.461**	94.825	100.709
Standar Deviasi Jalur	Jumlah	**0.410**	0.513	0.587
Kecepatan Konvergensi	Mean	**62.631**	77.105	86.316
Tumbukan	Jumlah	**0**	1	1

Gambar 4.17 menunjukan perbandingan konvergensi pada percobaan ke 13 dan ke 15. Pada gambar (a) menunjukan pada iterasi ke 30 terjadi konvergensi pertamakali yang mengakibatkan reinisialisasi partikel yang menghasilkan solusi baru yang optimal. Selain itu pada Tabel 4.8 menunjukan bahwa reinisialisasi partikel menjamin tidak terjadi konvergensi prematur, karena dapat keluar dari lokal minimum. Pengaturan parameter secara adaptif dapat mempercepat konvergensi 20 iterasi lebih cepat dibanding parameter konstan pada PSO standard.



Gambar 4.17 (a) Perbandingan kecepatan konvergensi pada pengujian ke 13
(b) Perbandingan kecepatan konvergensi pada pengujian ke 15

BAB 5
PENUTUP

5.1 Kesimpulan

Pada penelitian ini setelah permasalahan perencanaan jalur diformulasikan dengan menggunakan tiga buah fungsi tujuan yaitu panjang jalur, tingkat resiko tumbukan, dan keriteria *smooth,* algoritma PSO adaptif diaplikasikan dan dikembangkan dengan menambahkan fungsi distribusi gausian untuk mempercepat konvergensi pencarian solusi. Selain itu dari analisa kecepatan partikel PSO, ditambahkan reiniialisasi partikel untuk mencegah konvergensi prematur karena terperangkap dalam minimum lokal. Hasil simulasi dan perbandingan dengan metode AIW-PSO dan standard PSO menunjukan bahwa algoritma PSO yang diusulkan dapat mempercepat konvergensi kurang dari 150 iterasi pada halangan statis dan pada halangan dinamis konvergensi rata rata terjadi pada 200 iterasi. Kemudian dengan reiinisialisasi partikel dapat menghasilkan solusi global yang menghasilkan 3% jalur yang lebih pendek, 10% lebih *smooth* dan terjamin aman dari tumbukan.

5.2 Saran

Untuk mengembangkan penelitian ini analisa dilakukan secara stokastik untuk mengetahui perilaku PSO lebih mendalam. Selain itu variable keputusan ditambahkan dengan variable kecepatan robot agar mendapatkan hasil yang opimal dalam kondisi lingkungan yang dinamis.

Halaman ini sengaja dikosongkan

DAFTAR PUSTAKA

[1] I. Rudas, World Scientific and Engineering Academy and Society, and Budapesti Műszaki és Gazdaságtudományi Egyetem, Eds., "Mobile robot path planning using exact cell decomposition and potential field methods," *Proceeding SMO09 Proc. 9th WSEAS Int. Conf. Simul. Model. Optim.*, 2009.

[2] O. Khatib, "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots," *Int. J. Robot. Res.*, vol. 5, no. 1, pp. 90–98, Mar. 1986.

[3] H.-P. Huang and S.-Y. Chung, "Dynamic visibility graph for path planning," in *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, 2004, vol. 3, pp. 2813–2818.

[4] L. Wang, Y. Liu, H. Deng, and Y. Xu, "Obstacle-avoidance path planning for soccer robots using particle swarm optimization," in *Robotics and Biomimetics, 2006. ROBIO'06. IEEE International Conference on*, 2006, pp. 1233–1238.

[5] C.-J. Lin, T.-H. S. Li, P.-H. Kuo, and Y.-H. Wang, "Integrated particle swarm optimization algorithm based obstacle avoidance control design for home service robot," *Comput. Electr. Eng.*, vol. 56, pp. 748–762, Nov. 2016.

[6] B. Tang, Z. Zhu, and J. Luo, "Hybridizing Particle Swarm Optimization and Differential Evolution for the Mobile Robot Global Path Planning," *Int. J. Adv. Robot. Syst.*, vol. 13, no. 3, p. 86, Jun. 2016.

[7] F. Van den Bergh and A. P. Engelbrecht, "A convergence proof for the particle swarm optimiser," *Fundam. Informaticae*, vol. 105, no. 4, pp. 341–374, 2010.

[8] A. Nickabadi, M. M. Ebadzadeh, and R. Safabakhsh, "A novel particle swarm optimization algorithm with adaptive inertia weight," *Appl. Soft Comput.*, vol. 11, no. 4, pp. 3658–3670, Jun. 2011.

[9] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Self-Organizing Hierarchical Particle Swarm Optimizer With Time-Varying Acceleration Coefficients," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 240–255, Jun. 2004.

[10] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, 1998, pp. 69–73.

[11] A. Nickabadi, M. M. Ebadzadeh, and R. Safabakhsh, "A novel particle swarm optimization algorithm with adaptive inertia weight," *Appl. Soft Comput.*, vol. 11, no. 4, pp. 3658–3670, Jun. 2011.

[12] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Neural Networks, 1995. Proceedings., IEEE International Conference on*, 1995, vol. 4, pp. 1942–1948.

[13] F. Van den Bergh and A. P. Engelbrecht, "A convergence proof for the particle swarm optimiser," *Fundam. Informaticae*, vol. 105, no. 4, pp. 341–374, 2010.

[14] Y. Liu, X. Zhang, X. Guan, and D. Delahaye, "Adaptive sensitivity decision based path planning algorithm for unmanned aerial vehicle with improved

particle swarm optimization," *Aerosp. Sci. Technol.*, vol. 58, pp. 92–102, Nov. 2016.
[15] N. Habib, A. Soeprijanto, D. Purwanto, and M. H. Purnomo, "Mobile Robot Motion Planning to Avoid Obstacle Using Modified Ant Colony Optimization," *Appl. Mech. Mater.*, vol. 776, pp. 396–402, Jul. 2015.
[16] Z. Zeng, K. Sammut, L. Lian, F. He, A. Lammas, and Y. Tang, "A comparison of optimization techniques for AUV path planning in environments with ocean currents," *Robot. Auton. Syst.*, vol. 82, pp. 61–72, Aug. 2016.
[17] J. K. Kordestani, A. Rezvanian, and M. R. Meybodi, "An efficient oscillating inertia weight of particle swarm optimisation for tracking optima in dynamic environments," *J. Exp. Theor. Artif. Intell.*, vol. 28, no. 1–2, pp. 137–149, Mar. 2016.
[18] D. Alrijadjis, K. Tanaka, S. Nakashima, and S. Mu, "Application of a Modified PSO Algorithm to Self-Tuning PID Controller for Ultrasonic Motor," 2013, pp. 249–256.
[19] A. Djoewahir, K. Tanaka, and S. Nakashima, "Adaptive PSO-based self-tuning PID controller for ultrasonic motor," *Int. J. Innov. Comput. Inf. Control*, vol. 9, no. 10, pp. 3903–3914, 2013.

LAMPIRAN

Persamaan kecepatan dan posisi PSO:

$$V(t+1) = wV(t) + c_1 r_1\big(P - x(t)\big) + c_2 r_2\big(G - x(t)\big) \tag{0.1}$$

$$x(t+1) = x(t) + V(t+1) \tag{0.2}$$

dari persamaan (0.2) didapatkan

$$x(t) = x(t-1) + V(t) \tag{0.3}$$

Substitusikan persamaan (0.3) ke persamaan (0.1) didapatkan

$$\begin{aligned} V(t+1) = wV(t) &+ \emptyset_1[P - x(t-1) - V(t)] \\ &+ \emptyset_2[G - x(t-1) - V(t)] \end{aligned} \tag{0.4}$$

persamaan (0.4), dapat ditulis kembali

$$V(t+1) = (w - \emptyset_1 - \emptyset_2)V(t) + (\emptyset_1 P + \emptyset_2 G) - (\emptyset_1 + \emptyset_2)x(t-1) \tag{0.5}$$

kemudian dari persamaan (0.1) ketika $V(t)$ didapatkan

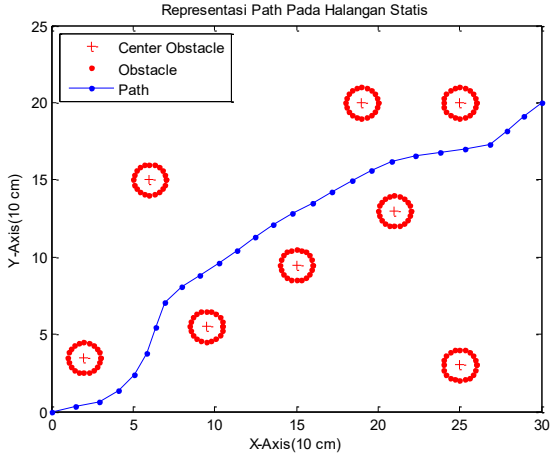
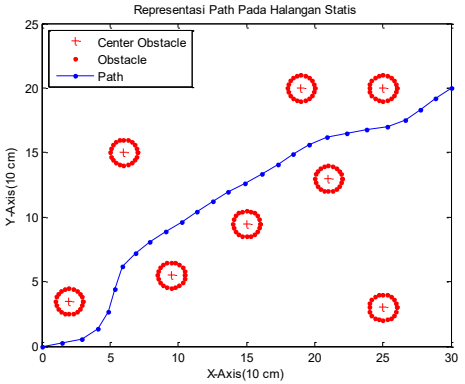
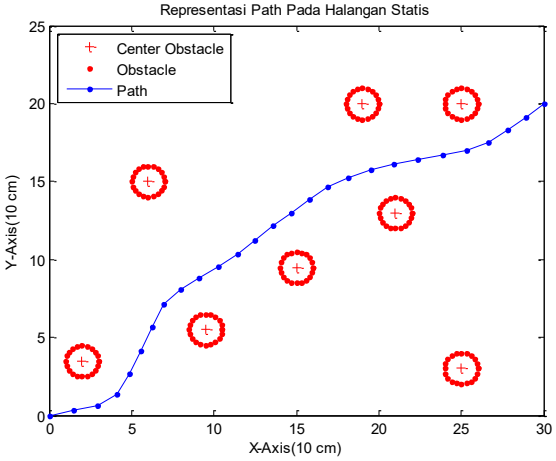
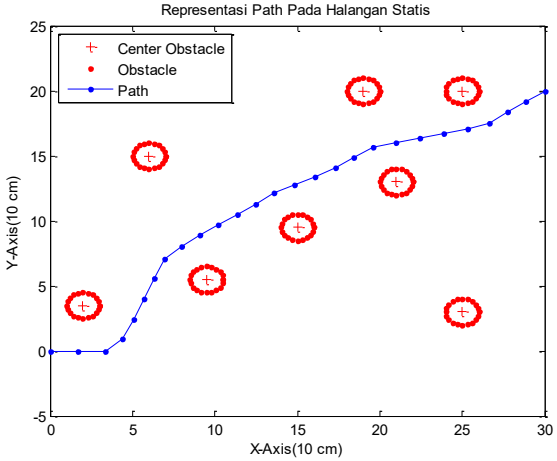
$$x(t-1) = \frac{-V(t) + wV(t-1) - (\emptyset_1 P + \emptyset_2 G)}{(\emptyset_1 + \emptyset_2)} \tag{0.6}$$

substitusikan persamaan (0.6) ke persamaan (0.5) didapatkan

$$V(t+1) = (1 + w - \emptyset_1 - \emptyset_2)V(t) + wV(t-1) \tag{0.7}$$

Halaman ini sengaja dikosongkan

Data Pengujian PSO yang Diusulkan Pada Halangan Statis

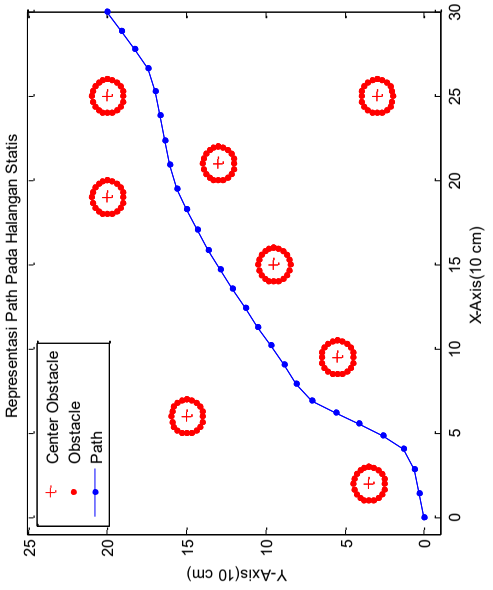
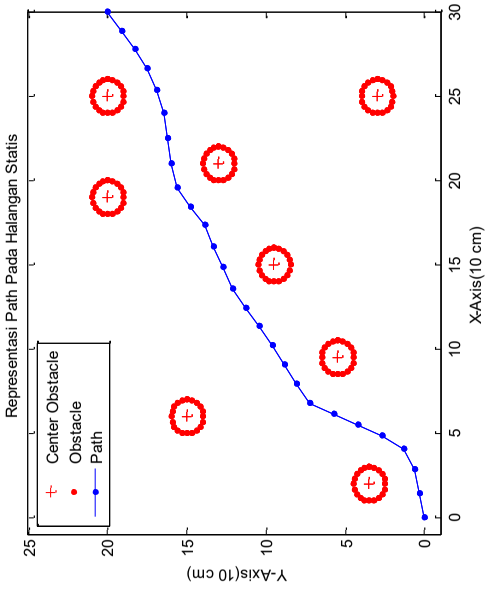
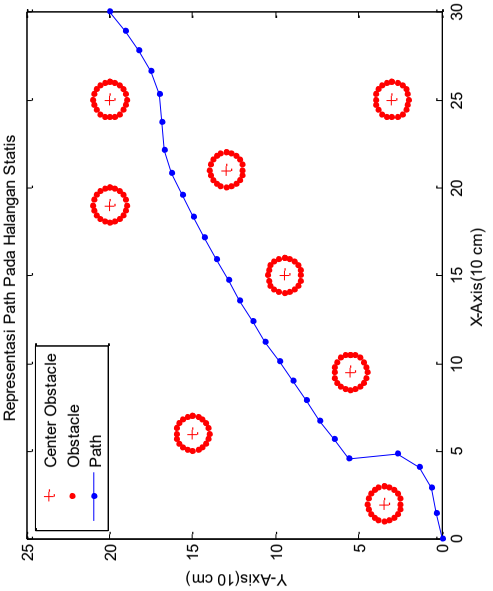
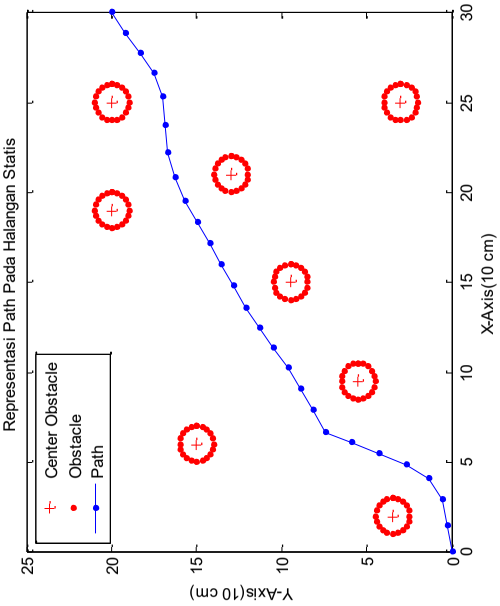
	PSO yang diusulkan					AIW-PSO				
No	**Representasi Jalur**	J_d	J_r	J_s	f	**Representasi Jalur**	J_d	J_r	J_s	f
1		37.8249918462072	0.122880415643474	81.7360314025960	54.2950785423699		37.8005911188118	1.92122570486885	83.6607056840914	56.4539579604989
2		37.7041993718924	0.00275011020105964	74.4084790786929	52.5886452978321		38.2594006583396	0.347264341415525	82.2636940059422	55.0594038009435

3		38.068822868145 2	0.005793654483507 99	90.41927980428 55	56.15847248348 58		37.90753373048 03	0.01365161455149 98	82.93552010858 13	54.50828936674 81
4		38.034650210467 3	0.011930538775586 4	91.51997583461 52	56.35057591616 59		37.72347785556 12	0.16045725572011 9	72.24891989454 27	52.33371909018 99

5		37.643726159393 1	0.42454049985745 1	77.623059665200 5	53.592878592290 7		37.701950549192 1	0.41211957711383 6	80.898183752365 9	54.293706876779 1
6		37.786472973213 5	0.50046846901214 5	78.166010271271 0	53.920143496479 9		37.694481310148 8	0.12037128516571 9	73.168548833913 2	52.448562362097 1

7		37.932382505094	0.88079744082344	84.508209450596	55.714821836037		40.624781197260	0.12404443770020	68.838650723983	54.516557797573
8		37.783912288710	0.02484658575917	76.348023356766	53.078363545823		38.343846261745	0.01837891935193	80.709064472292	54.504038075556

9		37.699825319144 3	0.42903959839498 0	77.033763101014 6	53.535617537742 3		38.095709543560 9	0.00301865714893879	80.531384797639 6	54.205005160237 8
10		37.791289991223 4	0.04484788005898 02	99.532541032511 8	57.742646077784 7		41.224667966340 4	2.07604713301547	121.62645506200 9	67.626006111757 6

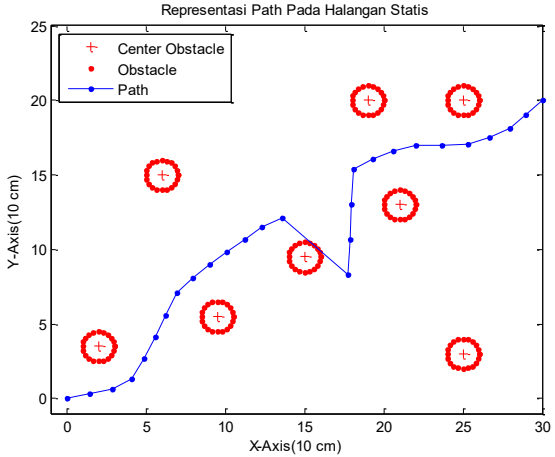
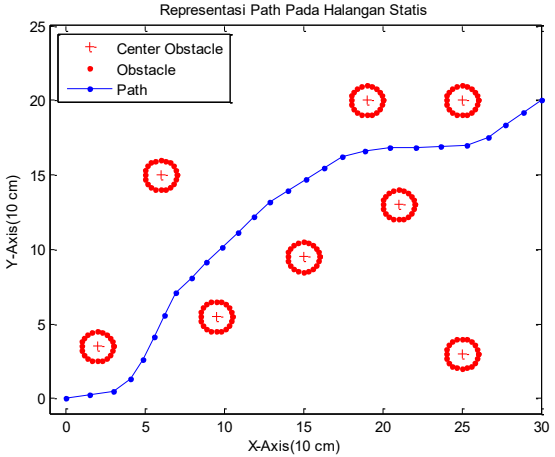
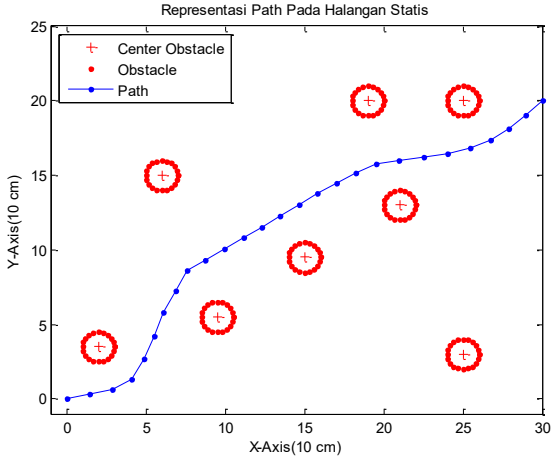
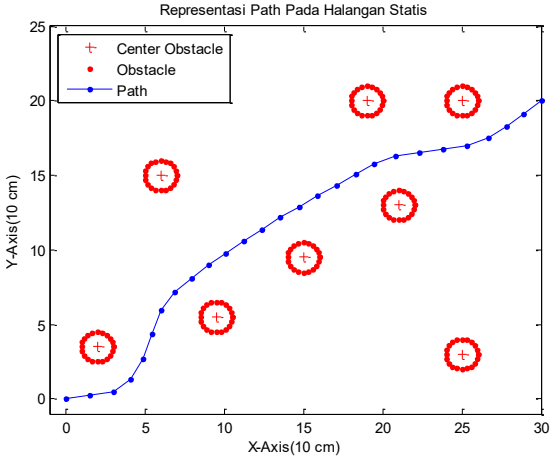
51.1223324232094	55.8057022574332
67.5349196669757	88.8233063006213
0.00497106689210991	0.280796788276310
37.6103774229222	37.7602442090327
	
58.6019928627778	54.3291260024330
100.369606532730	79.4375121197109
0.00279817957347550	0.557158075153015
38.5252733766583	37.8844655033378
	
11	**12**

13		38.691595692509 9	0.011933146334 2354	114.05650025651 7	61.514828890147 7		37.803482714484 6	0.5363845005611 18	88.047592968789 1	55.949385808803 5
14		37.680788019974 6	0.549854605098 061	88.108815663211 5	55.852405757714 9		42.020873272486 2	2.054806784684 44	156.04943650081 5	75.285567357333 8

15		37.953290554917 8	0	74.984999085282 6	52.950290371974 4		37.629711766371 9	0.568250229578173	77.582569679185 8	53.714475931787 2
16		37.717969271634 6	0.012085537058 7503	79.765338564580 2	53.683122521609 4		37.710283321598 5	0.014600944408 5893	67.192232864652 5	51.163330838937 6

17		37.6725026682869	0.503596958184311	90.432643207896	56.262628270629 2		37.8557874118240	0.290845535519528	86.7808614816695	55.5028052436775
18		38.0028311466915	0.168450750563793	97.641192344379 6	57.699520366131 2		37.7338297148582	0.063174398544010 3	80.7696066667165	53.9509254467455

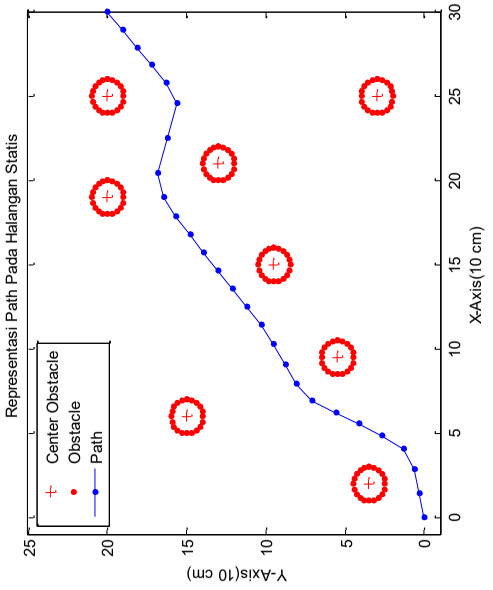
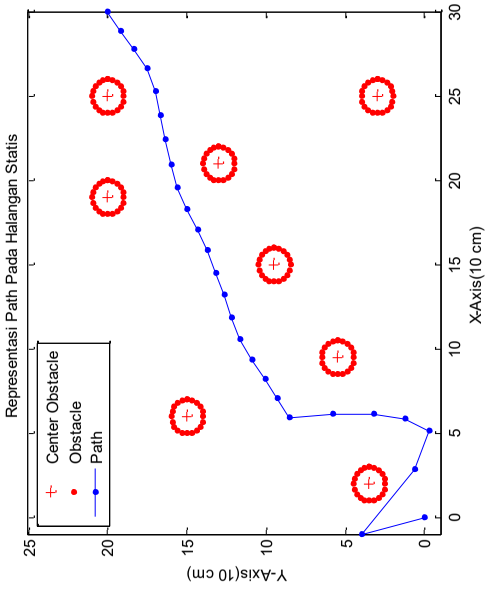
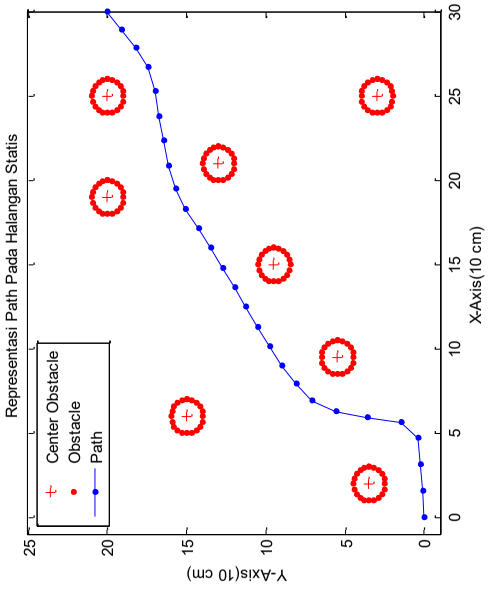
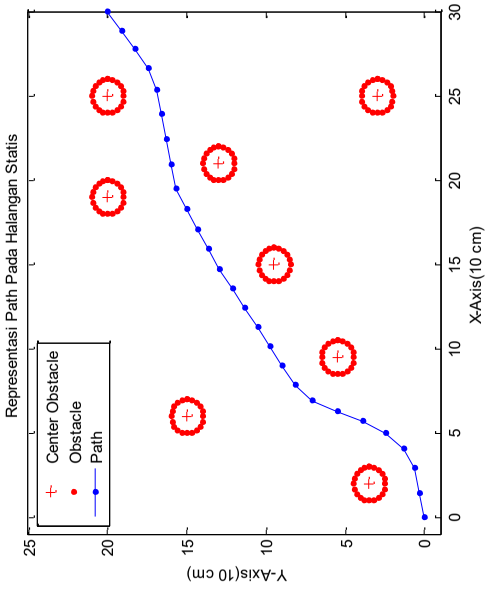
19		38.127167694933 7	0.008757533976342 38	78.465766539477 3	53.829078536805 5		37.647590114574 3	0.564801356782123	72.496783263136 0	52.711748123983 6
20		37.748519339261 6	0.319288244567404	82.069420506872 7	54.481691685203 5		37.782231426434 2	1.164053285120 02	87.356069156441 4	56.417498542842 5

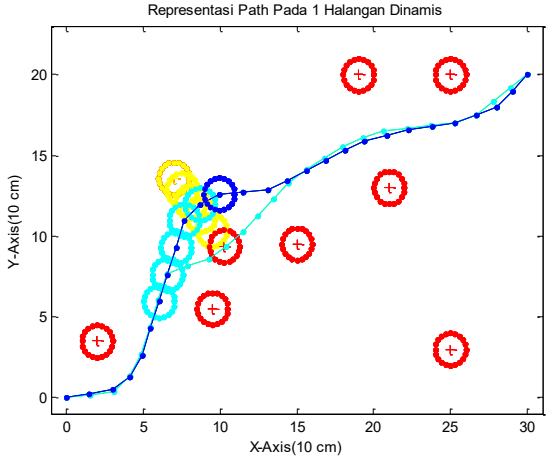
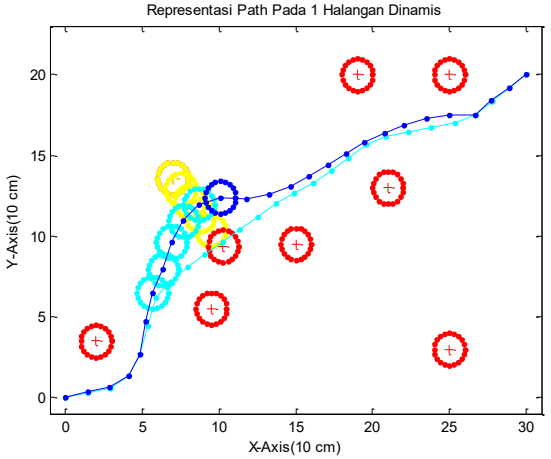
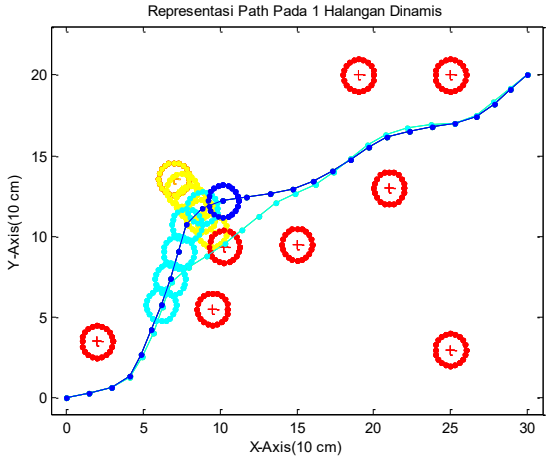
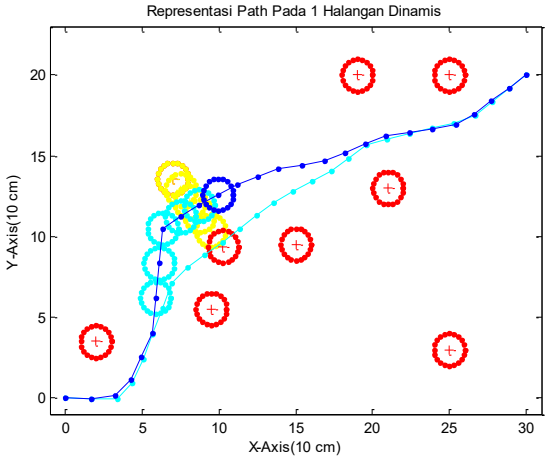
	PSO Standard									
No	**Representasi Jalur**	J_d	J_r	J_s	f	**Representasi Jalur**	J_d	J_r	J_s	f
1		45.0596010588833	0.0132844036907320	181.671749909069	81.4072354443879		38.4134759969014	0	77.8562592859660	53.9847278540946
3		37.9861760212738	0.0096144222297178 7	76.2769499119448	53.2511804258924		37.8140593186988	0.0717943246334141	76.9837250649904	53.2825986563303

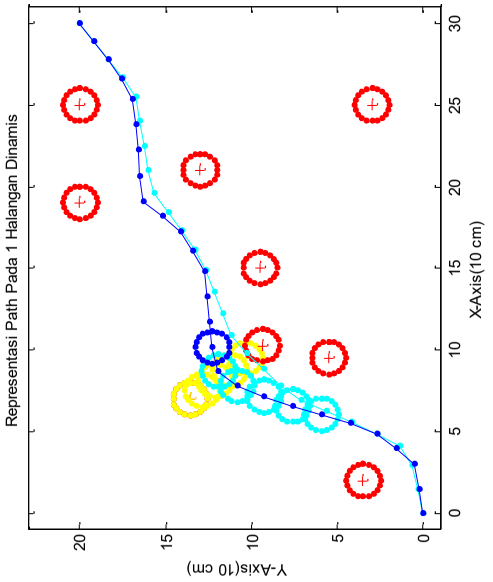
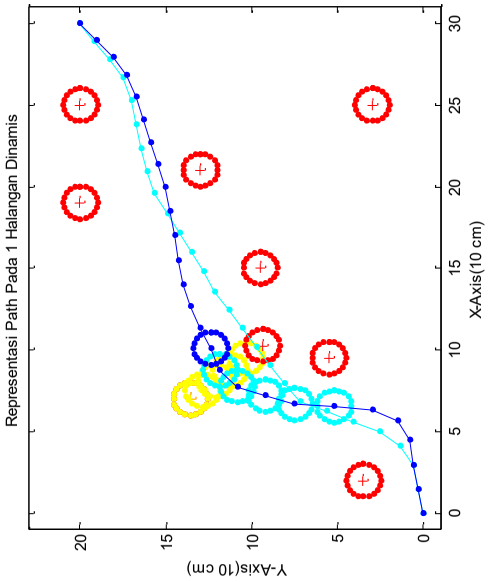
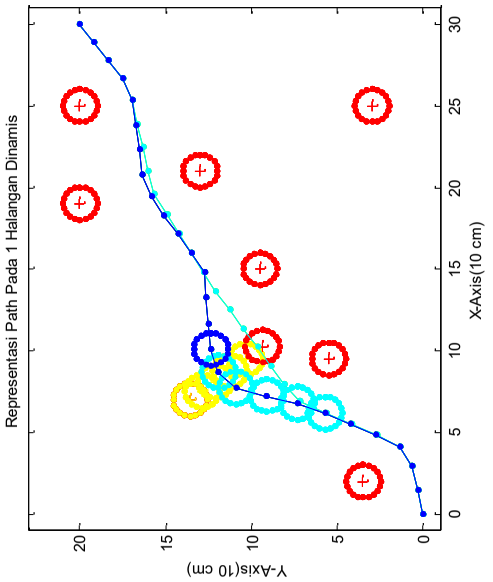
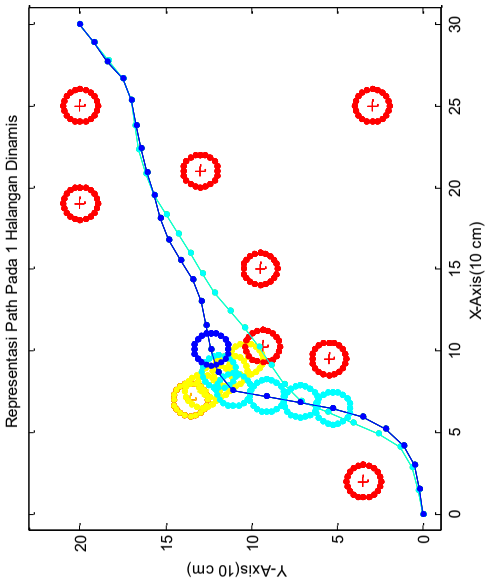
5		41.3057622788435	0.0007719177763032 55	116.507155415578	64.607965279735 3		40.3342702233621	6.82937624242727	129.000340086141	72.9637144830177
7		44.7633743062777	0.0771982822233808	186.228946712884	82.0863619310779		38.6143076109960	0	98.4657021327916	58.3074480375543

9		38.057738278931	0.0019400002032066 2	90.145528764798 1	56.088784032155 9		40.295950368146 4	0.10240474626582 5	82.861100070366 2	56.970575128485 4
11		37.622892963628 2	0.4725359909541 15	79.028853416983 5	53.901199637979 0		38.409456431778 5	0.23177322740026 7	86.243817406455 3	55.889993140469 9

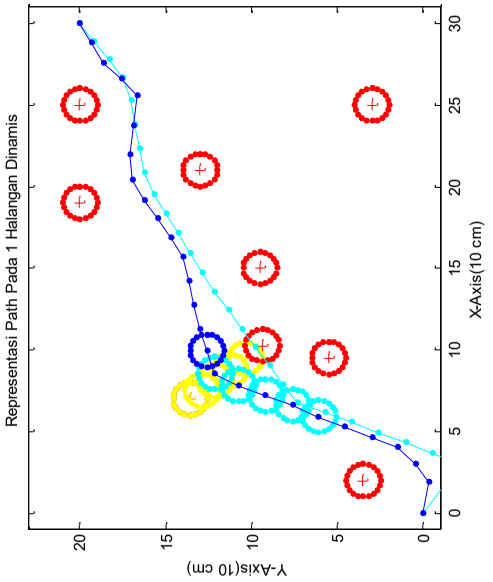
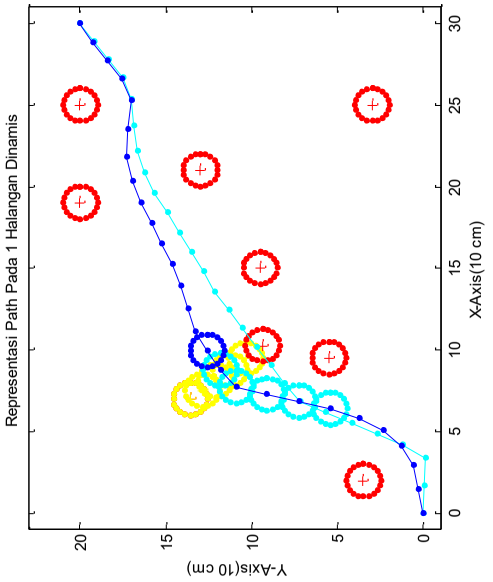
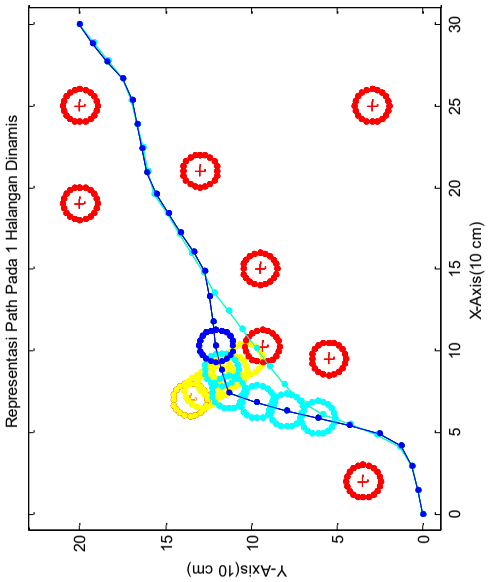
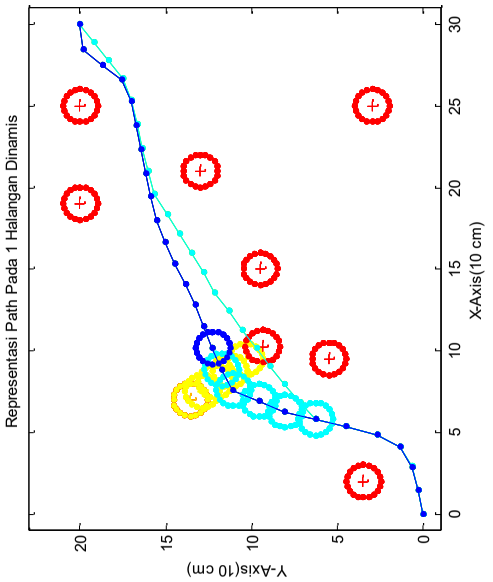
13		37.904431528797 4	0	79.386209980355 7	53.781673524868 6		43.882213816995 2	0.084505627468991 5	166.06304486081 3	77.179328416626 9
15		38.493464983689 6	0.002627918229753 7	104.48096371661 8	59.392285645266 2		37.824894566833 0	0.057113096179533 0	90.691368153549 6	56.020281293722 4

60.3444981526681	86.4733907459173
106.262681038326	194.618651367107
0.178989922230413	0.0498590247999298
38.9129720227724	47.4998014476960
	
59.6889053944997	52.6891589121041
94.8035165382223	73.9125086779325
1.84756552431627	0.153621876755068
38.8806365625390	37.7530352997625
	
17	**19**

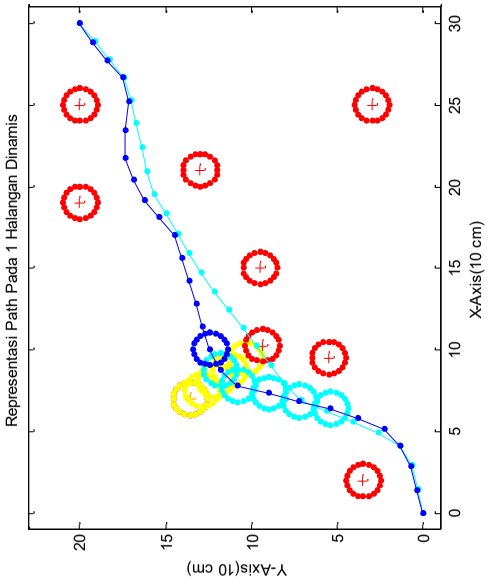
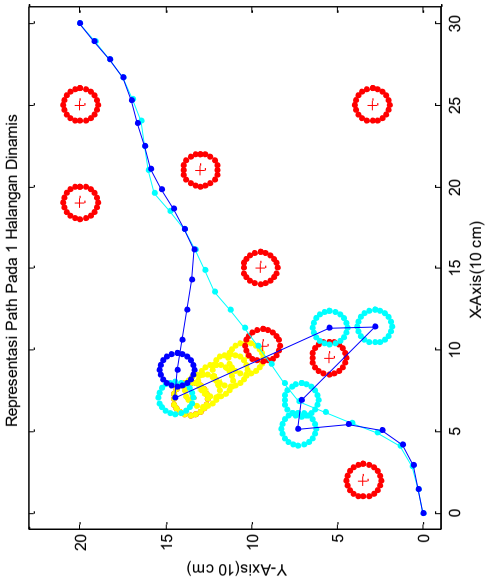
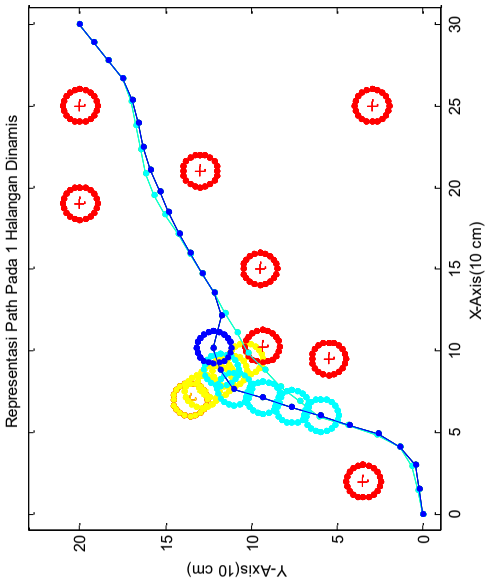
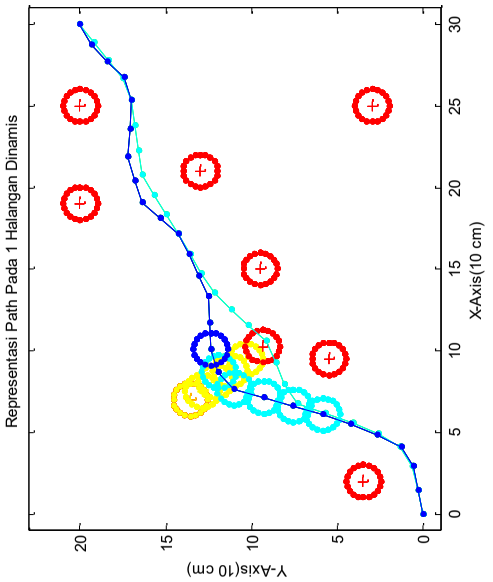
	PSO yang diusulkan					AIW-PSO				
No	**Representasi Jalur**	J_d	J_r	J_s	f	**Representasi Jalur**	J_d	J_r	J_s	f
1		39.4838891500707	0.000763045597307333	103.945795378680	60.2738112714040		39.4729409330982	5.21079837244881	139.565648392574	72.5968689840618
2		39.0871671351824	0.0516078754912197	100.975835550700	59.3339421208136		40.2946502075935	0.432909910828312	109.434267977932	62.6144137140082

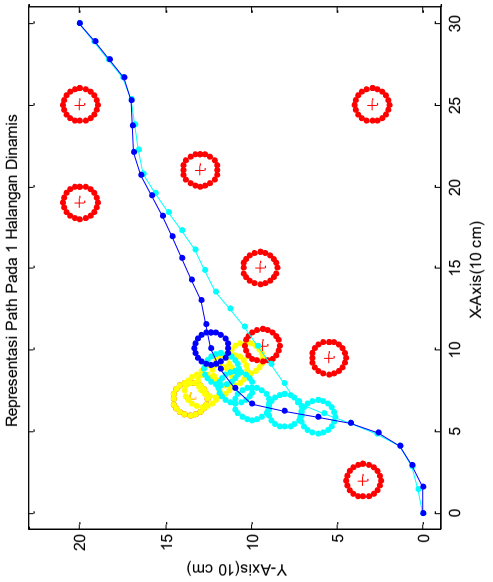
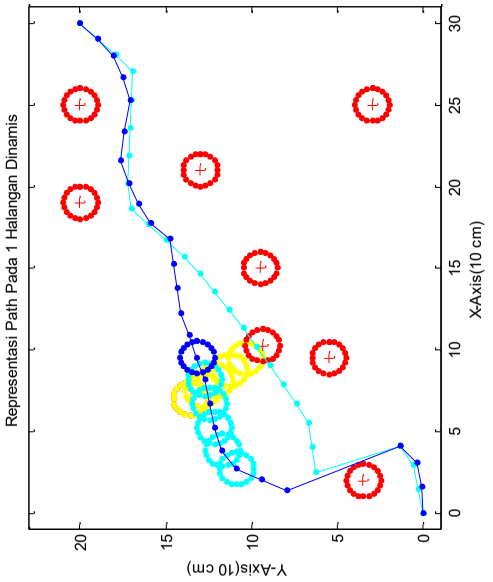
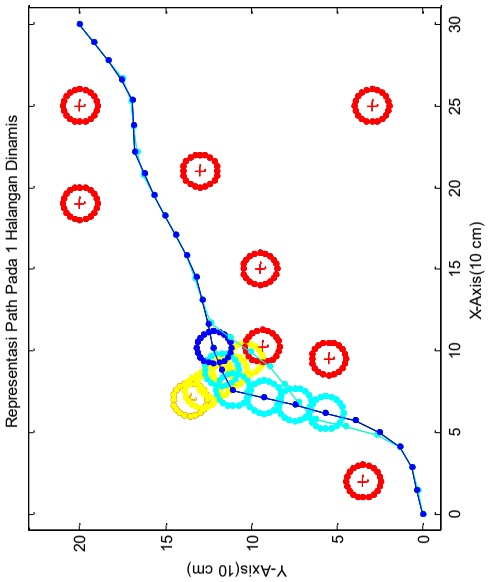
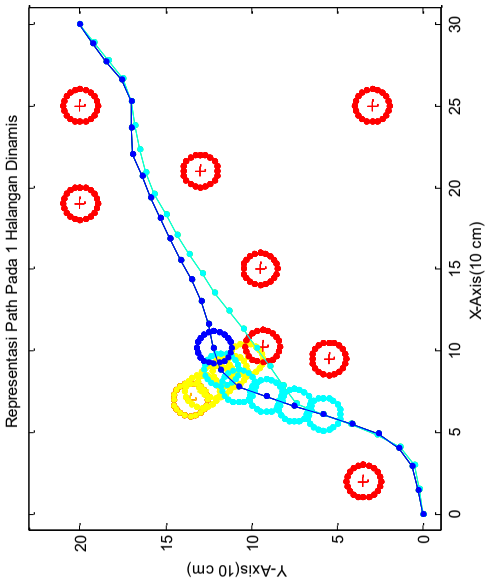
69.4680529359300	61.5372070382213
133.055826191549	103.320916993996
3.12208480032354	0.432654535249326
39.7348028972966	40.4403691041727
	
62.5509493006438	60.5378154254575
111.223150301257	104.331199658122
0.843601926989841	0.0571936386550354
39.4627173134026	39.6143818551780
	
3	4

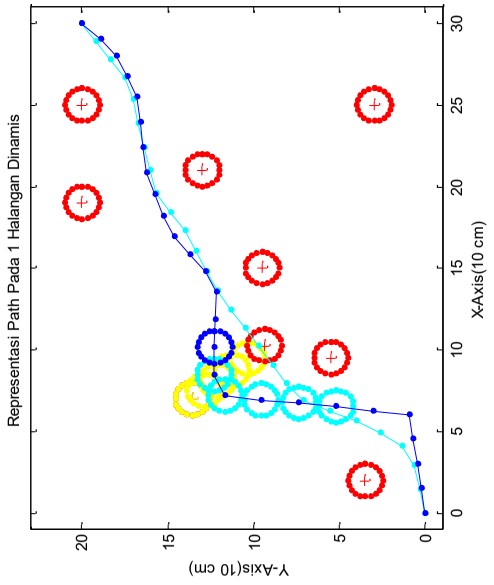
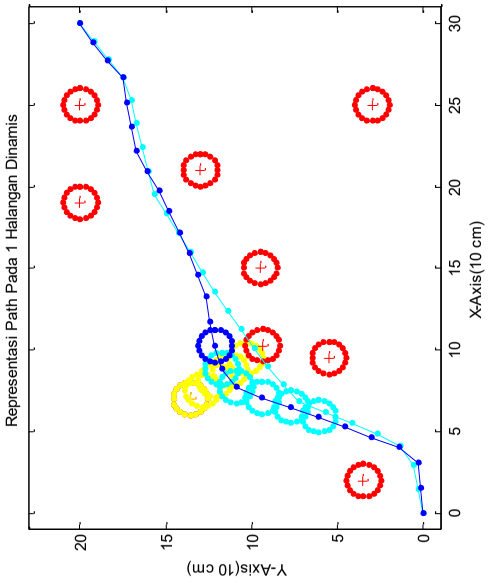
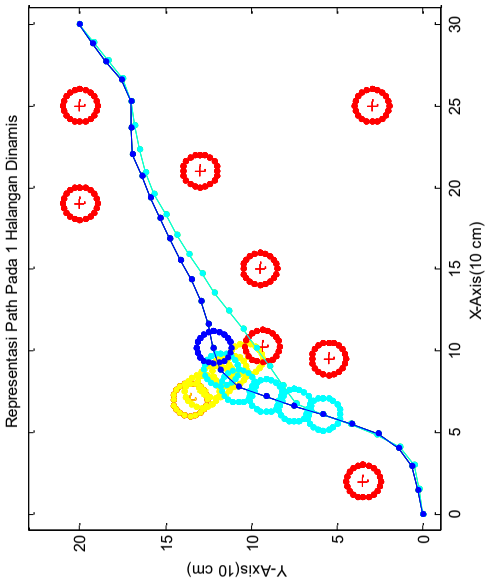
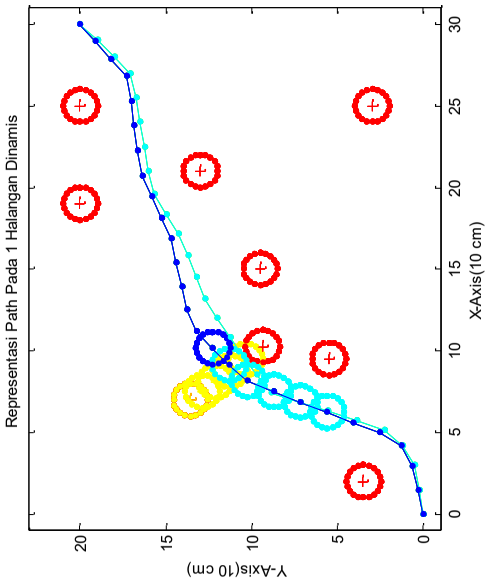
5		39.736308597978 0	0.27838482477483 3	129.91522715813 5	65.997738854379 9		40.211721699159 6	7.0269691958895 2	142.76492177594 1	75.791675250237 2
6		40.232956718232 2	1.3209545752546 2	116.66008778243 3	64.885928849973 5		40.890093371086 3	0.34471654195114 4	132.24478032340 8	67.683765977719 0

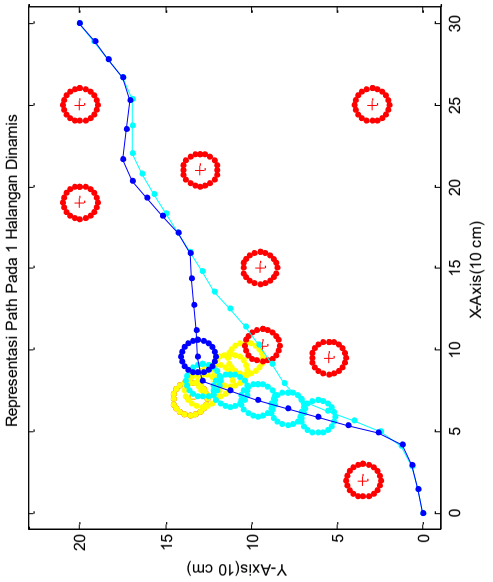
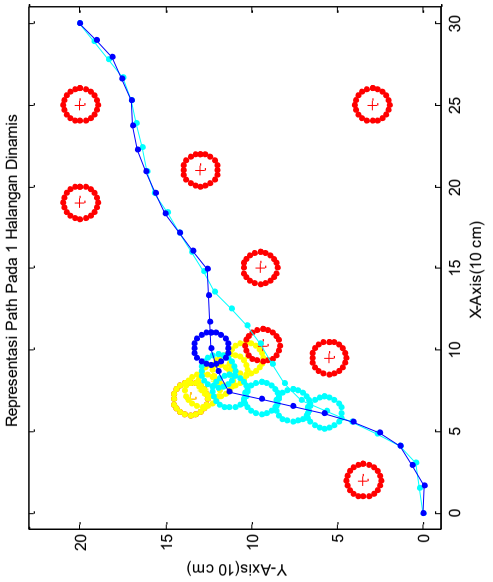
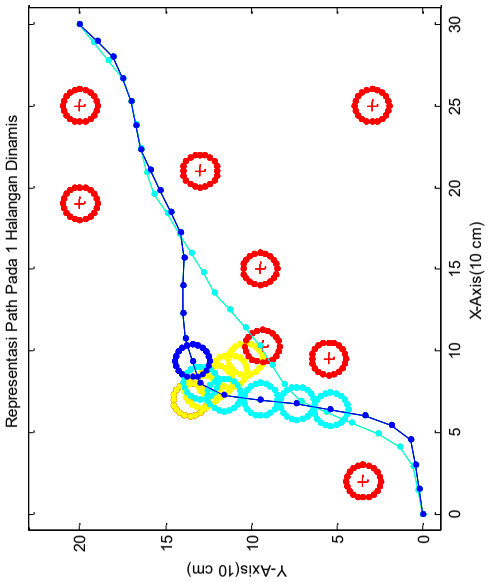
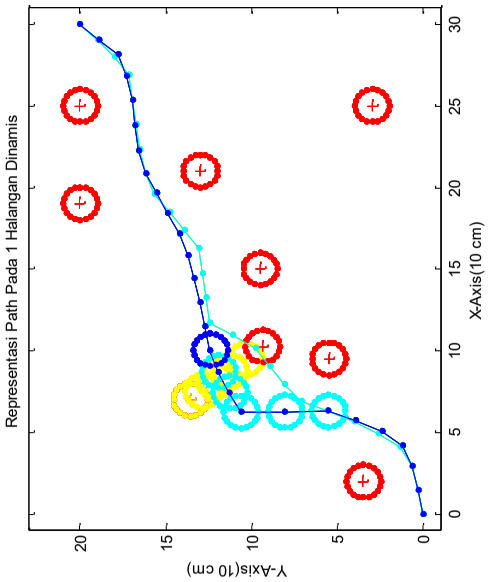
72.2209758602409	61.4139320139549
134.265223344306	107.371271292022
5.29706618524120	0.390307692015313
40.0708650061386	39.5493700635352
	
62.6412179843521	64.2362328841258
107.529815914457	113.880958715923
1.49155068031430	2.02856667737802
39.6437041211464	39.4314744635633
	
7	8

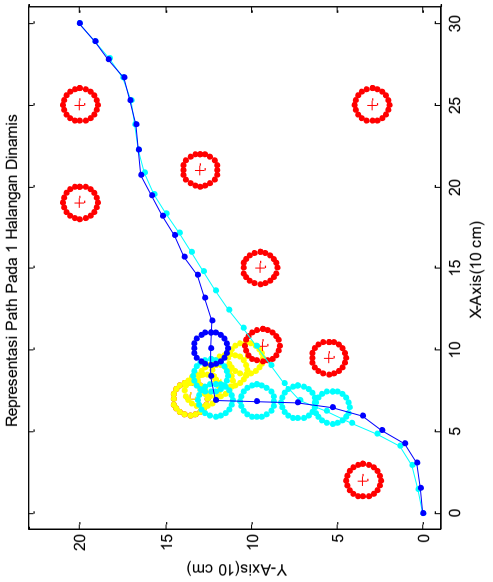
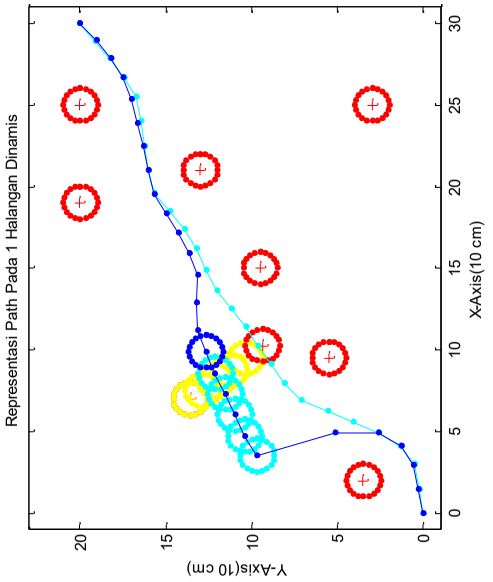
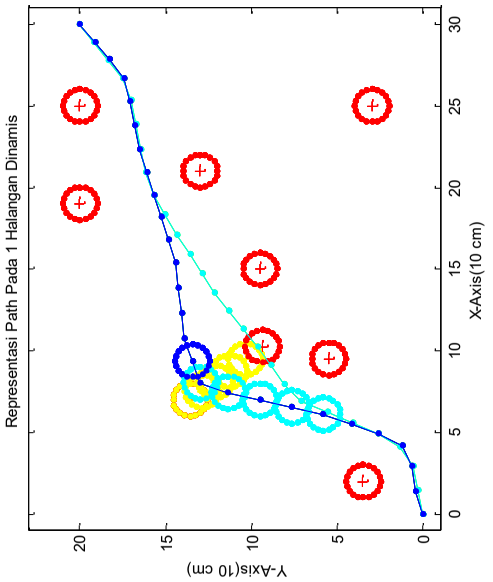
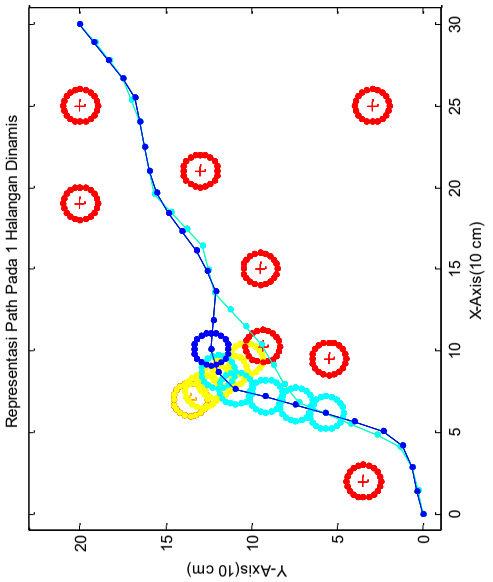
9		40.473027498611 5	0.090745110872307 9	126.97232887952 6	65.958238385388 9		39.649445315589	0.355846589108717	121.772035953693	64.359699095406 2
10		40.092667484748 5	2.304400110420 47	114.39030395058 7	65.275128385286 4		39.712378580844 6	2.355463825324 16	97.907501312460 5	61.649342668660 8

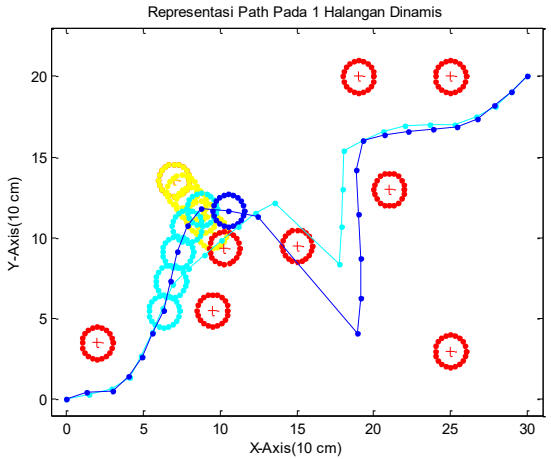
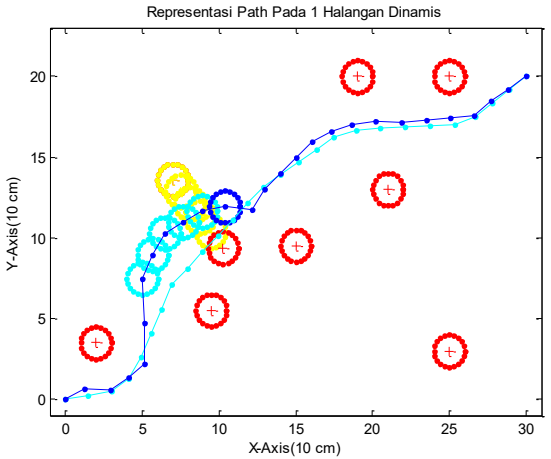
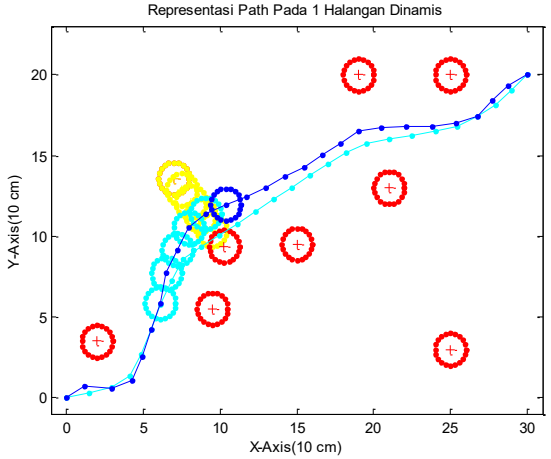
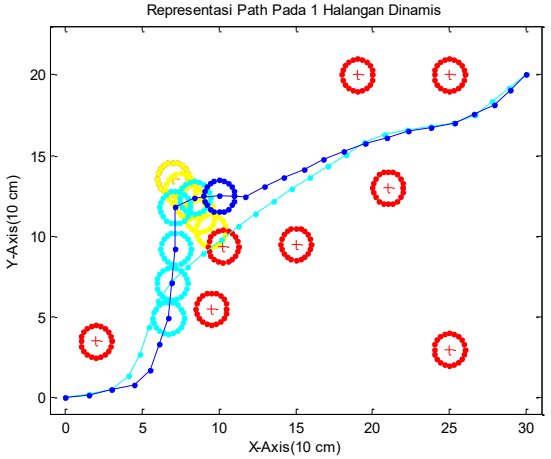
66.0362565750317	95.9758358988369
121.417595133154	217.720358994071
2.26302170945194	0.357485122953518
39.4897158389490	52.0742789770692
	
58.7670026574553	68.9864687816391
97.4693572257959	145.236981658210
0.0450991841219484	0.00750727477374813
39.2280320281742	39.9315651752234
	
11	**12**

62.6751689321606	85.5082605081063
106.731339573665	201.029166988862
1.87657212317483	0.521851066593473
39.4523288942529	44.7805760437405
	
60.0052900073636	60.8503367628980
100.194746637418	101.922820767430
0.640263585014544	1.43739411721203
39.3260770948654	39.0283784921999
	
13	**14**

71.1229877758754	58.4518380599857
144.115916057592	93.2842320504847
0.126707533145067	0.656008292330306
42.1730970312120	39.1389833575585
	
60.8503367628980	56.3113870359526
101.922820767430	87.0517881634994
1.43739411721203	0.00433709631958656
39.0283784921999	38.8966923069331
	
15	**16**

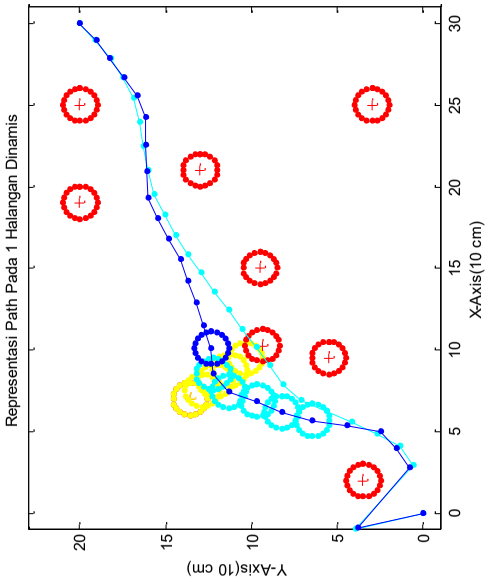
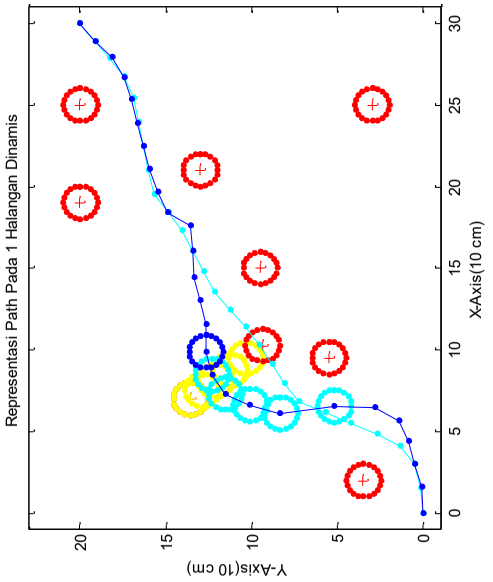
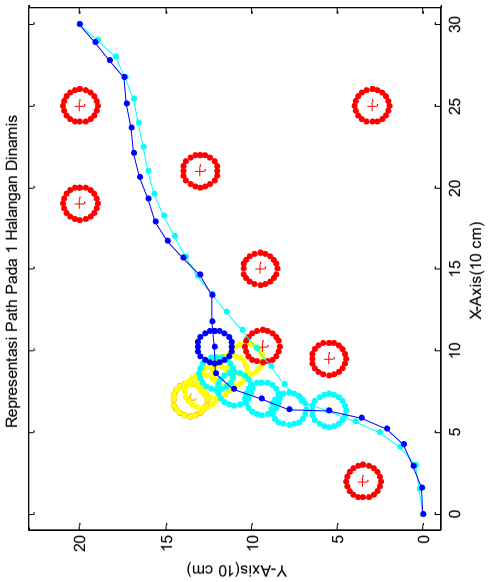
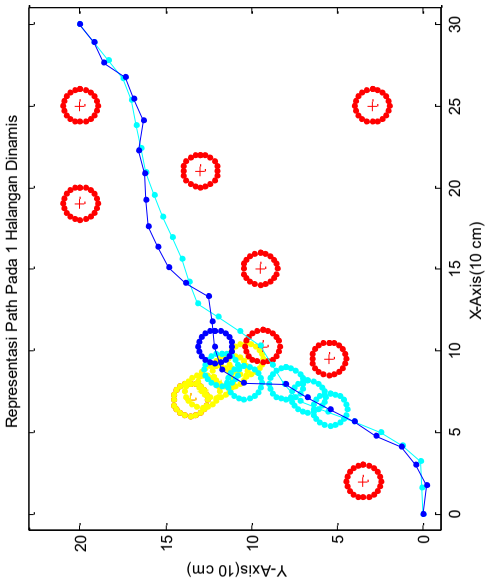
67.7477927223535	64.9071250887328
132.952740538921	122.278970738801
0.380395778060549	0.523975400188821
40.7768488365089	39.9273555407838
	
60.5009894612103	63.1820542254026
98.1678994356764	113.407386971857
0	0
40.8674095740750	40.5005768310311
	
17	**18**

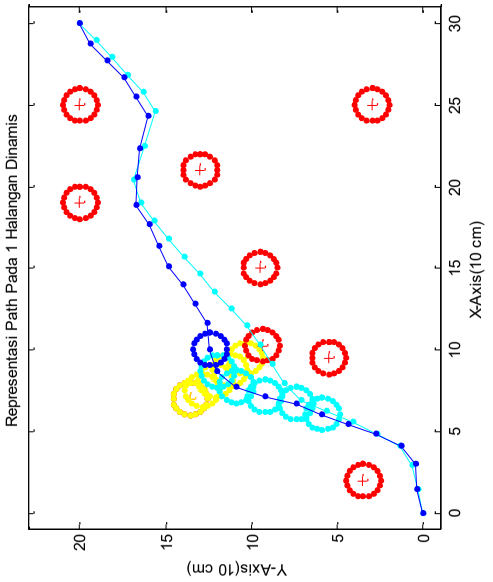
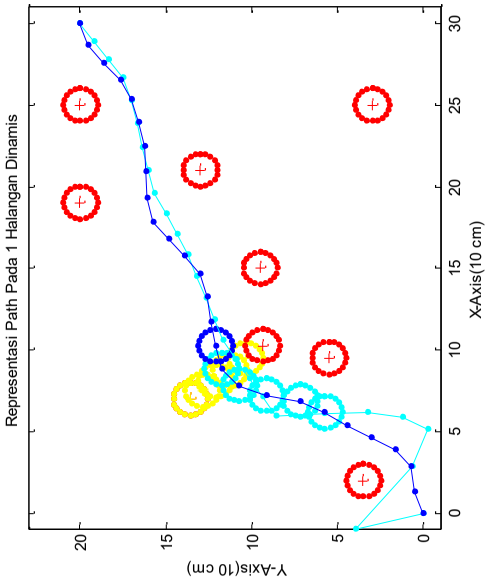
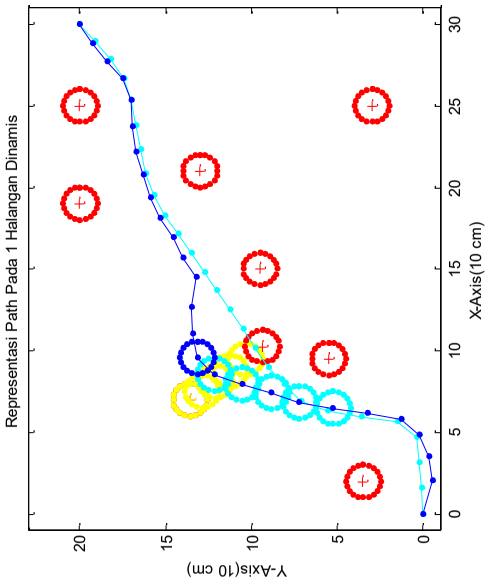
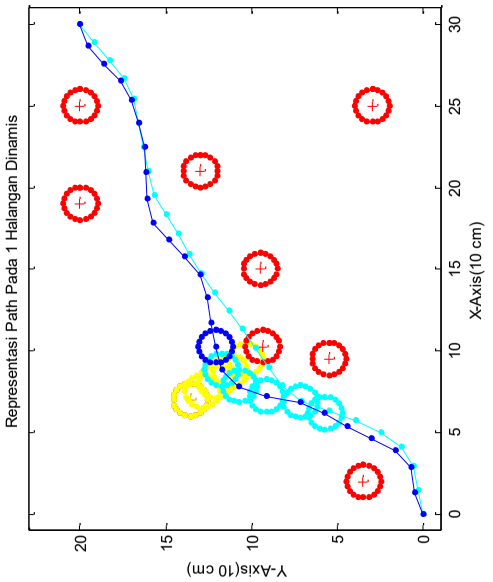
71.3042116085252	68.6826217281029
147.197574598865	131.773482069743
0.624432397258539	0.186567692110917
41.2402642914936	42.1413576220434
	
60.1446043291055	65.0295311311946
97.6868976254847	119.912263897035
0.346915049324328	1.28931604998896
40.2603097546842	39.7577623017988
	
19	**20**

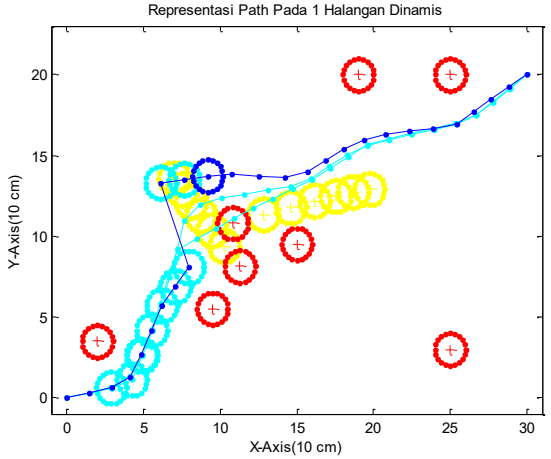
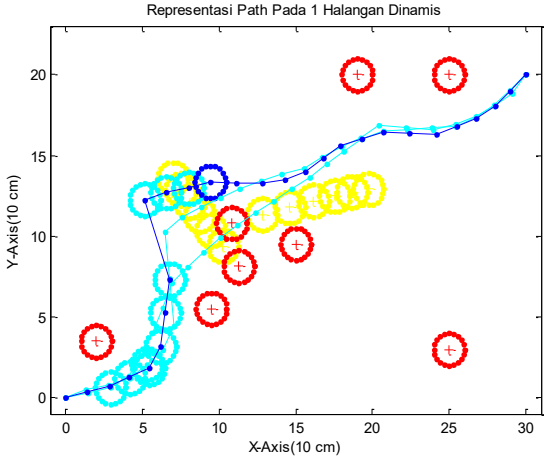
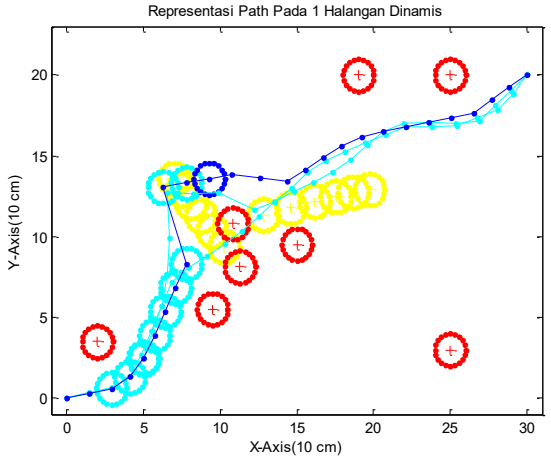
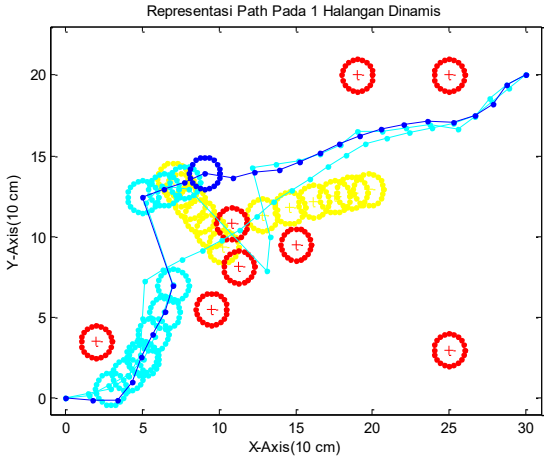
	PSO Standard									
No	**Representasi Jalur**	J_d	J_r	J_s	f	**Representasi Jalur**	J_d	J_r	J_s	f
1		53.714235589240	0.014703225780317 0	190.990694424324	91.927077469569 1		41.074854325686 3	9.787915503847 15	197.936208812356	90.450011592004 7
3		39.369472649747 7	0.880839178839459	144.060912575632	69.062494343713 6		41.059879032591 4	2.164591784272 30	128.608368412755	68.946144499414 7

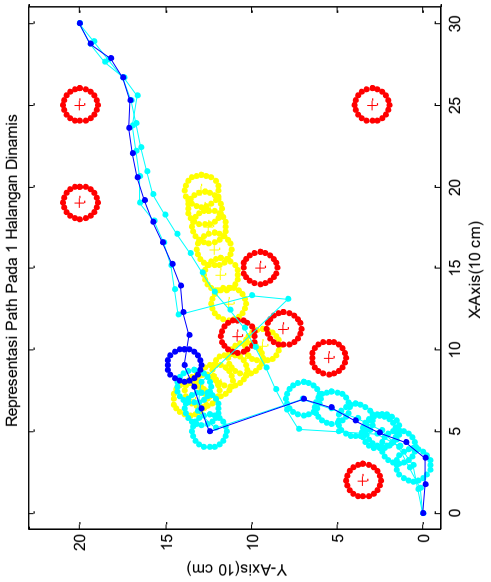
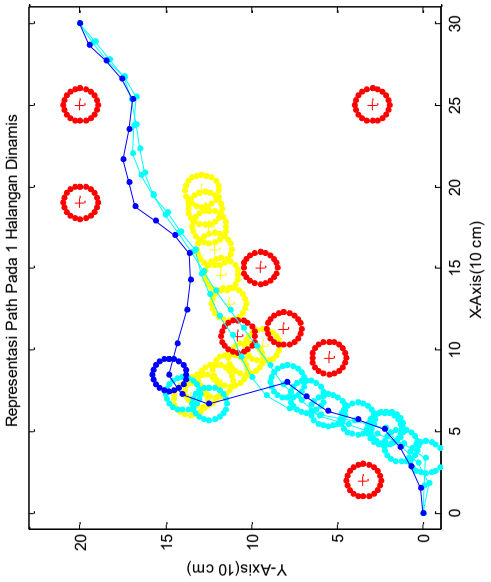
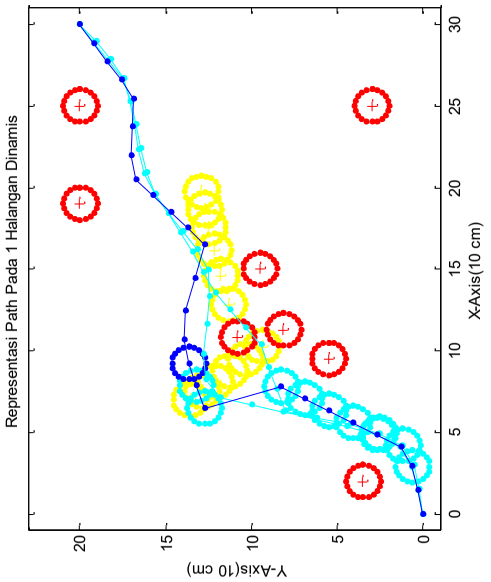
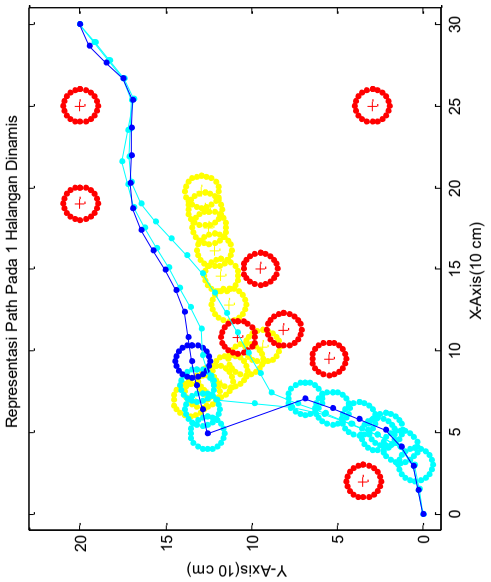
5		41.766824883111 10	0	134.285590063340	68.623942895779 1		40.923228881127 8	0 158.796647256933 72.6825583325143
7		40.923228881127 8	0	158.796647256933	72.6825583325143		40.113328297641 3	0.501067715306354 150.676344180786 70.7496648491048

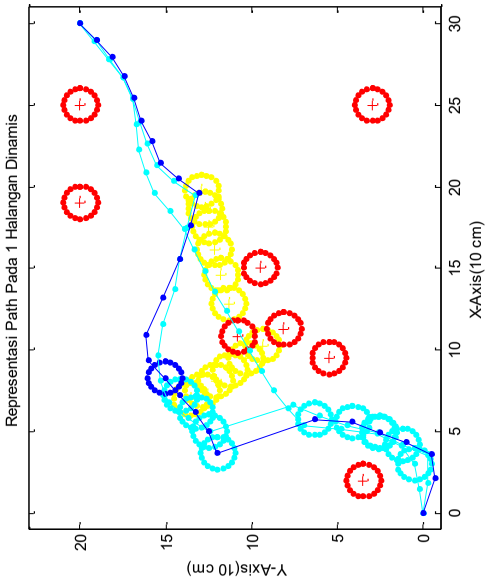
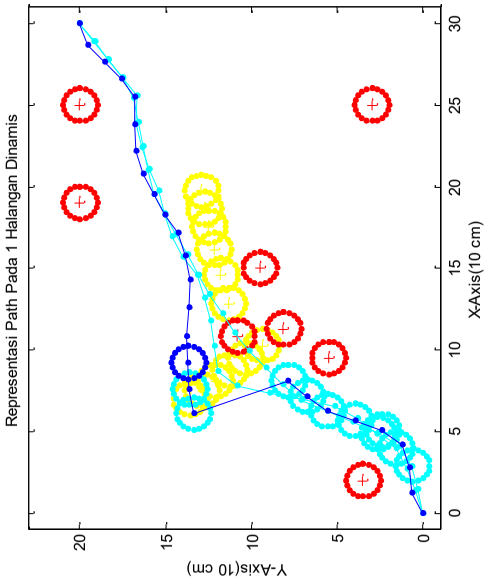
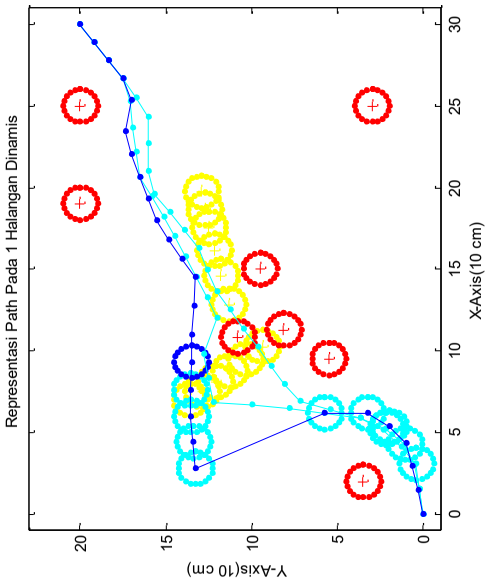
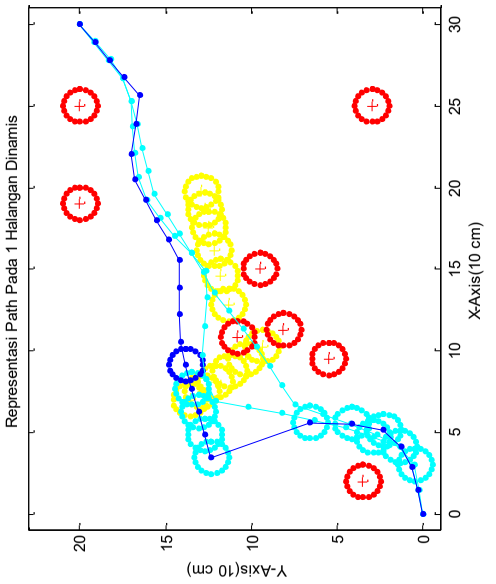
9		40.674949267265	2.5422560796061	168.54993013204	76.927191373281		41.449756287494	1.6158206955879	158.65762422138	74.797101827359
11		40.462879008816	8.6173032150272	162.61950331240	81.604082886325		40.837079465720	0.5986528325380	160.35086156772	73.505904611803

97.5604290184975	73.5101215360620
232.825141343821	153.071240977888
5.51126451559099	1.23555034259555
45.4841362341423	41.6603229978889
	
74.4788685768037	97.3355166477140
154.024702852419	226.654211932076
3.43527829301713	11.7134285999750
40.2386497133027	40.2912456613238
	
13	**15**

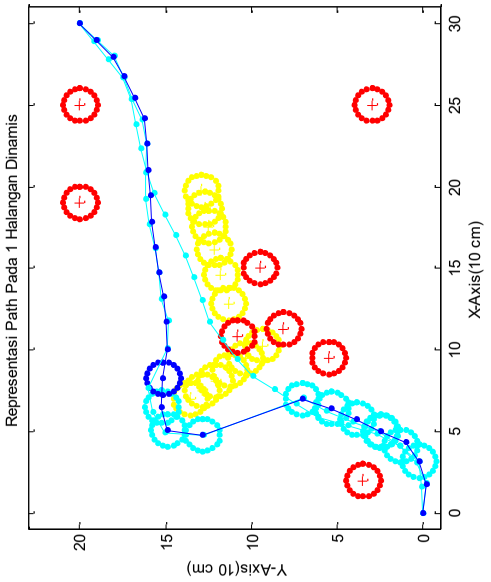
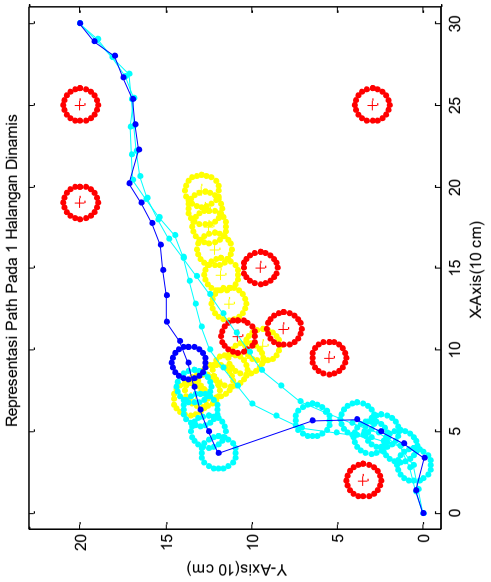
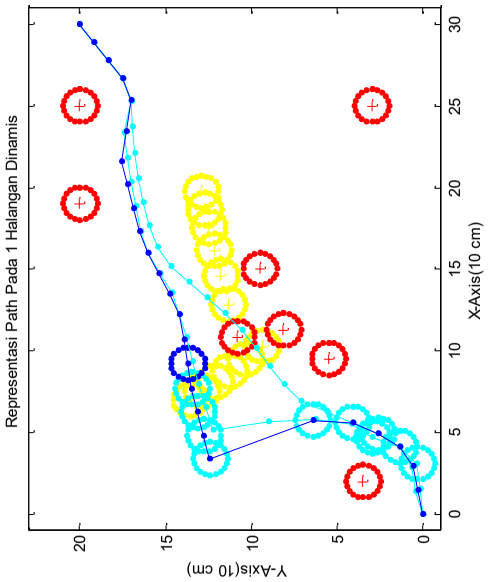
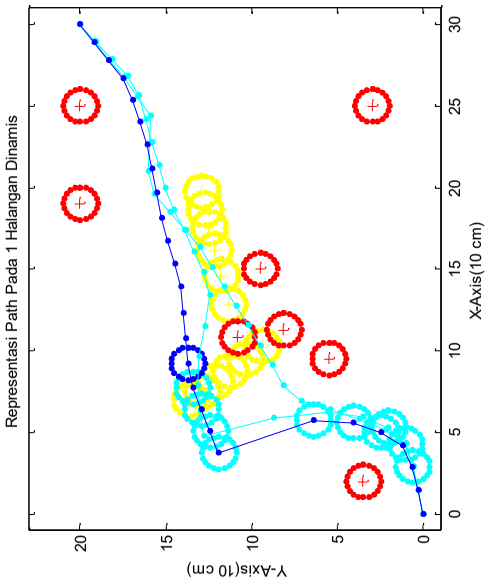
74.1936963215536	79.1824211141583
164.442439015163	146.434275563224
1.20315677896020	10.6939365080920
40.1020517395607	39.2016294934214
	
72.6496329145348	79.1824211141583
156.337536733514	146.434275563224
0	10.6939365080920
41.3821255678320	39.2016294934214
	
17	**19**

	PSO yang diusulkan					AIW-PSO				
No	**Representasi Jalur**	J_d	J_r	J_s	f	**Representasi Jalur**	J_d	J_r	J_s	f
1		43.209404187393 7	12.087196958187 9	144.145837119943	87.729414497568 7		43.617193126744 5	6.9399327674586 3	150.441014717690	80.645328837741 1
2		42.659459626250 2	17.600308100356 7	124.396708973298	88.249027245599 0		44.107221268275 3	3.2981755859091 8	190.841402998071	85.573677453798 6

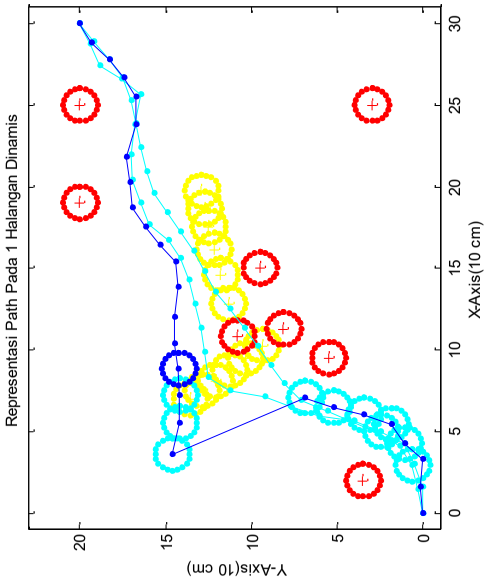
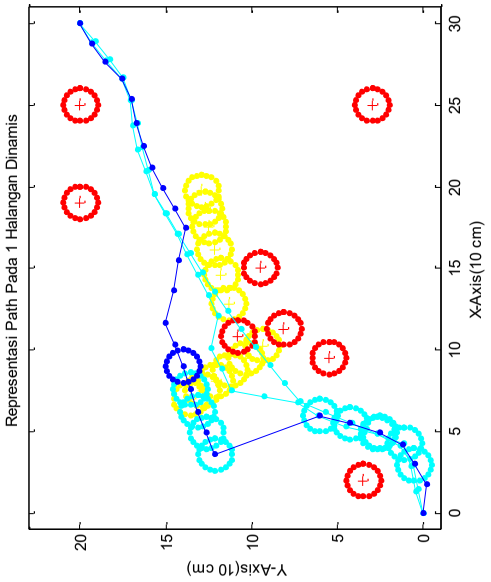
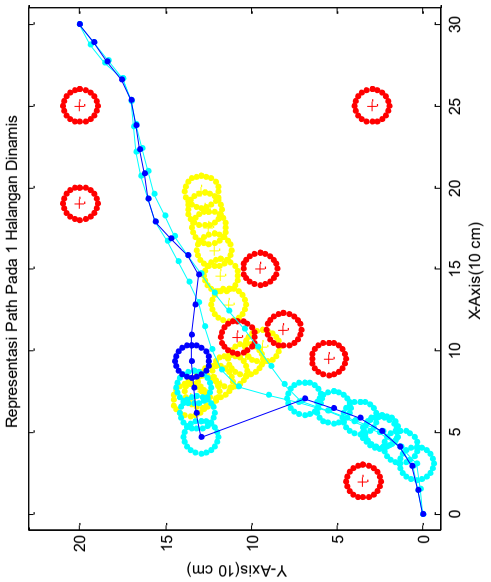
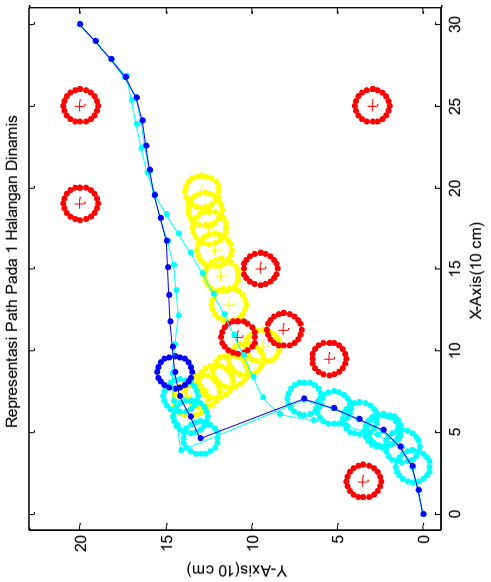
105.025693548264	96.3911355750040
243.719602804422	194.647491898921
12.0412145999164	13.6145789160759
44.2405583874631	43.8470582791439
	
92.2174116831814	75.1964509538711
162.318601249056	140.778874822726
12.3089513121357	3.19733965543910
43.3867750900080	43.8433363338869
	
3	4

85.5431147754382	86.9101156976861
181.246435440708	196.264177255691
2.76996134184020	0.0385101288651502
46.5238663454564	47.6187701176827
	
78.6246215717760	71.1410953384235
140.286445672904	130.833682360931
3.32399029700373	0.00265699208817249
47.2433421401915	44.9717018741490
	
6	5

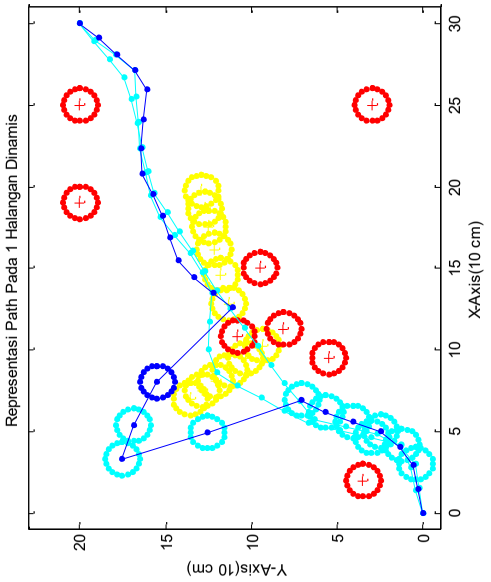
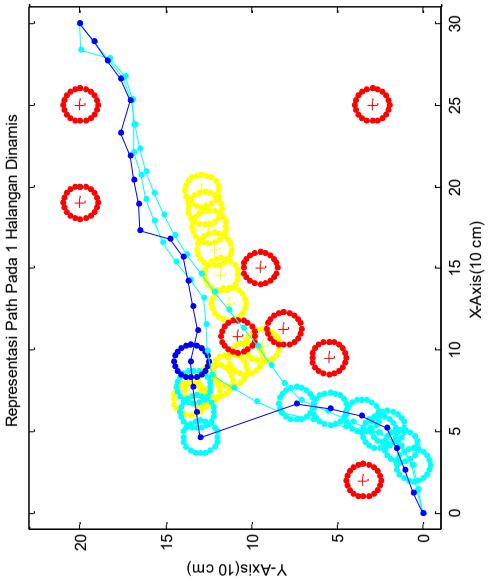
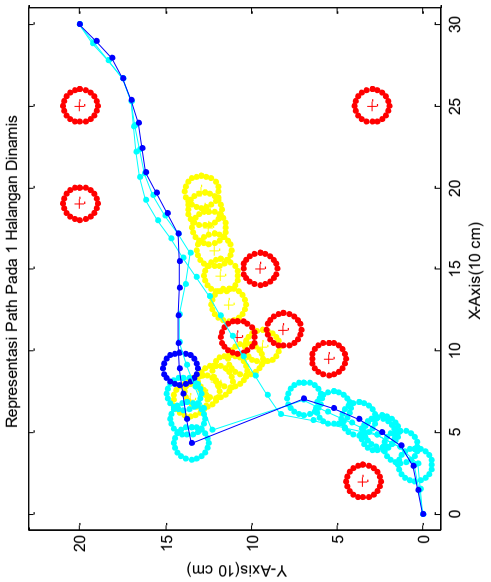
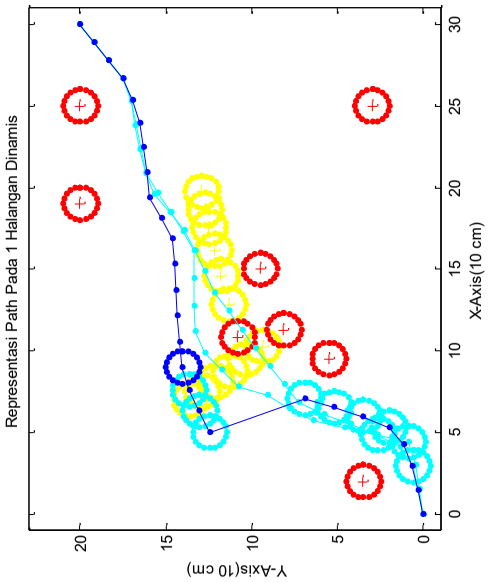
7		43.858165183265 0	1.4260558903783 5	119.59338530834 5	69.202898135312 4		47.047647012776 4	17.744321309524 4	222.83995058530 3	109.35995843936 1
8		44.318472159210 4	2.5521299339308 2	126.24355311932 5	72.119312717006 2		46.448221257825 1	3.7023316310458 2	195.50747547251 9	89.252047983374 6

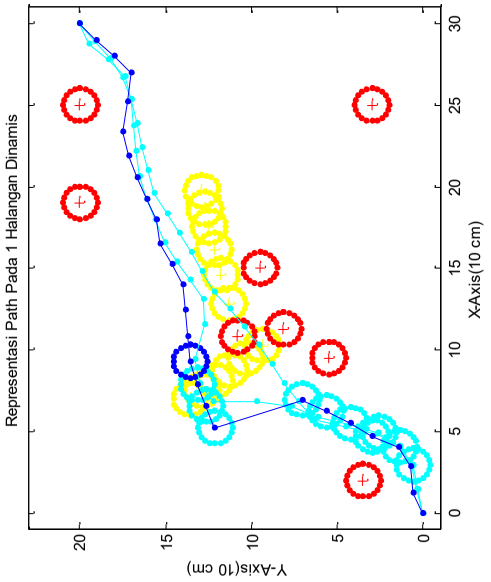
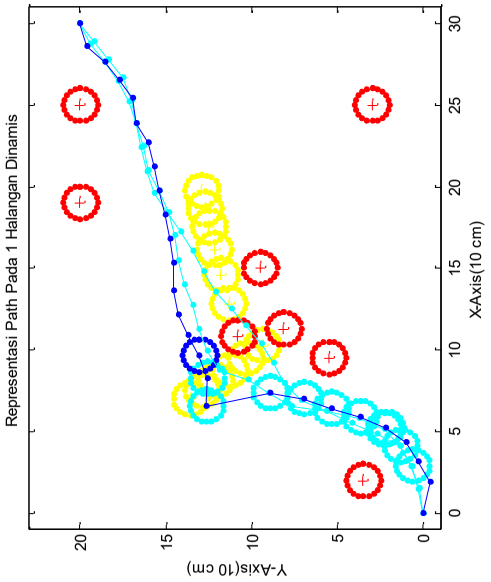
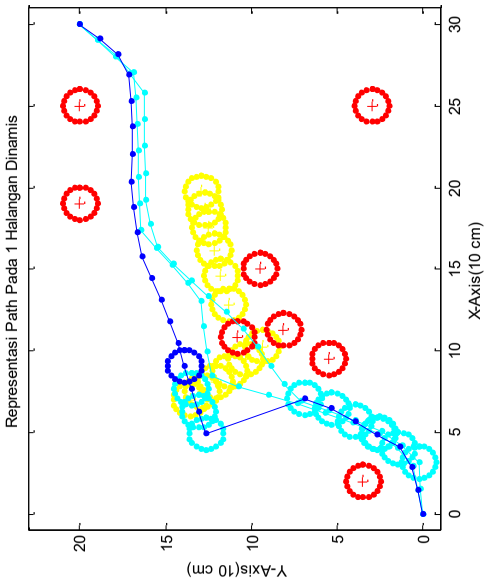
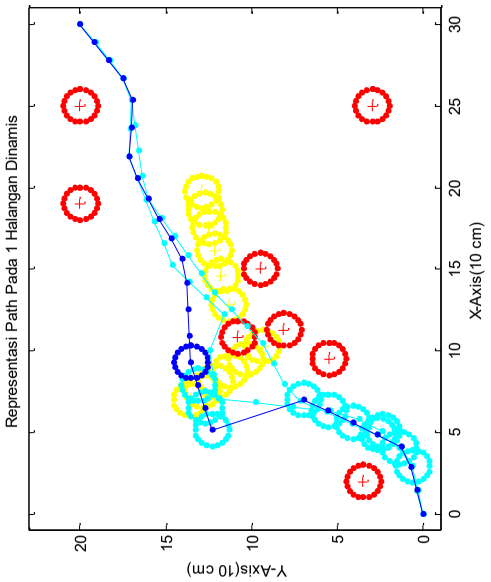
79.9193818241914	88.9356317808382
152.869375393256	217.061609882263
3.60206230804245	0.471434105456132
45.7434444374977	45.0518756989294
	
69.4391412631736	64.2564128100239
123.735289034558	102.745624320383
0	0.00158283016862271
44.6920834562620	43.7057051157786
	
9	**10**

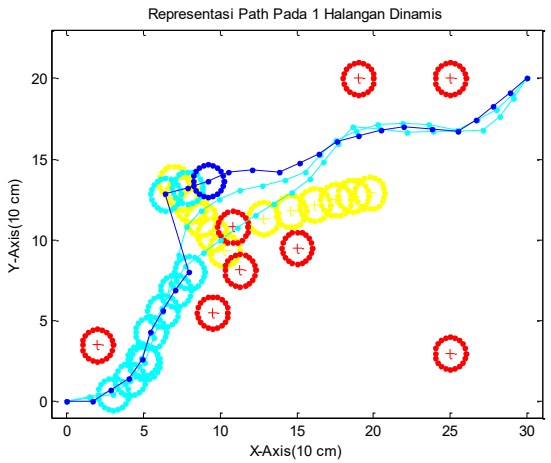
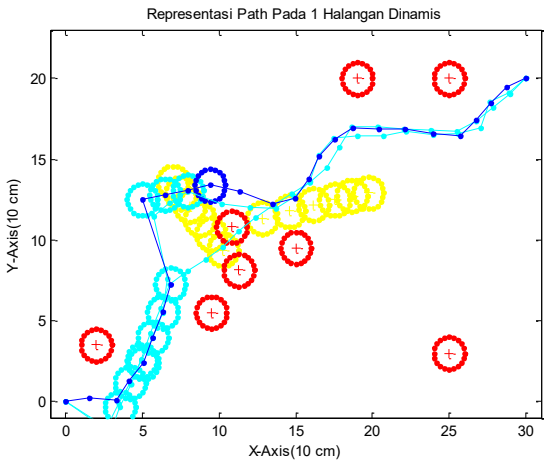
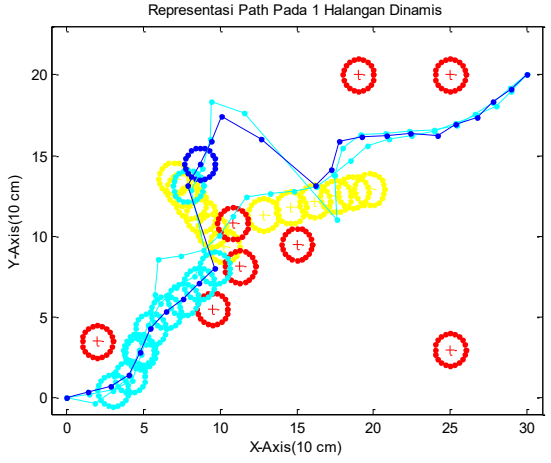
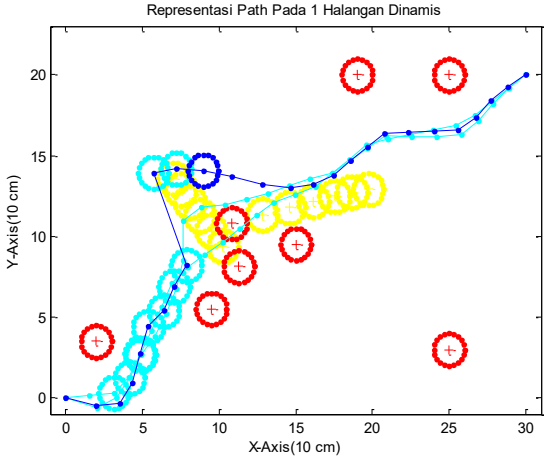
11		43.630664176659 5	11.376952049264 3	163.45277562920 5	87.698171351764 9		48.516726987581 0	6.1447502993280 9	199.49227245735 5	94.559931778380 1
12		43.861796856341 0	12.053735440451 7	104.45348504313 6	76.806229305420 0		45.522759432335 0	6.1813035572433 9	239.72783723008 6	99.649630435595 6

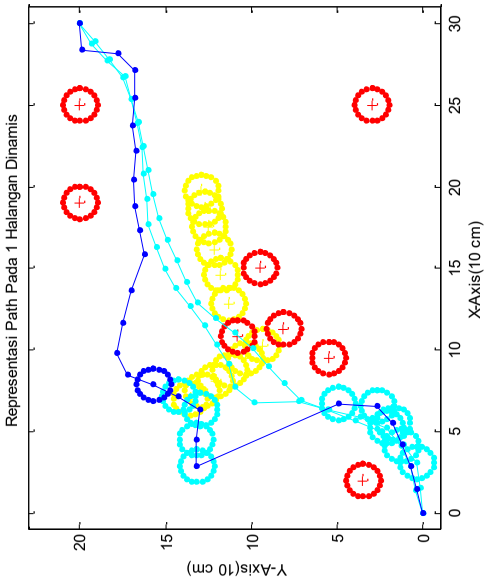
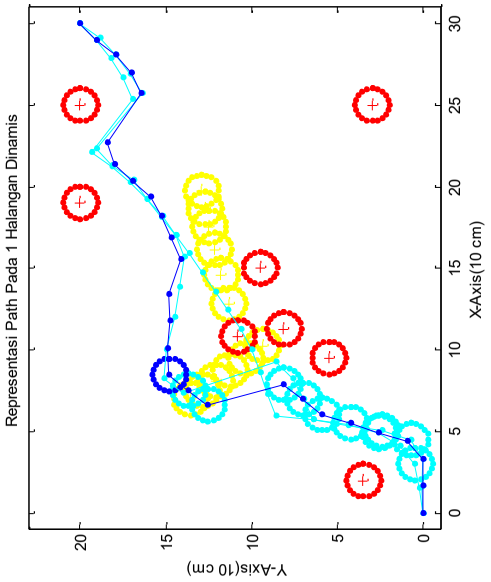
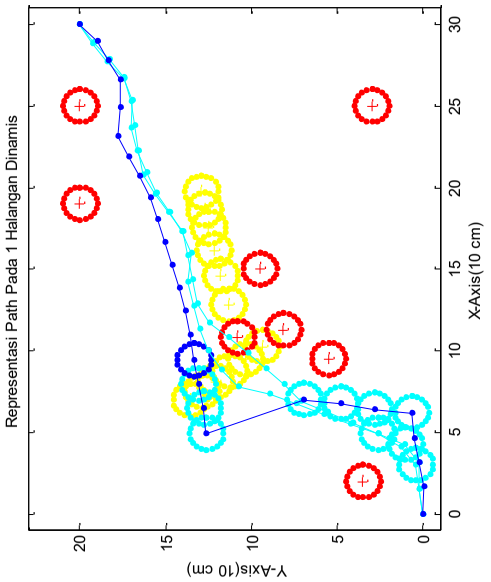
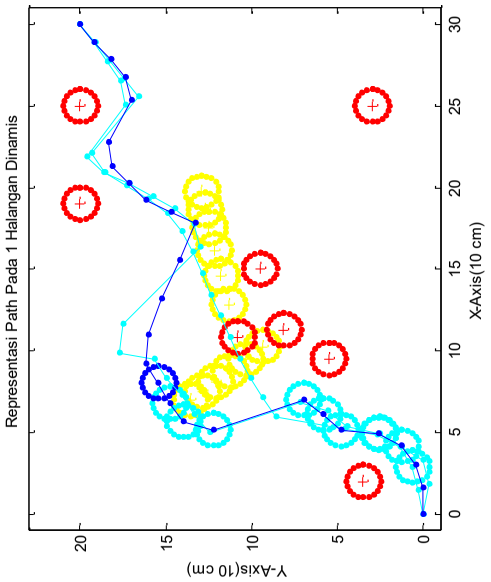
93.8917312482350	78.6761072559027
216.045928481271	163.286210876418
2.20070559957466	1.03747740391602
48.4818399524061	44.9813876767031
	
75.6887115801027	67.7691368601512
136.242352736493	109.709440408367
3.89500462953582	1.59197944425195
44.5452364032682	44.2352693342258
	
13	**14**

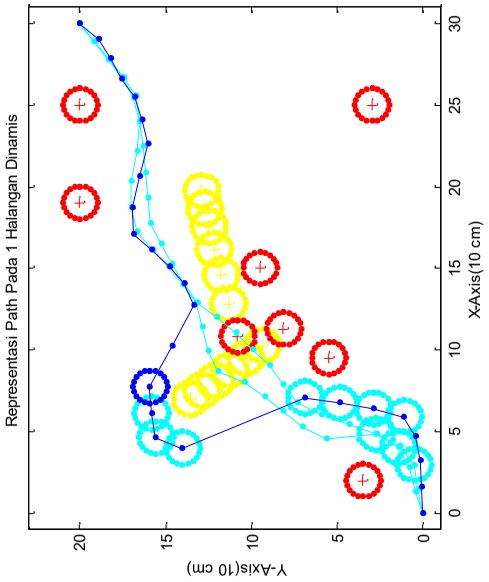
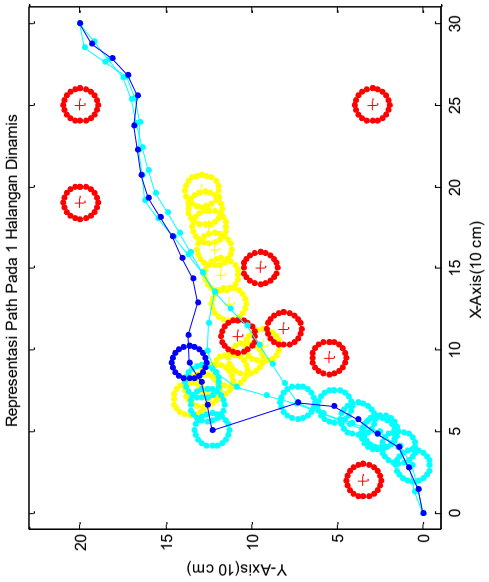
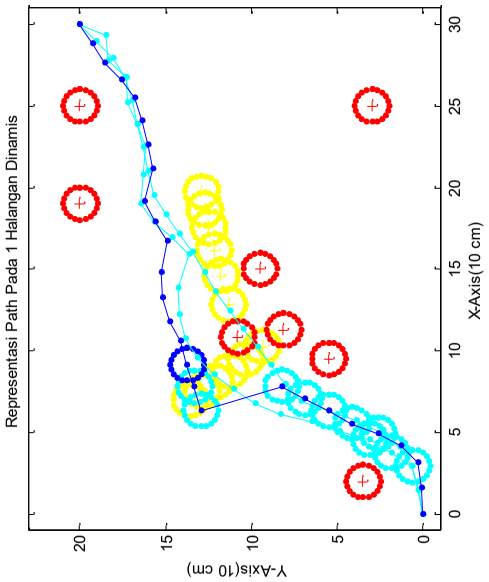
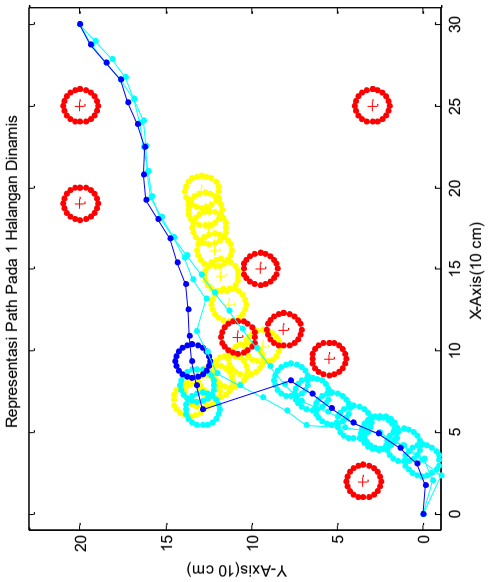
15		43.036442065226 4	3.023911524215 50	123.630048937533	70.786363769485		47.187555272346 0	15.858669024591 2	254.834357644332	114.013095825804
16		43.927789113498 8	7.295332544601 30	148.633976987938	80.949917055687 6		53.971171015885 1	4.938107031515 67	211.436185259186	101.196515099238

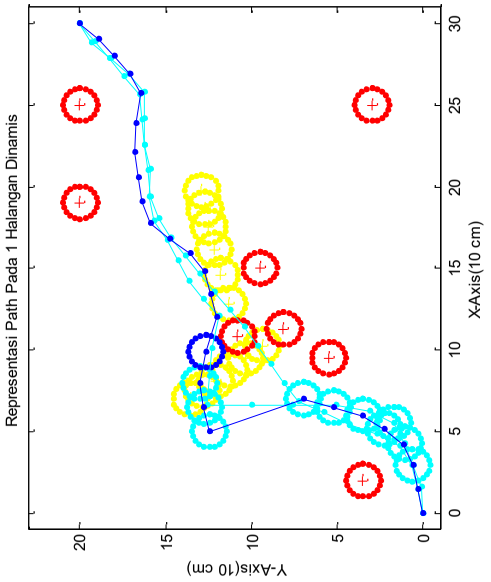
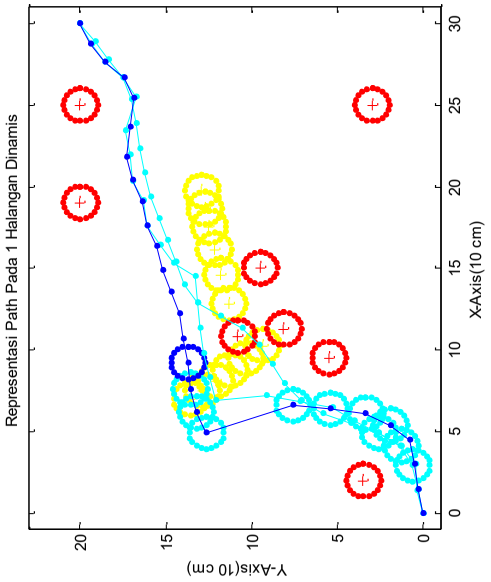
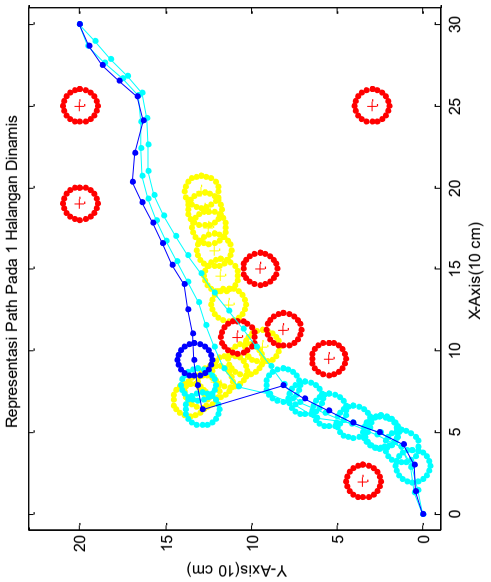
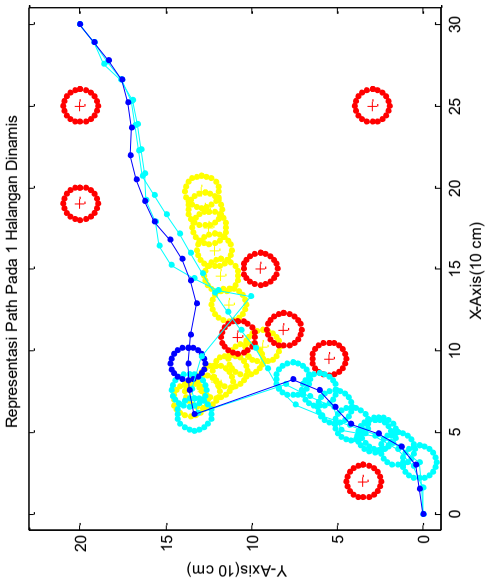
104.642074240331	106.244737577804
192.261119123739	218.596458175685
11.9103880722933	17.1783952559899
54.2794623432902	45.3470506866766
	
70.8244886216209	69.8481422099426
120.522530528008	122.304921737530
1.56997450837979	1.81753581952412
45.1500080076394	43.5696220429125
	
17	**18**

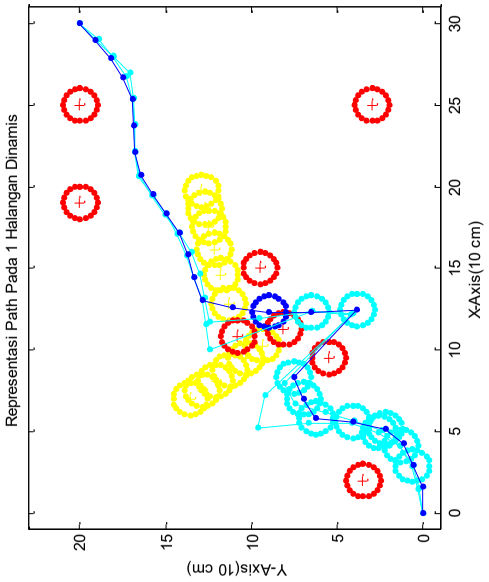
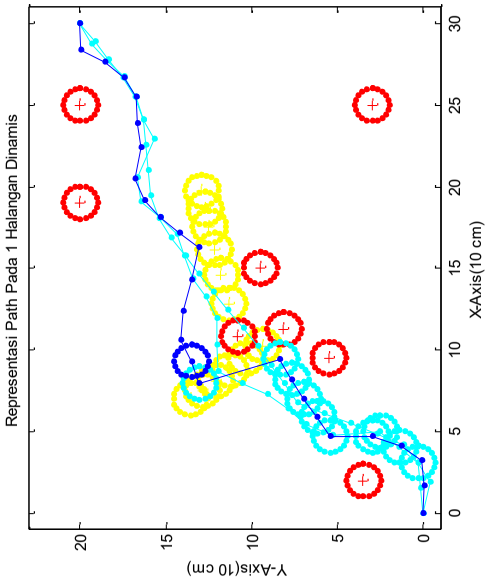
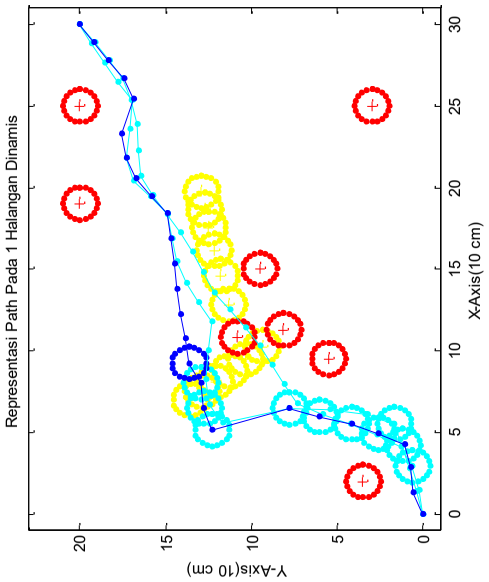
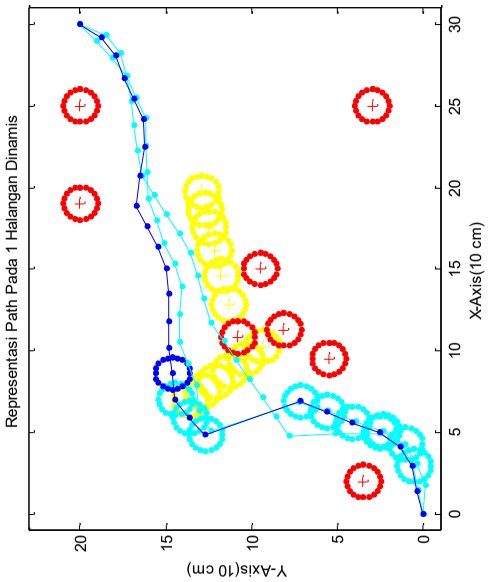
88.2016146645181	107.875244990420
180.304539379466	174.081632303441
9.03319062774070	30.7114440112472
43.1075161608842	42.3474745184849
	
69.2575898797400	71.4263227661011
118.716909959248	133.144012470026
1.72765958587291	1.62101859337790
43.7865483020175	43.1765016787180
	
19	**20**

	PSO Standard									
No	**Representasi Jalur**	J_d	J_r	J_s	f	**Representasi Jalur**	J_d	J_r	J_s	f
1		42.7157149906888	12.8938685745939	188.879310049466	93.3854455751760		45.7389793229892	4.67811861965499	227.655526544447	95.9482032515337
3		47.4740910393708	30.6477640133088	293.214898854560	136.764834823592		46.0992824643689	13.3268525680190	218.560420638209	103.138219160030

106.243867027837	111.406034394697
265.979495682259	260.986972273896
2.02096072519907	13.7645264726645
51.0270071661862	45.4441134672536
	
93.2539893352967	108.473494201387
160.161748238006	285.239843352366
15.7709555019756	3.54293849589879
45.4506841857198	47.8825870350147
	
5	7

98.6201910467324	95.5358233597559
221.926951006470	197.367082135497
4.12745413545879	12.5333316556827
50.1073467099796	43.5290752769739
	
94.0994128087984	94.0683865128585
197.495095783021	171.733925401313
11.5346888073172	16.9278716063589
43.0657048448769	42.7937298262369
	
9	**11**

93.3758609987546	84.7231345749390
180.998855112072	170.357705847059
12.2595345734841	6.50070166765874
44.9165554028560	44.1508917378685
	
100.374070923733	99.4700809101716
183.048930390620	182.770702321683
20.7307103227829	18.8606316859861
43.0335745228263	44.0553087598490
	
13	**15**

103.563449893326	128.039434053894
230.493951058957	312.098999081051
11.1524851472342	20.6037971794870
46.3121745343001	45.0158370581969
	
92.1761623172906	81.5273223176892
193.571750152280	174.812291049273
10.0423533464504	2.29452738875999
43.4194589403842	44.2703367190747
	
17	**19**

```
Program PSO Adaptif dengan Re-inisialisasi Partikel
function [titik_path] =PSO_Gauss(input,t,dim,start,finish)
 persistent inp it maxit minftot  N c1 upbnd lwbnd c2 brs w_max  i
j min_row y_max y_min idk Dsg y_path
 persistent Gbest Pbest f v w x fbest Ps d  Xmin Xmax Vmin Vmax
changerow succes_count max_row minf x_path kecep L e Ang P_length
P_danger P_sudut maxww
%*****************************************************************
*******
%                       Inisialisasi
%*****************************************************************
*******
   inp=input; maxit=300;
   N=30;
   if t==0
   y_max=mean(input(2,3:length(input)));
   y_min=-y_max;
   upbnd = y_max.*(ones(1,dim));  lwbnd = y_min.*(ones(1,dim));

x=rand(N,dim).*(repmat((upbnd),N,1))+rand(N,dim).*(repmat((lwbnd),
N,1));
   minftot=[];
   P_length=[];
   P_danger=[];
   P_sudut=[];
   kecep=[];
   maxww=[];
   V_max=y_max;
   v =rand(N,dim); %kecepatan awal
   Vmax=y_max.*ones(1,dim);  Vmin=y_min.*ones(1,dim);
   Xmax=(y_max+3).*ones(1,dim);  Xmin=(y_min-3).*ones(1,dim);
   w_max=1;    w_min=0;
   [brs,~]=size(x);
   f = zeros(N,1);
   end
   for i=1:brs

[f(i),~,~,~]=cost_function(inp,x(i,:),dim,start,finish);
       end
   it=1;
   Pbest=x;
   fbest=f;
   [~,idk]=min(f);
   Gbest=x(idk,:);
   w=w_max;
%*****************************************************************
*******
 %                      Running Algoritma Utama
%*****************************************************************
*******
while it<maxit
 succes_count=0;
c1=1.5*exp(-0.5*((it-0)/(0.5*maxit))^2)+0.5;
c2=1*exp(-0.5*((it-maxit)/(0.9*maxit))^2)+1;
% w=0.8;
 for j=1:brs
     for d=1:dim
```

```
        v(j,d)=w.*v(j,d)+(c1*rand())*(Pbest(j,d)-
x(j,d))+(c2*rand())*(Gbest(:,d)-x(j,d));
      end
        max_row=v(j,:) > Vmax;
        v(j,:)=v(j,:).*(1-max_row)+max_row.*Vmax;
        min_row=v(j,:) < Vmin;
        v(j,:)=v(j,:).*(1-min_row)+min_row.*Vmin;
        x(j,:)=x(j,:)+v(j,:);
        max_row=x(j,:) > Xmax;
        x(j,:)=x(j,:).*(1-max_row)+max_row.*Xmax;
        min_row=x(j,:) < Xmin;
        x(j,:)=x(j,:).*(1-max_row)+max_row.*Xmax;
        [f(j),~,~,~]=cost_function(inp,x(j,:),dim,start,finish);
end
%update Pbest
VV=mean(mean(abs(v)));
kecep=[kecep,VV];
changerow = f < fbest;
fbest=fbest.*(1-changerow)+f.*changerow;
Pbest(changerow,:)=x(changerow,:);
succes_count=size(find(changerow));
[minf,idk]=min(fbest);
Gbest=Pbest(idk,:);
minftot=[minftot;minf];
[~,L,e,Ang]=cost_function(inp,Gbest,dim,start,finish);
P_length=[P_length;L];
P_danger=[P_danger;e];
P_sudut=[P_sudut;Ang];
Ps=succes_count(1)/N;
w=(exp(-0.5*((it-0)/(0.5*maxit))^2))*Ps;
maxww=[maxww,w];
if mean(mean(abs(v)))<1e-3
   x=repmat(Gbest,N,1)+(rand(N,dim).*repmat(3,N,dim)-
rand(N,dim).*repmat(3,N,dim));
end
it=it+1;
end
Dsg=input(1,2);
assignin('base','objct_fnctn',[minftot,P_length,P_danger,P_sudut])
;
assignin('base','kcepatan',[kecep;maxww]);
x_path=(linspace(0,Dsg,dim+2));
y_path=[0 Gbest 0];
titik_path=[x_path;y_path];
```

RIWAYAT PENULIS

Novendra Setyawan dilahirkan di Lampung Timur dari pasangan Mansur dan Maryatun pada tanggal 19 November 1992. Pendidikan formalnya dimulai di SDN 1 Nabang Baru – Lampung Timur, SMPN 1 Sekampung – Lampung Timur, SMAN 3 Metro – Metro, dan kemudian melanjutkan studi di Teknik Elektro Universitas Muhammadiyah Malang. Setelah menyelesaikan studi tingkat strata, penulis melanjutkan studi magister di Teknik Elektro Institut Teknologi Sepuluh Nopember dengan bidang keahlian Teknik Sistem Pengaturan. Penulis telah menyelesaikan siding tesis pada tanggal 8 juni 2017 sebagai salah satu syarat untuk mendapatkan gelar **Magister Teknik (M.T.).** Penulis menyukai hal hal seperti musik, olahraga futsal, dan sains teknologi. Selama menempuh pendidikan tinggi penulis aktif sebagai anggota tim Worksop Robotika UMM dan aktif mengikuti Kontes Robot Indonesia. Saat ini penulis bertugas sebagai tenaga pengajar di Teknik Elektro Universitas Muhammadiyah Malang. Penulis sangat menerima kritik dan saran dari pembaca, guna pengembangan penelitian ini kedepannya. Harapan dari penulis, buku ini dapat bermanfaat bagi semua

Penulis

ndr.setya@gmail.com

Halaman ini sengaja dikosongkan