Islamic University Of Gaza
Deanery Of Higher Studies
Faculty Of Engineering
Electrical Engineering Department

Optimized Fuzzy Tracking Of A Moving Object With A Robotic Eye System

Presented by:
Ahmed S. A. Alostaz

Supervised by:
Dr. Basel Hamad

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master in Electrical Engineering – Control Systems

1435 هـ - 2014 م

نموذج رقم (1)

إقـــــرار

أنا الموقع أدناه مقدم الرسالة التي تحمل العنوان:

أقر بأن ما اشتملت عليه هذه الرسالة إنما هو نتاج جهدي الخاص، باستثناء ما تمت الإشارة إليه حيثما ورد، وإن هذه الرسالة ككل أو أي جزء منها لم يقدم من قبل لنيل درجة أو لقب علمي أو بحثي لدى أي مؤسسة تعليمية أو بحثية أخرى.

DECLARATION

The work provided in this thesis, unless otherwise referenced, is the researcher's own work, and has not been submitted elsewhere for any other degree or qualification

Student's name: اسم الطالب: أحمد سامي أحمد الأستاذ

Signature: التوقيع:

Date: التاريخ: ٢٠١٤/٧/٧

Islamic University Of Gaza
Deanery Of Higher Studies
Faculty Of Engineering
Electrical Engineering Department

Optimized Fuzzy Tracking Of A Moving Object With A Robotic Eye System

Presented by:
Ahmed S. A. Alostaz

Supervised by:
Dr. Basel Hamad

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master in Electrical Engineering – Control Systems

1435 هـ - 2014 م

بسم الله الرحمن الرحيم

الجامعة الإسلامية – غزة
The Islamic University - Gaza
مكتب نائب الرئيس للبحث العلمي والدراسات العليا هاتف داخلي 1150

الرقم Ref ج س غ/35/
2014/06/28
التاريخ Date

نتيجة الحكم على أطروحة ماجستير

بناءً على موافقة شئون البحث العلمي والدراسات العليا بالجامعة الإسلامية بغزة على تشكيل لجنة الحكم على أطروحة الباحث/ **أحمد سامي أحمد الأستاذ** لنيل درجة الماجستير في كلية *الهندسة* قسم الهندسة الكهربائية – أنظمة التحكم وموضوعها:

المتحكم المثالي الضبابي لتتبع الأجسام المتحركة عن طريق نظام العين الألية

Optimized fuzzy tracking of a moving object with a robotic eye system

وبعد المناقشة العلنية التي تمت اليوم السبت 30 شعبان 1435هـ، الموافق 2014/06/28م الساعة الواحدة ظهراً بمبنى القدس، اجتمعت لجنة الحكم على الأطروحة والمكونة من:

د. باسل محمود حمد	**مشرفاً ورئيساً**
د. حاتم علي العايدي	**مناقشاً داخلياً**
د. إياد محمد ابو هدروس	**مناقشاً خارجياً**

وبعد المداولة أوصت اللجنة بمنح الباحث درجة الماجستير في كلية *الهندسة*/ قسم الهندسة الكهربائية – أنظمة التحكم.

واللجنة إذ تمنحه هذه الدرجة فإنها توصيه بتقوى الله ولزوم طاعته وأن يسخر علمه في خدمة دينه ووطنه.

والله ولي التوفيق،،،

مساعد نائب الرئيس للبحث العلمي والدراسات العليا

أ.د. فؤاد علي العاجز

P.O. Box 108, Rimal, Gaza, Palestine fax: +970 (8) 286 0800 فاكس Tel: +970 (8) 286 0700 ص.ب. 108 الرمال، غزة، فلسطين هاتف
public@iugaza.edu.ps www.iugaza.edu.ps

ABSTRACT

Fuzzy proportional-integral-derivative (FPID) controller is used to control the robot eye, that has two degree of freedom(2DOF), to achieve a real time video object tracking. The controller uses vision-based object tracking as feedback and generates velocity commands for the robot eye in order to keep the object of interest (OBI) in video at the center of the image frame. Because of the nonlinearity of the proposed system, conventional PID controller is not optimal controller as shown in the results of comparison which is presented between FPID and PID for controlling the robot eye in real time .

The particle filter (PF) is used to track OBI in video by estimating its position in image frame. A new optimized bootstrap particle filter (OBPF) is introduced solving problem of particle filter on video object tracking for rigid motion. The comparison among OPF,PF-BFO and PF-PSO for video object tracking is presented in this work by using Matlab program. The results show that OPF method presents outstanding performance versus PF-PSO and PF-BFO.

A new adaptive optimized bootstrap particle filter (AOBPF) is introduced for tracking object that has variable speed. The comparison between OPF and AOPF for video object tracking in real time is tackled in this research. The results show that AOBPF is more robust.

The robot eye which is presented by the researcher is a webcam and two servo motors. Besides, video object tracking is implemented by using OpenCV. Furthermore, FPID controller is implemented by using visual C++. In addition, Arduino platform is used to communicate with visual C++ by using universal serial bus (USB) and to drive the two servomotors of robot eye.

ملخص

يستخدم المتحكم التناسبي التفاضلي التكاملي الغامض (FPID) كوحدة تحكم للسيطرة على روبوت العين لتحقيق تتبع لجسم معين من خلال لونه (كائن الفيديو) اثناء التقاط مباشر للفيديو. يستخدم المتحكم موقع كائن الفيديو على الصورة (التتبع المرئي) لتحديد السرعة التي سيسير عليها المحركات من أجل الحفاظ على الكائن في وسط الصورة. ومن خلال التجربة العملية تمت المقارنة بين المتحكم التناسبي التفاضلي التكاملي(PID) و المتحكم التناسبي التفاضلي التكاملي الغامض في قدرتهم على التحكم في الروبوت. و لقد أثبتت النتائج تفوق FPID , وذلك لأن النظام غير خطي.

يتم استخدام فلتر الجسيمات(PF) لتتبع كائن الفيديو عن طريق تقدير موقعه في الصورة. لحل مشكلات فلتر الجسيمات في التتبع المرئي تم تقديم نوع جديد من الفلتر (OBPF) . بمقارنة الفلتر الجديد بالفلاتر السابقة من نوع الجسيمات كانت نتائجه ممتازة مع الكائن ذو السرعة الثابتة وتفوقت عل نتائج الفلاتر السابقة , أما للكائن ذو السرعات المختلفة تم استخدام فلتر جديد آخر (AOBPF), حيث من خلال التجربة العملية توضح أن OBPF أداؤه غير عالي مع الكائن ذو السرعات المختلفة بعكس AOBPF

تم تركيب روبوت العين باستخدام محركان ذوا راجع ذاتي (servo motor) و استخدام كاميرة من نوع (webcam) لالتقاط الفيديو, و كانت لها المقدرة لالتقاط 25 صورة في الثانية. تتبع كائن الفيديو يتم باستخدام برنامج مكتوب بأداة OpenCV . وتم تنفيذ وحدة تحكم FPID باستخدام Visual C++. و تستخدم لوحة اردوينو لتواصل مع Visual C++ باستخدام الناقل التسلسلي العالمي (USB) لتحكم بالمحركان.

DEDICATION

I dedicate this work to my mother and my father, who encourage me and help to complete my study, who give me everything, love, time, attention and money.

I also dedicate my work to my beloved, my sisters, my wife and my daughter: Sama

ACKNOWLEDGEMENT

Thank to Allah, the lord of the worlds, for his mercy and limitless help and guidance. My peace and blessings be upon Mohammed the last of the messengers.

I thank my supervisor Dr. Basil Hamed for his time, suggestions, considerations, ideas and advices during this theses. Special thanks to Dr. .Hatem Elaydi and Dr. Iyad Abu Hadrous -thesis examiners- for their patience and guidance.

My thanks go to my best friends for their help. Special thanks to my friend Ali Alnemnem . I also thank my wife for her infinite support and patience. World will be not enough to thank my mother for everything in my life.

TABLE OF CONTENT

LIST OF TABLES:

LIST OF FIGURES:

ABBREVIATIONS

ABFO_PSO	Adaptive Bacteria Foraging Optimization Based Particle Swarm Optimization
AOBPF	Adaptive Optimized Bootstrap Particle Filter
BFO	Bacteria Foraging Optimization
BIA	Behavior Initiating Agent
BPF	Bootstrap Particle Filter
COA	Centroid Of Area
FLC	Fuzzy Logic Control
FLS	Fuzzy Logic Supervisory
FPID	Fuzzy Proportional-Integral-Derivative
GAO	Genetic Algorithm Optimization
IBVS	image based visual servo
IPF	Intelligent Particle Filter
MC	Monte Carlo
MIMO	Multi Input Multi Output
MM	Maximum Method
MOM	Mean Of Maximum
OBI	Object Of Interest
OBPF	Optimized Bootstrap Particle Filter
PDF	Probability Density Function
PF	Particle Filter
PF-BFO	Particle Filter Based Bacteria Foraging Optimization
PF_PSO	Particle Filter Based Particle Swarm Optimization
PID	Proportional-Integral-Derivative
PSO	Particle Swarm Optimization
PWM	Pulse Width Modulation
SIR	Sequential Importance Resampling
SIS	Sequential Importance Sampling
TSK	Takagi-Sugeno_Kang
USB	Universal Serial Bus

Chapter 1 : INTRODUCTION

1.1 Introduction

Visual object tracking has drawn significant attention in the field of computer vision due to its enormous worth in applications including visual surveillance [1], human-computer interaction, intelligent transportation [2], robot navigation [3], etc. The heart of solving visual object tracking problem consists of answering two questions: what does the object look like (feature level) and where is the object (tracking level).

The applications of object tracking for visual feedback are mostly used in robot control and computer vision. The real time tracking algorithm consists of two modules: the first module is designed to acquire the images, detect the interested moving objects, segment the object or track of such object from frame to frame and finally estimate the position of this object, then deliver this position as a desired value to the second module; the second module is designed as a position controller to maintain the object in the camera view. In this case, the information extracted from the vision sensor is used as a feedback signal in control.

Recently, particle filters have been extensively used in visual tracking field. They proved to be a robust method of tracking due to their ability of solving non-linear problems. Video object tracking through using PF suffers the same problems that PF do: degeneracy phenomenon and sample impoverishment. Degeneracy phenomenon where after a few iterations, all but one particle will have negligible weight is an undesirable effect in particle filter. Sample impoverishment occurs when the likelihood is very narrow or the likelihood distribution function lies at the tail of prior distribution.

In recent years, hybrid control system, between fuzzy and classical controllers, has been combined to design a controller such as Fuzzy plus PID and Fuzzy Logic Supervisory (FLS). Through this thesis, Fuzzy logic control (FLC) is considered as an important controller for on-line tuning of PID parameters.

In this thesis, fast and robust object tracking is proposed. OBPF is used to solve problem of particle filter on video object tracking for rigid motion. AOBPF is used for tracking object that has variable speed, while FPID is used to make movements of camera faster and more stable.

1.2 Statement of problem

This work proposes tracking moving objects based on color in real time. There are four problems that affect the system:

Problem related to accuracy of video object tracking using color.

1. Problem related to accuracy of video object tracking using color. The effects of this problem is reduced using OBPF filter on video object tracking.
2. Computational time required to estimate moving objects position. Using OBPF achieves fasts and robust tracking.
3. In real time, variable speed of OBI make particle filter loss tracking. AOBPF is proposed to solve this problem.
4. Computational time required to maintain the object in the center. The time required to maintain the object in the center of camera view is minimized by using FPID controller.

1.3 Research objectives

The main objectives of this research are as follow:

1. Strengthen the video object tracking based on particle filter by making corporation between particle filter and ABFO_PSO.
2. Make tracking faster buy using fuzzy PID controller.
3. Reduce computational time of video object tracking system

1.4 Methodology

Figure 1.1: Tracking System Components

There has been an increase of interest in how to reach the optimal way to fast and accurate visual tracking system, In this thesis I will try to reach this target.

There are nine steps to achieve video object tracking system shown in Figure(1.1), that aims to maintain OBI at the center of camera frame :

1. Writing matlab program to achieve video object tracking based by using OBPF.
2. Evaluating the video object tracking based on OBPF result by comparing it with previous studies.
3. Using open CV tool to write C++ program to achieve video object tracking based on OBPF for rigid motion and based on AOBPF for tracking object that has variable speed.
4. Making Interface between Arduino (motor driver) and visual c ++ using USB.
5. Developing a mechanical design of the robot eye .
6. Designing PID controller to centralize OBI.
7. Designing F PID controller to centralize OBI.
8. Evaluating AOBPF by comparing it with OPF in real time tracking.
9. Evaluating the F PID result by comparing it with the PID result.

1.5 Literature review

1. In 2013 (Ming Li) proposed intelligent particle filter (IPF) by using Genetic Algorithm Optimization (GAO) and Particle Swarm Optimization Algorithm (PSO) [4]. Both PSO and (GAO) were used in resample step of PF. PSO is used to solve particle degradation problem and GAO was used to solve impoverishment problem. The simulation results show that the performance of IPF was better than standard PF do. In this thesis, PSO was used in resample step to solve both problem of PF, and a new technique in predict step of PF is proposed to solve PF problems.

2. In 2009 (Long Ye, et.al) introduced genetic Monte Carlo sampling method, and then used it in the resampling step of particle filter with the basic idea of solving particle degeneration problem by means of evolution thought[5]. The principle of this framework is to balance the two contradictory sides: particle effectiveness and particle diversity. The simulation result demonstrated the advantages of this method comparing with the other particle filtering frameworks. In this thesis, a novel optimized particle filter is proposed to solve PF problems.

3. In 2008 (Gongyuan Zhang, et.al) proposed particle filter based on PSO [6]. PSO was used in resample step of PF to solve its problem. Simulation results showed PSO operation on particles set can overcome sample impoverishment problem. In this thesis, PSO is used in resample step to solve both problems of PF. In addition, a new technique in predict step of PF is proposed to solve PF problems.

4. In 2013(Alhanjouri & AlOstaz) made comparison between Bacteria foraging optimization (BFO) and PSO for video object tracking using particle filter [7]. Unprecedented method, particle filter based Bacteria foraging optimization (PF-BFO), in video object tracking was provided. The results showed that the new particle filter performed accurately and achieved stable tracking. It solved both degeneracy phenomenon and impoverishment problem. This method has the advantages of both adding mean shift to PF and improving resample method of PF. However, applying it in real time video tracking needs processor that has high velocity. In this study, OBPF is used to solve the problem of velocity.

5. In 2013 (Zhang Xiaowei, et.al) proposed an object tracking algorithm evolutionary Particle filter based on self-adaptive multi-features fusion [8]. The algorithm constructed the importance function that approached the posterior probability distribution by fusing the color cue and the texture cue adaptively. The results of the experiment showed that the proposed object tracking algorithm has more powerful anti-interference ability and higher accuracy for tracking. In this these, a new AOBPF is proposed for real time tracking.

6. In 2010 (Hao ,et.al) proposed video object tracking based on swarm optimized particle Filter. PF was combined with PSO [9]. PSO improved the way of resample particle to deal with the problem of sample impoverishment. Using particle filter based particle swarm optimization (PF_PSO) in video object tracking showed the strength of it. In this thesis, PF-PSO is improved by enhancing predict step.

7. In 2009 (Zhou Hao, et.al) proposed Video Object Tracing Based on Particle Filter with Ant Colony Optimization [10]. ACO algorithm optimized the sample set before re-sampling step. Target state estimation was computed according to the optimized samples. Experiment results demonstrated that the proposed algorithm effectively improved the efficiency of video object tracking system.

8. In 2007 (Bai & Liu) proposed improved object tracking with particle filter and mean shift [11]. The mean-shift algorithm was added to the particle filter in object tracking. The mean shift improves the weights of particles before resample. That overcame the degeneracy problem of conventional particle filters and required less computational cost. In this thesis, optimization technique is applied to predict step of PF to avoid computational time of mean shift.

9. In 2013 (Qadir & Semke) proposed vision based neuro-fuzzy controller for a two Axes Gimbal System with Small UAV [12]. The controller generated position and velocity commands for the gimbal motion controller in order to move the gimbal. The gimbal moved accordingly to keep the object of interest in the video at the center of the image frame. A basis function based neuro-fuzzy system and a learning algorithm was developed for the controller to address the dynamic and non-linear characteristics of the gimbal movement. In this work, a different type of intelligent controller (FPID) is used.

10. In 2011 (El-Bardini, et.al.) proposed real time object tracking using image based visual servo (IBVS) technique [13]. The real time object tracking based on IBVS was implemented. The algorithm merge the feature based and the modified frame difference technique to detect, segment and track Static and Moving objects in real time. The position controller was designed and implemented to generate a PWM signal that used to control the position and direction of the DC-Motor, to maintain the tracked object in the center of camera view, via hardware circuits. In this study, FPID controller is used.

11. In 2010 (Chang & Campoy) proposed a neural behavior initiating agent (BIA) to integrate relevant compressed image [14]. Information coming from others cooperating and specialized neural agents. Using this arrangement, the problem of tracking and recognizing a moving icon for robotic eye was solved by partitioning it into three simpler and separated tasks.

1.6 Contributions

As a result of this work following are achieved:

- Proposing a new comparison between PF and PSO and a new comparison between PF and BFO.
- Proposing a new OBPF for video object tracking.
- Proposing a new AOBPF for tracking object with variable velocity.
- Implementing the system in real time.
- Designing FPID controller for robot eye which makes tracking faster and more stable.

1.7 Thesis structure

This thesis is organized as follows: Chapter 2 reviews modern optimization theories. Chapter 3 covers the topic of particle filter algorithm. Chapter 4 reviews using particle filter in object tracking. Chapter 5 describes the control design of robot eye and discusses the experiment results. In the chapter 6, the conclusion and future work are discussed.

Chapter 2 : OPTMIZATION THEROY

2.1 Particle swarm optimization

2.1.1 Introduction

Particle swarm optimization is a swarm intelligence technique which was proposed by Kennedy and Eberhart in1995 [15]. Particle swarm optimization is based on the behavior of a colony or swarm of insects, such as ants, termites, bees, and wasps; a flock of birds; or a school of fish as shown in Figure (2.1). The particle swarm optimization algorithm mimics the behavior of these social organisms. The word particle denotes, for example, a bee in a colony or a bird in a flock. Each individual or particle in a swarm behaves in a distributed way using its own intelligence and the collective or group intelligence of the swarm. As such, if one particle discovers a good path to food, the rest of the swarm will also be able to follow the good path instantly even if their location is far away in the swarm. Optimization methods based on swarm intelligence are called behaviorally inspired algorithms as opposed to the genetic algorithms, which are called evolution-based procedures.

Figure 2.1 : Particle Swarm Optimization In Nature

2.1.2 Particle Swarm Algorithm

Reynolds [16] proposed a behavioral model in which each agent follows three rules as shown in Figure (2.2):

- Separation: Each agent tries to move away from its neighbors if they are too close.
- Alignment: Each agent steers towards the average heading of its neighbors.
- Cohesion: Each agent tries to go towards the average position of its neighbors.

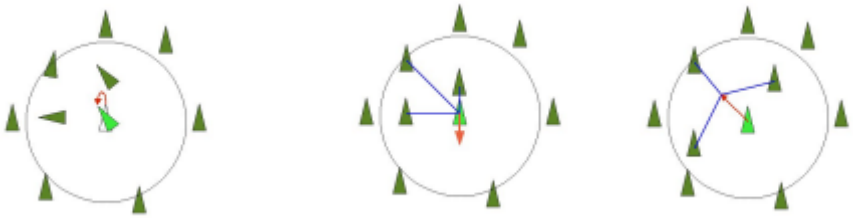


Figure 2.2: Reynolds Behavioral Model Of PSO

Kennedy and Eberhart [15] included a roost in a simplified Reynolds-like simulation in which each agent follows three rules as shown in Figure(2.3):

- Each agent was attracted towards the location of the roost.
- Each agent remembered where it was closer to the roost.
- Each agent shared information with its neighbors (originally, all other agents) about its closest location to the roost.

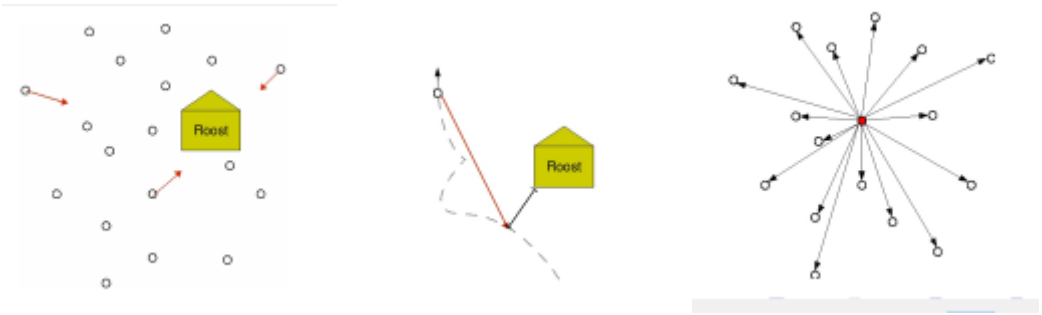


Figure 2.3: Kennedy And Eberhart Behavioral Model Of PSO

The PSO is developed based on the following model as in [17]:

1. When one bird locates a target or food (maximum of the objective function), it instantaneously transmits the information to all other birds.
2. All other birds gravitate to the target or food (maximum of the objective function), but not directly.
3. There is a component of each birds own independent thinking as well as its past memory.

Thus the model simulates a random search in the design space for the maximum value of the objective function. As such, gradually over much iteration, the birds go to the target (or maximum of the objective function).

The PSO procedure can be implemented through the following steps:

4. Assume the size of the swarm (number of particles) is N.
5. Generate the initial population of X randomly as X1, X2... XN (positions of particle).
6. Find the velocities of particles. All particles will be moving to the optimal point with a velocity. Initially, all particle velocities are assumed to be zero.
7. In the ith iteration, find the following two important parameters used by a typical particle j :

a) The historical best value of $X_j(i)$ (coordinates of jth particle in the current iteration i), P_{best}, with the highest value of the objective function, $f[X_j(i)]$ encountered by particle j in all the previous iterations.

b) The historical best value of $X_j(i)$ (coordinates of all particles up to that iteration),$Gbest$, with the highest value of the objective function $f[X_j(i)]$, encountered in all the previous iterations by any of the N particles.

c) Find the velocity of particle j in the ith iteration as follows:

$$V_j(i) = V_j(i-1) + c_1 * [(P_{best,i} - X_j(i-1)] + c_2 * [(G_{best} - X_j(i-1)] \quad (2.1)$$

d) Find the position or coordinate of the jth particle in ith iteration as

$$X_j(i) = c_3 * X_j(i-1)] + V_j(i) \quad (2.2)$$

where c_1, c_2 and c_3 are constant

8. Check the convergence of the current solution. If the positions of all particles converge to the same set of values, the method is assumed to have converged. If the convergence criterion is not satisfied, step 4 is repeated by updating the iteration number as $i = i + 1$, and by computing the new values of $P_{best,i}$and G_{best} .The iterative process is continued until all particles converge to the same optimum solution.

2.2 Bacteria Foraging Optimization

2.2.1 Introduction

Bacteria Foraging Optimization, proposed by Passino in 2002, is a new comer to the family of nature-inspired optimization algorithms. Application of group foraging strategy of a swarm of E.colibacteria in multi-ptimal function optimization is the key idea of the algorithm. Bacteria search for nutrients in a manner to maximize energy obtained per unit time. Individual bacterium also communicates with others by sending signals. A bacterium takes foraging decisions after considering two previous factors. The process, in which a bacterium moves by taking small steps while searching for nutrients, is called chemotaxis and key idea of BFO is mimicking chemotactic movement of virtual bacteria in the problem search space.

2.2 2. The Bacteria Foraging Optimization Algorithm

Like other swarm intelligence algorithms, BFO is based on social and cooperative behaviors found in nature. In fact, the way Bacteria look for regions of high levels of nutrients can be seen as an optimization process. The bacteria foraging strategy can be explained by four processes as in [18]:

A. Chemotaxis :

This process simulates the movement of an E.colicell through swimming and tumbling via flagella as shown in Figure(2.4). Biologically an E.coli bacterium can move in two different ways. It can swim for a period of time in the same direction or it may tumble, and alternate between these two modes of operation for the entire lifetime. Suppose $\boldsymbol{\theta^i(j,k,l)}$ represents $\boldsymbol{ith}$ bacterium at $\boldsymbol{jth}$ chemotactic, $\boldsymbol{kth}$ reproductive and $\boldsymbol{lth}$ elimination-dispersal step. $\boldsymbol{C(i)}$ is the size of the step taken in the random direction specified by the tumble (run length unit). Then in computational chemotaxis, the movement of the bacterium may be represented by

$$\theta^i(j+1,k,l) = \theta^i(j,k,l) + c(i) * \frac{\Delta(i)}{\sqrt{\Delta(i)^T * \Delta(i)}} \quad (2.3)$$

Where Δ indicates a vector in the random direction whose elements lie in [-1, 1].

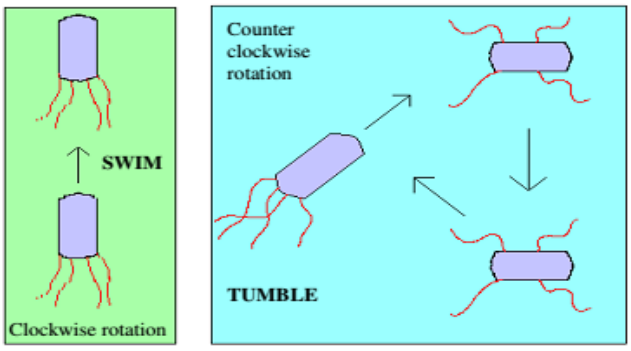


Figure 2.4: Swim and tumble of bacterium

B. Swarming [19]:

For the bacteria to reach at the richest food location, it is desired that the optimum bacterium till a point of time in the search period should try to attract other bacteria so that together they converge at the desired location more rapidly. To achieve this, a penalty function based upon the relative distances of each bacterium from the fittest bacterium till that search duration, is added to the original cost function. Finally, when all the bacteria have merged into the solution point, this penalty function becomes zero. The cell-to-cell signaling in E. coli swarm may be represented by the following function.

$$J_{cc}\left(\theta, P(j,k,l)\right) = \sum_{i=1}^{s} J_{cc}\left(\theta, \theta^{i}(j,k,l)\right) = \sum_{i=1}^{s}\left[-d_{attractant} * e^{\left(-w_{attractant}\sum_{m=1}^{P}\left(\theta_m - \theta_m^i\right)^2\right)}\right] + \sum_{i=1}^{s}\left[-h_{repellant} * e^{\left(-w_{repellant}\sum_{m=1}^{P}\left(\theta_m - \theta_m^i\right)^2\right)}\right] \quad (2.4)$$

Where $\theta = [\theta_1, \ldots, \theta_p]$ is a point on the optimization domain and θ_m^i is the m^{th}component of the i^{th} bacterium position θ^i. $d_{attractant}$, $h_{repellant}$, $w_{attractant}$ and $w_{repellant}$ are constant.

C. Reproduction:

The least healthy bacteria eventually die while each of the healthier bacteria asexually split into two bacteria, which are then placed in the same location. This keeps the swarm size constant.

D. Elimination and dispersal:

Gradual or sudden changes in the local environment where a bacterium population lives may occur due to various reasons e.g. a significant local rise of temperature may kill a group of bacteria that are currently in a region with a high concentration of nutrient gradients. Events can take place in such a fashion that all the bacteria in a region are killed or a group is dispersed into a new location.

In what follows, the outline of BFO algorithm step by step is briefly[18]:

1. Initialize parameters

$n, S, Nc, Ns, Nre, Ned, Ped, C(i), \theta^i, (i = 1,2,\ldots,S)$

Where:

n: $dimension\ of\ the\ search\ space$

S: $the\ number\ of\ bacterium,$

Nc : $chemotactic\ steps,$

Ns : $swim\ steps,$

Nre : $reproductive\ steps,$

Ned: $elimination\ and\ dispersal\ steps,$

Ped: $probability\ of\ elimination,$

$C(i)$: $the\ chemotactic\ step\ size\ during\ each\ run\ or\ tumble$)

2. Elimination-dispersal loop:$l = l + 1$
3. Reproduction loop: $k = k + 1$
4. Chemotaxis loop: $j = j + 1$

For $i = 1$ to S, take a chemotactic step for bacteria as follows.

a) Compute fitness function, $J(i, j, k, l)$ and let

$$J(i, j, k, l) = J(i, j, k, l) + J_{cc}\left(\theta^i(j, k, l), P(j, k, l)\right) \quad (2.5)$$

b) Let $J_{last} = J(i, j, k, l)$ to save this value since we may find better value via a run.

c) Tumble: generate a random vector $\Delta(i) \in R^n$ with each element$\Delta_m(i), m = 1,2,\ldots S$ a random number on$[-1, 1]$.

d) Compute $J(i, j + 1, k, l)$ with$\theta^i(j + 1, k, l)$.

e) Swim:

i. Let m=0 counter for swim length.

ii. While m<Ns if not climbed down too long

- Let m= m +1
- If $J(i, j + 1, k, l) < J_{last}$, let $J_{last} = J(i, j + 1, k, l) <$ Then, another step of size $C(i)$ in this same direction will be taken as (2.3) and use the new generated $\theta^i(j + 1, k, l)$ to compute the new $\theta^i(j + 1, k, l)$
- Else let m=Ns.

5. If $j < Nc$,go to Step 4. In this case, continue chemotaxis since the life of the bacteria is not over.
6. Reproduction:
   - For the given k and l, and for each $i = 1,2,\ldots S$, let

     $$J_{health}^{i} = \sum_{j=1}^{N_c+1} J(i,j,k,l) \quad (2.6)$$

   - The Sr bacteria with the highest $Jhealth$ values die, and the other Sr bacteria with the best values split, and the copies that are made are placed at the same location as their parent.
7. If $k < Nre$ go to Step 3. In this case, the number of specified reproduction steps is not reached; start the next generation in the chemotactic loop.
8. Elimination-dispersal: for $i = 1 \ldots\ldots S,$ with probabilityPed, eliminate and disperse each bacteria, which results in keeping the number of bacteria in the population constant . To do this, if a bacterium is eliminated, simply disperse one to a random location on the optimization domain. If l<Ned, then go to Step 2,otherwise,end.

2.3 Adaptive BFO based on PSO(ABF_PSO)

2.3.1 Introduction

Recently BFO has attracted much attention from the computational intelligence community and has been applied successfully to a number of engineering problems, such as optimal power flow scheduling harmonic estimation, PID controller design load forecasting, stock market prediction and face recognition etc. However, BFO has the problem of low convergence speed and it usually results in sustained oscillation, especially on flat fitness landscapes, when a bacterium cell is close to the optima. To improve BFO search performance, some variants of BFO was proposed which includes parameters modification [20],[21] and hybrid algorithms[22],[23]. Adaptive BFO based on PSO[24] is used to propose a new particle filter.

2.3.2 ABF_PSO Algorithm

ABF_PSO algorithm combines both BFO and PSO [24]. The aim is to make PSO able to exchange social information and BFO able to find new solution by tumble, swim, reproduction, elimination and dispersal in the same algorithm.

In ABFO_PSO algorithm, PSO is used to update the length of Chemotaxis step. The velocity of bacteria still has random direction because of tumble behavior of BFO, but the magnitude of the velocity is calculated by using social behavior of PSO. C(i) is calculated as the magnitude of the velocity of particle in PSO:

$$\mathbf{C(i)} = \mathbf{w} * \mathbf{norm}\big(\Delta(\mathbf{i})\big) + \mathbf{c_1} * \mathbf{norm}(\boldsymbol{\theta}_{\mathbf{best}} - \boldsymbol{\theta}_{\mathbf{i}}) + \mathbf{c_2} * \mathbf{norm}(\boldsymbol{\theta}_{\mathbf{gbest}} - \boldsymbol{\theta}_{\mathbf{i}}) \quad (2.7)$$

θ_{best} is updated every swim step. θ_{gbest} is updated every reproduction step.

Because this combination updates the C(i) by using social information of PSO and swarming step represents social behavior in BFO, the swarming step is avoided in ABFO_PSO. Using social information of PSO and avoiding swarming step make BFO faster and more robust. The detail of algorithm is shown below:

1) Initialize input parameter:

S_p : Total number of bacteria in the population.
N_c : The number of chemotaxtis steps.
N_{re} : The number of reproduction steps.
N_s : The length of swim
N_{ed} : The number of elimination-dispersal events.
P_{ed} : Elimination-dispersal probability.
$C(i)$: The size of the step taken in the random direction.

2) Create random initial swarm bacteria $\theta^i(j,k,l)$ and initialize their fitness J_{health}, J_{best} , J_{gbest}

3) For i =1, 2 … N_{ed}
For j =1, 2 … N_{re}
For k =1, 2 … N_c
For l =1, 2 …S_p

Compute fitness function using eq. (2.5)

End for

Update J_{best}

For l =1, 2 …S_p

Perform chemotaxis step:

1) Calculate magnitude of step ($C(i)$) using eq.(2.7)
2) Perform tumble (indicate direction of step)
3) Swim until improve fitness or arrive the length of swim .
4) Fitness is calculated using only object function.

The detail of tumble and swim is proposed in BFO algorithm.

End for

End for

- Calculate J_{health} using eq.(2.6)
- The S bacteria with the highest J_{health} values die and the remaining S bacteria with the best values split.
- Update J_{gbest}

End for

Perform eliminating _dispersal step for all bacteria $\theta^i(j,k,l)$ with probability $0 < Ped < 1$.

End for

Chapter 3 : PARTICLE FILTER

3.1 Introduction:

Particle filter algorithm is a filtering method which uses Monte Carlo idea within the framework of Bayesian estimation theory. It approximates the probability distribution by using particles and discrete random measure which consist of their weights; it updates new discrete random measure recursively according to the algorithm . When the sample is large enough, the discrete random measure approximates the true posteriori probability density function of the state variable. The particle filter algorithm is applicable to any non-linear non-Gaussian system. But the standard particle filter does not consider the current measured value, which will lead to particles with non-zero weights become less after some iterations, This results in particle degradation; re-sampling technique was used to inhibit degradation, but this will reduce the particle diversity, and results in particle impoverishment. To overcome both problems, optimization algorithms are combined with particle filter. In this thesis we propose a new optimized particle filter.

3.2 The Monte Carlo Principle :

The Monte Carlo (MC) principle is a stochastic sampling approach that analytically approximates intractable numerical integration problems. This principle uses a set of independent and identically distributed random samples to approximate the true integral representing the required probability density function PDF[25].

The MC principle states that if independent, random, and equally weighted samples are drawn out from the required posterior distribution, then the required posterior PDF $P(x_{0:T}|, y_{1:T})$ can be approximated by the average or a collection of these samples as given in (3.1):

$$P(x_{0:T}|, y_{1:T}) = \frac{1}{N}\sum_{i=1}^{N} \delta(x_{0:T} - x_{0:T}^{i}) \tag{3.1}$$

Where δ represents the Dirac delta function, $\{x_k^i \mid i = 1, \ldots, N\}$. a drawn set of large N samples or particles with equal weights$\{w^i = 1/N \mid i = 1, \ldots, N\}$. Now the expectations $I(f)$ of any function are approximated in summation form as given in (3.2) and (3.3):

$$E[\, g(x)|y_{1:T}\,] = E[g(x_{0:T})P(x_{0:T}|, y_{1:T})]dx_{0:T} \tag{3.2}$$

$$E[\, g(x)|y_{1:T}\,] = \frac{1}{N}\sum_{i=1}^{N} f(x_{0:T}^{i}) \tag{3.3}$$

For the MC method, independent particles are required from the posterior density function. However, it is not always possible to draw out independent and uniformly distributed samples from the required density function $P(x_{0:k}|, y_{1:k})$ as it may be multivariate or nonstandard. In these cases, the importance sampling method is used.

3.3 Importance Sampling Method

The importance sampling method states that if it is difficult to obtain samples directly from a distribution, then samples can be generated from the importance density function $\pi(x \mid y_{1:T})$, from which we can easily draw samples. That is similar to the desired posterior distribution, but the weights of these generated samples need to be adjusted to represent the target function as closely as possible. More detail is found in [25],[26],[27].

3.4 Sequential Importance Sampling

Sequential importance sampling (SIS) is a sequential version of importance sampling. The SIS algorithm can be used for generating importance sampling approximations to filtering distributions of generic state space models of the form[27]:

$$x_k \sim p(x_k|x_{k-1}\ ) \tag{3.4}$$

$$y_k \sim p(y_k|x_{k-1}\ ) \tag{3.5}$$

Where $X_K \in Rn$ is the state at time step k and $Y_K \in Rm$ is the measurement. The state and measurements may contain both discrete and continuous components. The detail of SIS are the following:

- Initialization

  Draw the states

  $$X_0^i \sim P(\ X_0\ ) \qquad i = 1\ \ldots . N_s \tag{3.6}$$

- For each k = 1, . . . , T do the following:

  (1) Draw samples x_k^i from the importance distributions

  $$x_k^i \sim \pi\left(x_k^i \middle| x_{k-1}^i\ , y_k\ \right)\ \ i = 1\ \ldots . N_s \tag{3.7}$$

  (2) Update the particles a weight W_k^i according to

  $$W_k^i \propto W_{k-1}^i \frac{\mathrm{P}(\ Z_k\ |X_k^i\ )\ \mathrm{P}(\ X_k^i\ |X_{k-1}^i\ )}{\pi(X_k^i|X_{k-1}^i\ ,Y_k\ )} \tag{3.8}$$

  (3) Normalize the weights

  $$W_k^i = W_k^i / \sum_{j=0}^{Ns} W_k^j \tag{3.9}$$

  (4) Estimate sate:

  $$\hat{x}_k = E[\ x_k|y_k\ ] = \sum_{i=1}^{N}\ W_k^j * x_k^i \qquad i=1\ \ldots\ N_s \tag{3.10}$$

A common problem with the SIS particle filter is the degeneracy phenomenon, where after a few states all but one particle will have negligible weight. This degeneracy implies that a large computational effort is devoted to updating particles whose contribution to the approximation of the posterior density function is

almost zero. This problem can be overcome by increasing the number of particles. In addition, the use of the resampling technique is recommended to avoid the degeneracy of the particles.

3.5 Sequential Importance Resampling (Particle Filter)

The idea of the resampling procedure is to remove particles with very small weights and to duplicate particles with large weights. Although the theoretical distribution represented by the weighted set of samples does not change, resampling introduces additional variance to estimates. Resampling procedure can be described as follows [27]:

1. Interpret each weight w_k^i as the probability of obtaining the sample index i in the set $\{x_k^i \,|i = 1,\ldots,N\}$.
2. Draw N samples from that discrete distribution and replace the old sample set with this new one.
3. Set all weights to the constant value $w_k^i = 1/N$.

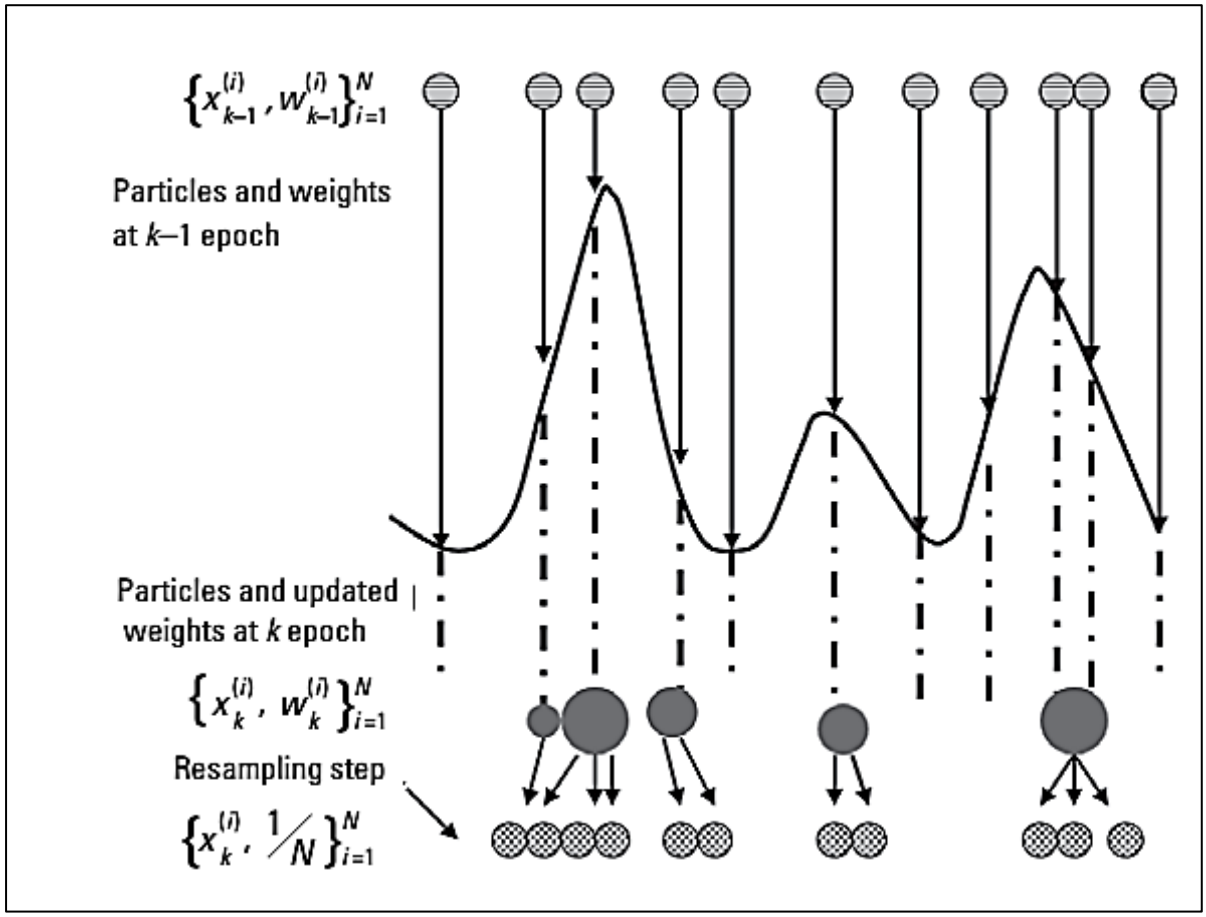


Figure 3.1: Sequential Importance Resampling Algorithm[26]

By using Sequential Importance Resampling (SIR) algorithm we can definite PF as a Bayes estimation algorithm based on Monte Carlo method. It performs the

posterior probability density function via a number of weighted particles and then resample. Here, resample means that PF eliminates particles that have small weights and concentrates on particles with large weights as shown in Figure(3.1). The details of algorithm is shown below[27]:

- Initialization : draw the states as in eq.(3.6)
- For each k = 1, . . . , T do the following:
  (1) Draw samples x_k^i from the importance distributions as in eq. (3.7)
  (2) Update the particles a weight W_k^i as in eq. (3.8)
  (3) Normalize the weights as in eq. (3.9)
  (4) Resample: Multiply/Suppress samples $x_{0:k}^{(i)}$ with high/low W_k^i obtain N_s random samples $x_{0:k}^{(i)}$approximately distributed according to $P(x_{1:k} \mid y_{1:k})$
  (5) Estimate sate: x_s^i particle after sample

$$\hat{x}_k = E\left[x_k \middle| y_k \right] = \sum_{i=1}^{N} W_k^j * x_{sk}^i * \delta(x_{sk}^i - x_s^i) \qquad i = 1, \ldots \ldots \ldots N_s \quad (3.11)$$

  (6) Set all weights to the constant value

$$W_k^i = 1/N_s \quad (3.12)$$

There are many different types of resampling methods [28], all of which are unbiased but differ in the variance aspect of the new un-weighted density function being generated such as:

1. Multinomial Resampling
2. Systematic Resampling
3. Residual Resampling
4. Stratified Resampling

3.6 Types of particle filter

There are a number of pragmatic PF techniques that have been introduced in the literature. Here are some of the more robust and popular techniques that have been applied to a wide variety of problems:

1. Bootstrap Particle Filter (BPF).
2. Auxiliary Particle Filter
3. Regularized Particle Filter
4. Linearized Particle Filter
5. Rao-Blackwellized Particle Filter.
6. Likelihood Particle Filter.
7. Gaussian Particle Filter.
8. Extended Particle Filter.

In this thesis we used Bootstrap Particle Filter with optimization algorithms and with Gaussian weight function.

3.6.1 Bootstrap Particle Filter

The basic of bootstrap algorithm developed by Gordon, Salmond and Smith is one of the first practical implementations of the processor to the tracking problem. It is the most heavily applied of all PF techniques due to its simplicity. The bootstrap filter is a variation of SIR where the dynamic model $\boldsymbol{p(xk|\ xk-1)}$ is used as the importance distribution. This makes the implementation of the algorithm very easy.

One of the primary features as well as shortcomings of this technique is that it does not only use the latest available measurement in the importance proposal, but also the previous particles,$\boldsymbol{x(k-1)}$. The corresponding weight becomes quite simple and only depends on the Likelihood. In order to achieve convergence, it is necessary to resample at every time step. In order to construct the bootstrap PF, follow steps as shown in Figure (3.2)[26]:

1. Initialization: Generate the initial state ,$x_i(0)$
   Generate the process noise, $w_i(t)$
2. Prediction : give positions for particles according to the model and previous position.

$$x_i(k) = A(x_i(k-1), u(k-1), W_i(k-1)) \tag{3.13}$$

3. Weights of particle (update state): generate the likelihood, $c(y(t)|xi(k))$ using the current particle and measurement
4. Normalize weight: sum of weight =1.
5. Resample particles: resample the set of particles retaining and replicating those of highest weight.
6. Estimation: estimate the best value using eq. (3.11).
7. Generate the new set,

$$\{x_i(k), W_i(t)\} with \hat{} W_i(t) = 1/N_p\}$$

3.7 Problem of SIR (PF):

However, the re-sampling step reduce degeneracy problem, it introduces other practical problems, such as the problem of sample impoverishment. It occurs when the likelihood is very narrow or the likelihood distribution function lies at the tail of prior distribution. This sample impoverishment can be solved through enlarging the sample set to cover the whole state space and to ensure estimation successfully. Therefore, the computation will be negatively affected.

To solve both problems optimization algorithms as PSO, genetic algorithm and BFO are combined with particle filter. In this thesis, a new optimized particle filter is proposed.

3.8 Particle Filter Based Optimization Theory

3.8.1 Particle Filter Based Particle Swarm Optimization

We can see there are many similarities between PSO and particle filter. PSO can be consider as particle filter without predict step and with time invariant weight function (objective function). To show difference, we represent PSO in particle filter form as shown in Figure (3.2), the detail of comparison is below:

1. Both PF and PSO have initialization step, weight step and resample step.
2. In PSO, weight function ($F(x)$ objective function) is time invariant. In PF, weight function is time varying ($F(x, t)$) .
3. PSO doesn't have prediction step.
4. In PSO , resample step can be implemented by eq.(2.1) and eq.(2.2). In PF, resampling step depends on algorithms proposed in section 3.4
5. PSO doesn't need normalize step.

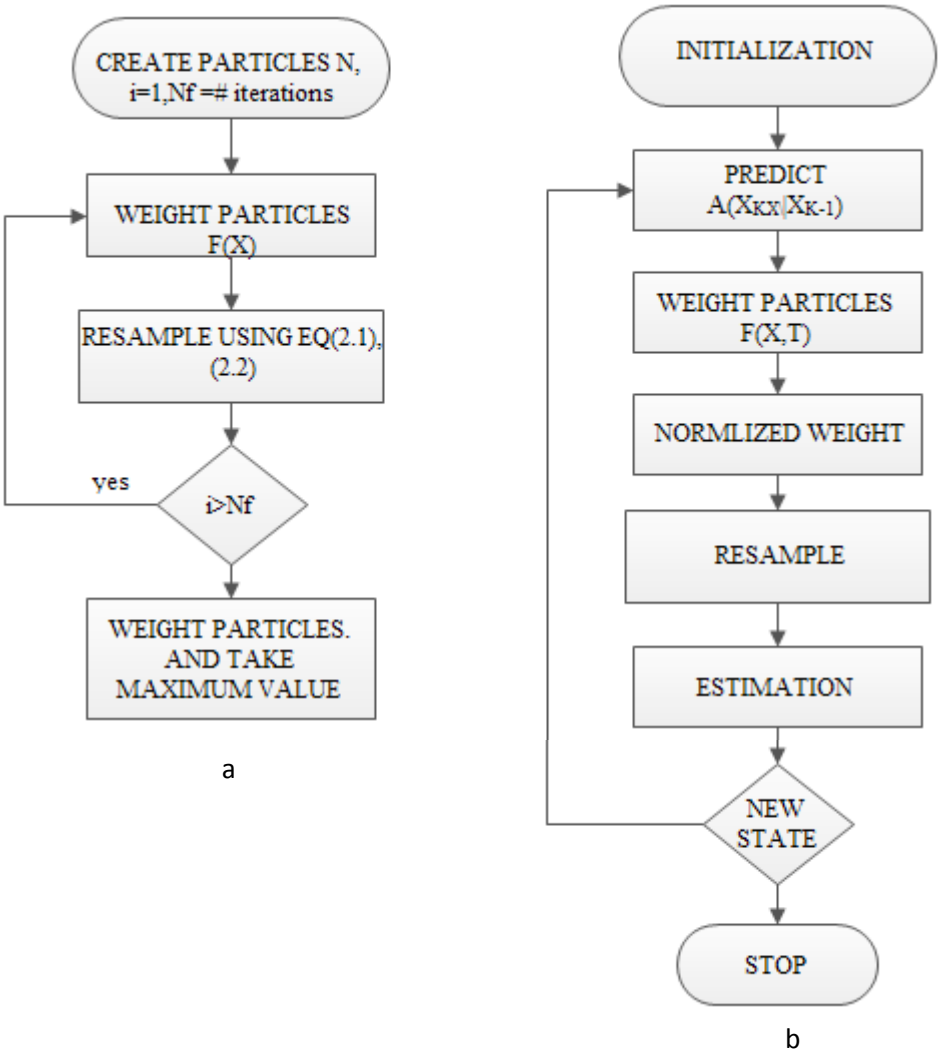


Figure 3.2: (A) PSO Flowchart , (B) PF Flowchart

From previous comparison, it is to combine PF with PSO. To make this combination, do the following changes in PF :

1. Use PSO movements as in eq.(2.1)& eq.(2.2) in resample step
2. Avoid normalize step.
3. Use the following equation for estimation:

$$\widehat{X} = \frac{\sum_{i=1}^{N_S} X_k^i}{N_S} \quad (3.14)$$

3.8.2 Particle Filter Based Bacteria foraging Optimization

Like PSO, to show difference, we represent BFO in particle filter form as shown in Figure (3.3). The detail of comparison is below:

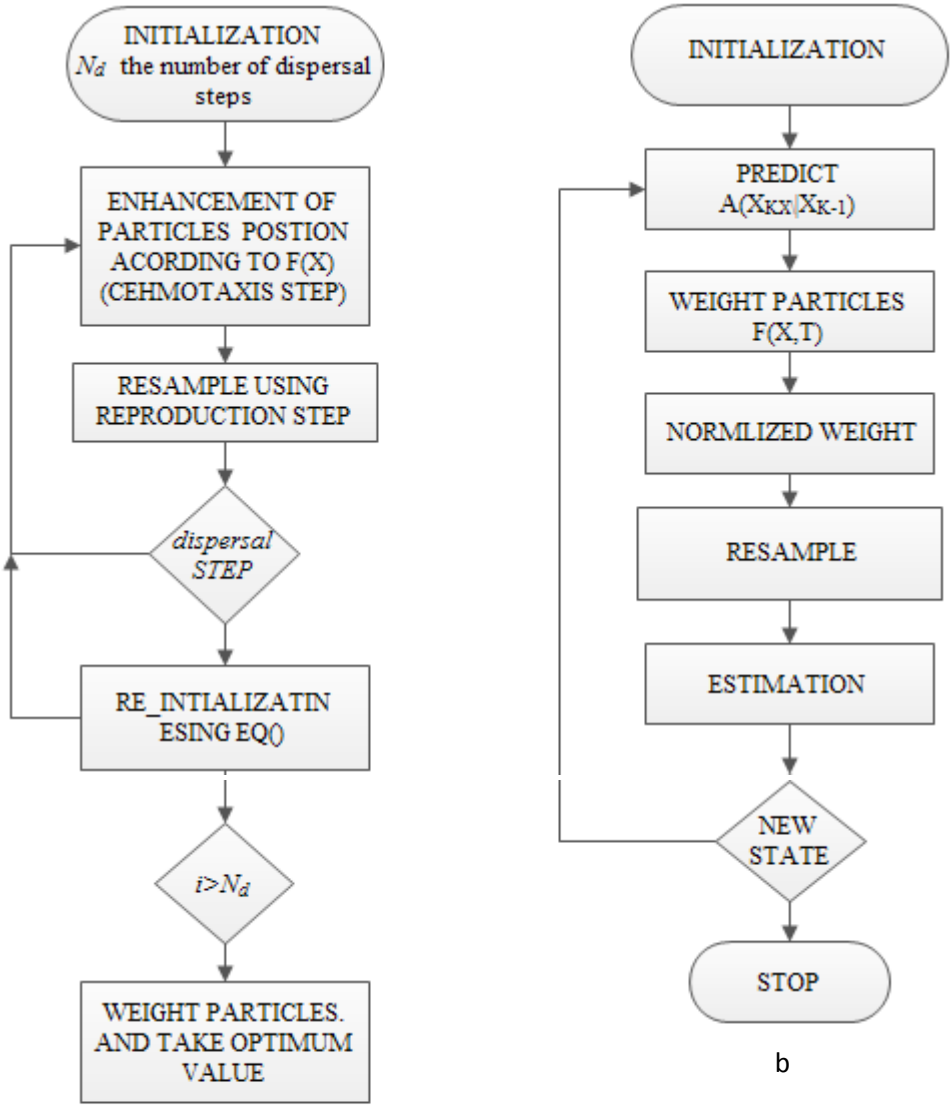


Figure 3.3: (A): BFO Flowchart , (B) PF Flowchart

1. Both PF and BFO have initialization step, weight step and resample step.
2. BFO doesn’t have prediction step.
3. In BFO, weight function ($F(x)$ objective function) is time invariant, however, in PF, weight function is time varying ($F(x\,,t)$) .
4. In BFO, resample step can be implemented by reproduction step. In PF, resampling step depends on algorithms proposed in section 3.4
5. BFO doesn't need normalize stage.
6. In BFO, dispersal step, that mean bacteria go to new environment, can represent new state.
7. In BFO, Chomotaxis step doesn't only weight bacteria, but also it enhances their weight using tumble and swim movement.

From previous comparison, it is easy to combine PF with BFO. To make this combination, do the following changes in PF :

1. Use BFO reproduction step in resample stage.
2. Avoid normalize step.
3. Use chemotaxis algorithm in weight step.
4. Use eq.(3.14) for estimation:

CHAPTER 4: VIDEO OBJECT TRACKING

4.1 Introductions

Video tracking is the process of locating a moving object (or multiple objects) over time using a camera as shown in Figure(4.1). It has a variety of uses, some of which are: human-computer interaction, security and surveillance, video communication and compression, augmented reality, traffic control, medical imaging and video editing Video tracking is a time consuming process due to the amount of data that is contained in video. Adding further to the complexity is the possible need to use object recognition techniques for tracking.

To perform video tracking an algorithm analyzes sequential video frames and outputs the movement of targets between the frames. There are a variety of algorithms; each has its own strengths and weaknesses. The intended use is considered important when choosing which algorithm to use. There are two major components of a visual tracking system: target representation and localization, as well as filtering and data association.

Target representation and localization are mostly bottom-up processes. These methods give a variety of tools for identifying the moving object. Locating and tracking the target object successfully depends on the algorithm. For example, using tracking is useful for identifying human movement because a person's profile changes dynamically. Typically the computational complexity for these algorithms is low. The following algorithms are the most common target representation and localization methods:

- Blob tracking: segmentation of object interior (for example blob detection, block-based correlation or optical flow)
- Kernel-based tracking (mean-shift tracking): an iterative localization procedure based on the maximization of a similarity measure (Bhattacharyya coefficient).
- Contour tracking: detection of object boundary (e.g. active contours or Condensation algorithm)
- Visual feature matching: registration

Filtering and data association are mostly top-down processes, which involves incorporating prior information about the scene or object, dealing with object dynamics, and evaluation of different hypotheses. These methods allow the tracking of complex objects along with more complex object interaction like tracking objects moving behind obstructions. Additionally the complexity is increased if the video tracker (also named TV tracker or target tracker) is not mounted on rigid foundation (on-shore). On the other hand, a moving ship (off-shore), where typically an inertial measurement system is used to pre-stabilize the video tracker to reduce the required dynamics and bandwidth of the camera system. The computational complexity for these algorithms is usually much higher. The following are some common filtering algorithms:

- Kalman filter: an optimal recursive Bayesian filter for linear functions subjected to Gaussian noise.
- Particle filter: useful for sampling the underlying state-space distribution of nonlinear and non-Gaussian processes.

In this thesis, video object tracking based on color is implemented using a new OBPF .

Figure 4.1: Car Tracking In Video Frames

4.2 Video object tracking using PF:

To achieve video object tracking by PF as in [7], [9], [10], Bootstrap PF is used as the following:

1. Applying PF algorithm to each frame of the video in sequence.
2. Sampling the particles of the first frame in pixels and organizing them in set X.
3. Predicting set X in every frame using Randomly Gaussian Noise U.

$$X_t = A * X_{t-1} + B * U_{t-1} \quad (4.1)$$

4. Using the value of the pixel for weighting the particle in comparison with previously taken value of the tracked video object by using Gaussian function
5. Normalizing the weights.
6. Re-sampling the particles according to their new weights.
7. Estimating the location of the video object using eq. (3.14).
   Generate the new set,$\{x_i(k), W_i(t)\} with \hat{} W_i(t) = 1/N_p\}$ and goto to next frame.

Many researches were made to improve performance of particle filter in video object tracking. Some of this studied the techniques of weight particles [29],[30],[31] . Another tried to improve the algorithm of particle filter by adding mean shift [8]and combining it with optimization algorithms[7],[8],[9],[10],[11]. In this thesis, the algorithm of PF is improved.

.

In this thesis, simple weighing cues are used to show the strength of combining particle filter with optimization algorithms. Moreover, two cue for weighing are used:

1) Histogram color of the particle -pixel- value H_i is compared with wanted histogram color pixel H_{target}:

$$D_H = \text{norm}(H_k^i - H_{target}) \quad (4.3)$$

$$W_H^i = \frac{1}{\sqrt{2\pi\sigma_{color}}} e^{\frac{-D_H^2}{2\sigma_{color}^2}} \quad (4.4)$$

2) The position of the particle X_i is compared with the best previous position X_{target}which equals previous location of the tracked video object.

$$D_p = \text{norm}(X_k^i - X_{target}) \quad (4.5)$$

Where D_p^2 is distance in pixels.

$$W_P^i = e^{\frac{-D_P^2}{2\sigma_{postion}^2}} \quad (4.6)$$

The weight of particle is calculated according to equation:

$$W_k^i = W_H^i * W_P^i \quad (4.7)$$

In this thesis, two cue is used in saved movie, but first cue is just used in real time video. In addition, To increase the robust of tracking, optimization algorithms are combined with PF.

4.3 Video object tracking using particle filter based on PSO

Combining particle filter with PSO is proposed in [6]. This combination is used in video object tracking as in [9]. In this thesis the same algorithm is performed, but the number of PSO iteration (N_i) is controlled. The algorithm is shown below:

- Initialization: Generate the initial states,X
  Generate the initial weights, W
  Generate the initial particle speed, V

For $i = 1$ ... ...the number of frame

- Prediction: give positions for particles using eq.(4.1)

$$X_k = A * X_{k-1} + B * U_{t-1}$$

For k: 1…N_i

- Weights of particle (update state): generate the likelihood, $c(y(t)|xi(k))$ using the current particle and measurement

- Resample particles:
  1. update $P_{best,i}$, G_{best}.
  2. find new positions of particles used eq.(2.1) and eq. (2.2)

  End for

- Estimation: estimate the best value using eq. (3.11).

End for

4.4 Video objects tracking using particle filter based on BFO

Combining PF with BFO on video object tracking use bacteria as particles, their health as weights and bacterial foraging strategy process as below [7]:

Elimination and dispersal:

We consider video frames elimination and dispersal events. Each frame has new measurements and new predictions about the particles.

Chemotaxis step:

We use Chemotaxis step to improve the weights of the particles -health of bacteria - before re-sampling. Our ability to change the number of Chemotaxis steps and the length of swim enables us to control the improvement of weight more than mean shift does as in [8].

Swarming:

Since swarming provides social behavior between bacteria, it improves the weight of particle. Strength particles increase the weight of neighbor particle. Therefore, particles center in groups.

Reproduction:

Reproduction is used to resample particle. Consider $\mathbf{S_d}$, low weighted particles, as dead bacteria and $\mathbf{S_h}$, high weighted particles, as bacteria alive and splitting. $\mathbf{S_d}$ does not equal $\mathbf{S_h}$, and $\mathbf{S_h}$ is able to split more than a time to keep the swarm size constant. The ability to re-produce more than once in one iteration and to choose the number of $\mathbf{S_h}$ **and** $\mathbf{S_d}$ provides high control for re-sampling step in particle filter.

The algorithm is shown below:

- Initialization: Generate the initial states , X
  Generate the initial weights, W
  $Nc : chemotactic\ steps.$
  $Ns : swim\ length.$
  $Nre : reproductive\ steps.$

For i=1……..the number of frame

- Prediction : give positions for particles using eq.(4.1)

$$X_k = A * X_{k-1} + B * U_{t-1}$$

For k: 1…N_{re}

- Weights of particle (update state): generate the likelihood, $c(y(k)|x(k))$ using the current particle and measurement

For k: 1…N_c

Perform chemotaxis step:

1. Perform tumble (indicate direction of step)
2. Swim until improve fitness or arrive the length of swim (N_s).

End for

The detail of tumble and swim is proposed in BFO algorithm.

- Resample particles:

The S bacteria with the highest J_{health} values die and the remaining S bacteria with the best values split.

End for

- Estimation: estimate the best value using eq. (3.11).

End for

Chapter 5: THE ROBOT EYE DESIGN AND RESULTS

5.1 Introduction:

The Object tracking using camera is an important task within the field of computer vision. The increase of high powered computers, the availability of high quality inexpensive video cameras and the increasing need for automated video analysis has generated a great deal of interest in object tracking algorithms.

In this study, the real time tracking algorithm for a robot eye is designed as position controller**.** Like any closed control system, the robot eye system has three main blocks: plant, controller and feedback as shown in Figure (5.1).

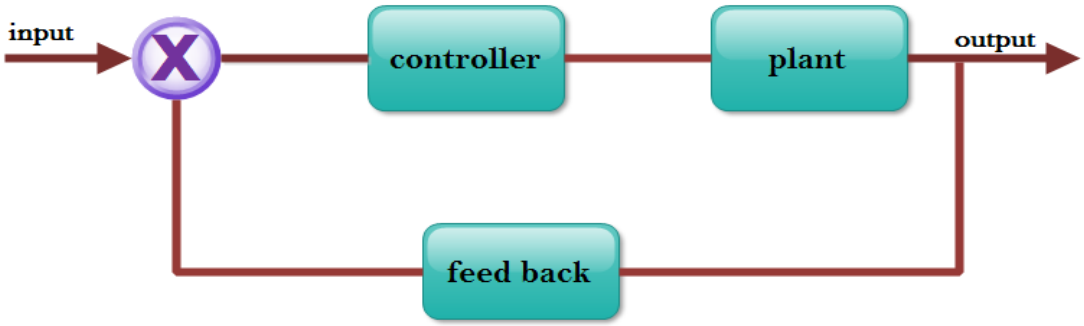


Figure 5.1: Basic Block Of Closed loop System

Figure (5.2) shows the closed system of robot eye that aims to maintain the OBI in the center of camera using its color . USB Camera sends acquired image to the visual C++ program which presented by the researcher. The position of OBI represented as feedback signal is estimated in the acquired image by applying AOBP or OBPF using OPENCV tool in visual C++. After that, visual C++ program implements FPID control to calculate the pulse width modulation (PWM) signals for each motor to centralize OBI. Then, visual C++ program sends the PWM signals to Arduino through USB connection. Finally, Arduino drives two servo motors.

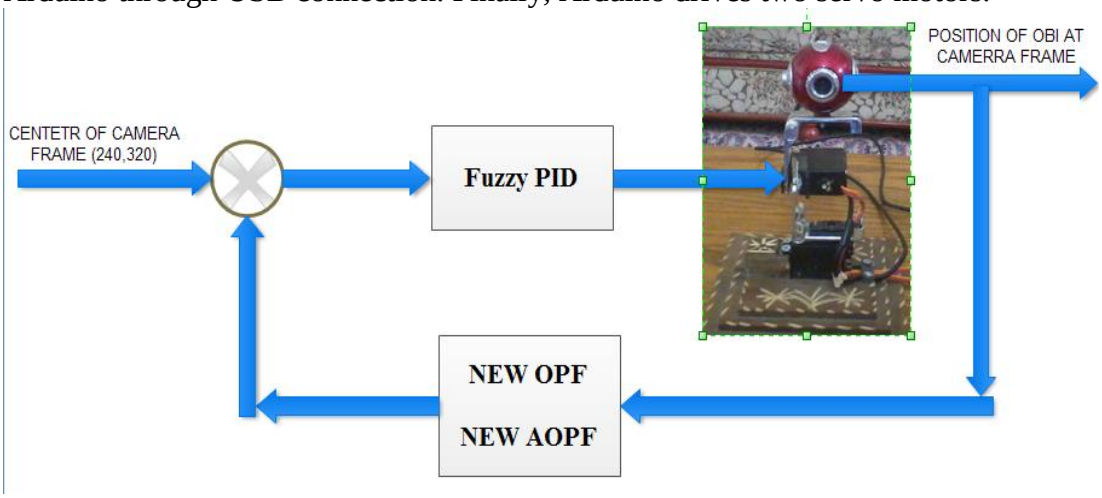


Figure 5.2: Closed Control System Of The Robot Eye

5.2 Plant

The implementation of the AROBOT EYE system is divided into hardware and software.

- The hardware used in the proposed architecture includes the following items:

HP laptop (ProBook 450s):

The specifications of laptop is used in experiment are Intel core i7 processor and 8GB DDR3. It is used to receive image from camera and to implement AOBPF on it for detecting position of OBI. Moreover, FPID control is implemented to calculate the velocities of two motors. It sends the information to driver (Arduino) using USB.

Low cost Webcam (25fps):

Webcam is used as sensor that takes image and sends it to laptop. It has resolution 480*640.

Figure 5.3: Low Cost Webcam

Two servo motors (TOWE PRO MG995):

Two servo motors are used to move camera to centralize OBI. However Towepro mg995 is standard servo, it is hacked to be continuous rotation servo motor. To control motors, the PWM signal is used as following:

- At 1500 Hz ,motor is stop.
- At higher frequency, motor turns counterclockwise.
- At lower frequency motor turns clockwise.
- The period time of signal is 20ms.
- At 1300 Hz, motor has maxima speed at right direction. At 1700 Hz, motor has maxima speed at left direction.

Figure 5.4: Servo Motor(Mg995)

ARDUINO (MEGA 2560):

Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. Using an Arduino simplifies the amount of hardware and software development you need to do in order to get a system running. The Arduino hardware platform already has the power and reset circuitry setup as well as circuitry to program and communicate with the microcontroller over USB. On the software side, Arduino provides a number of libraries in C++ to make programming the microcontroller easier.

Arduino (MEGA 2560) is used as driver for two servo motors which takes information from laptop.

Figure 5.5: Arduino(MEGA 2560)

- The software used in the proposed architecture includes the following items:

1) Matlab software:

- It is used to implement video object tracking by OBPS and compares it with previous algorithm.
- It is used to simulate FPID control.

2) Visual studio C++: two libraries are used:

- OpenCV library[33]:

  Open Source Computer Vision Library is a library of programming functions mainly aimed at real-time computer vision, developed by Intel, and now supported by Willow Garage and Itseez. It is used to implement video object tracking in real time.

- Fuzzy control library:

  It is programmed by researcher to implement FPID controller in real time.

5.2 Controller

5.2.1 Conventional PID controllers

PID controllers are the most widely-used type of controller for industrial applications. They are structurally simple and exhibit robust performance over a wide range of operating conditions. In the absence of the complete knowledge of the process these types of controllers are the most efficient of choices.

As shown in Figure(5.6), the three main parameters involved are Proportional (P), Integral (I) and Derivative (D). The proportional part is responsible for the following desired set-point, while the integral and derivative part account for the accumulation of past errors and the rate of change of error in the process respectively.

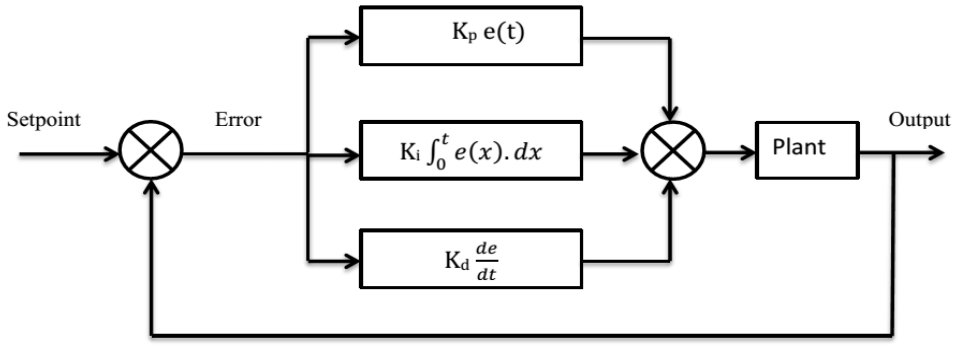


Figure 5.6: Basic Block Of PID Controler

A standard equation of PID controller is

$$u(t) = K_b(\, e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{d}{dt}\mathrm{E}\,(t) \qquad (5.1)$$

In this thesis digital practical form is used [32]:

$$u[k] = U_0 + K_P\, e[k] + K_d(\, e[k] - e[k-1)] + K_i \sum_0^k e[k-2] \qquad (5.2)$$

Manual tuning PID:

If the system must remain online, the tuning method to be used is such as:

1) Set K_d and K_i values to zero.
2) Increase the K_P until the output of the loop oscillates. then the K_P should be set to approximately half of that value.
3) Then increase K_i until any offset is corrected in sufficient time for the process. However, too much K_i will cause instability.
4) Finally, increase K_d, if required, until the loop is acceptably quick to reach its reference after a load disturbance.

5.2.2 Fuzzy logic controller:

The basic parts of every fuzzy controller are displayed in the following Figure (5.7). FLC is composed of a fuzzification interface, knowledge base, inference engine, and defuzzification interface.

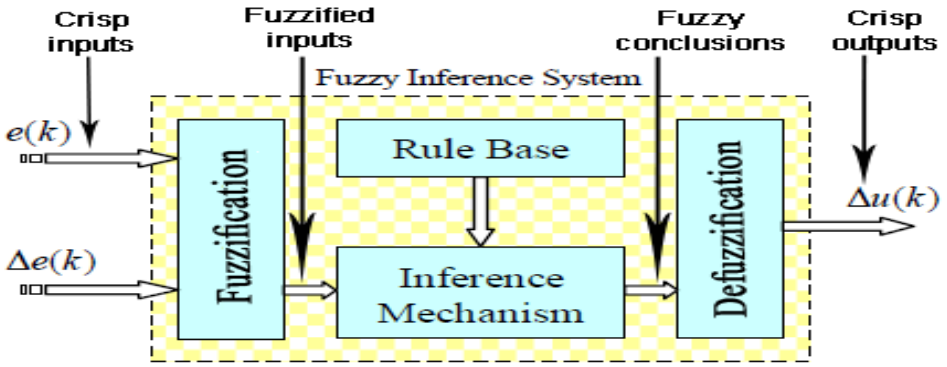


Figure 3.7: Basic Parts Of Fuzzy Logic Controller

The fuzzifier maps the input crisp numbers into the fuzzy sets to obtain degrees of membership. It is needed in order to activate rules, which are in terms of the linguistic variables. The inference engine of the FLC maps the antecedent fuzzy (IF part) sets into consequent fuzzy sets (THEN part). The defuzzifier maps output fuzzy sets into a crisp number, which becomes the output of the FLC.

For the development and implementation of a fuzzy controller, the required inputs and outputs must be determined first. The next step is to determine (approximately) the number, shape and width of the membership functions of the fuzzy inputs and outputs. Then, the fuzzy rules must be set in a linguistic form that will describe the relation between the inputs and the outputs. The final steps are to defuzzyfy the output function to produce a discrete value [34].

There are a lot of inference methods which deals with fuzzy inference like: Mamdani method, Larsen method, Tsukamoto method, and Takagi-Sugeno_Kang (TSK) method. In this research ,Mamdani method is used.

5.2.3 Supervisory Fuzzy Control:

There are four types of Fuzzy supervisory control:
1. Fuzzy replaces PID.
2. Fuzzy replaces operator.
3. Fuzzy adjusts PID parameters.
4. Fuzzy adds to PID control.

In this thesis, the fuzzy adjusts PID control is used where fuzzy logic used to adjust of PID parameter as shown in Figure(5.8).

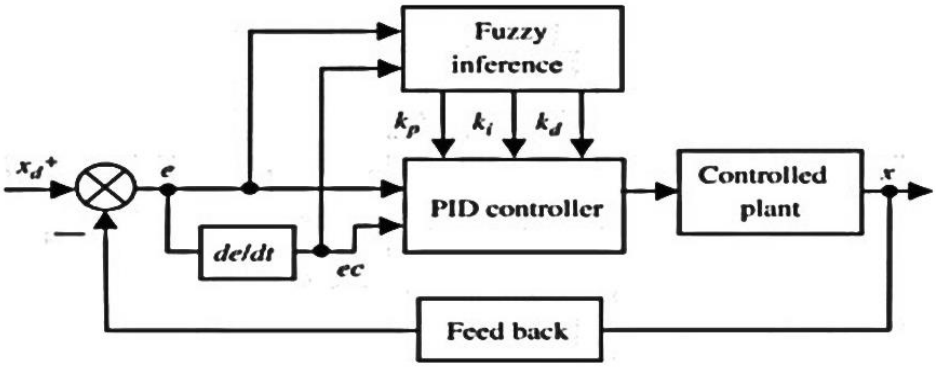


Figure 5.8: Block Diagram Of FPID Controller

5.2.4 FPID design for the robot eye:

To design FPID controller for a robot eye, follow the next steps:

1- Determining the process states and control variables:

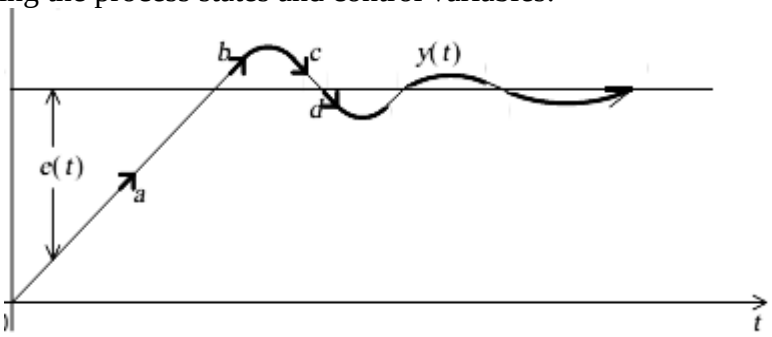


Figure 5.9: Analysis Of The Response Of The System

The physical process is a position control. Thus, point r is a target position to be reached; it is the center of camera frame (240,320). In this application, the process state is the overall controlled system output, y (t), which is also position. The system is multi input multi output (MIMO), because it has two inputs, error and change of error, and it control two motors.

2- Fuzzification:

Fuzzification strategy is used to create the degree of fuzzy membership function for each inputs and outputs. Two motors have the same memberships. Tow input sets are used in controller as shown in Figure(5.10) and Figure(5.11).

1) Error set:

$$Error(k) = center\ of\ imag\ e\ frame - \ postion\ of\ tracking\ object$$

Range: [-320,320];
Memberships: {NB, NM, NS, ZERO, PS, PM, PB}

2) Change of error:

$$Derror(k) = Error(k) - \ Error(k-1)$$

Range: [-220,220];
Memberships: {NB NM, NS, ZERO, PS, PM, PB}.

Triangular memberships are used for NM, NS, ZERO, PS and PM. SMF membership is used for PB. ZMF membership ship is used for NB. Types of memberships are identified experimentally.

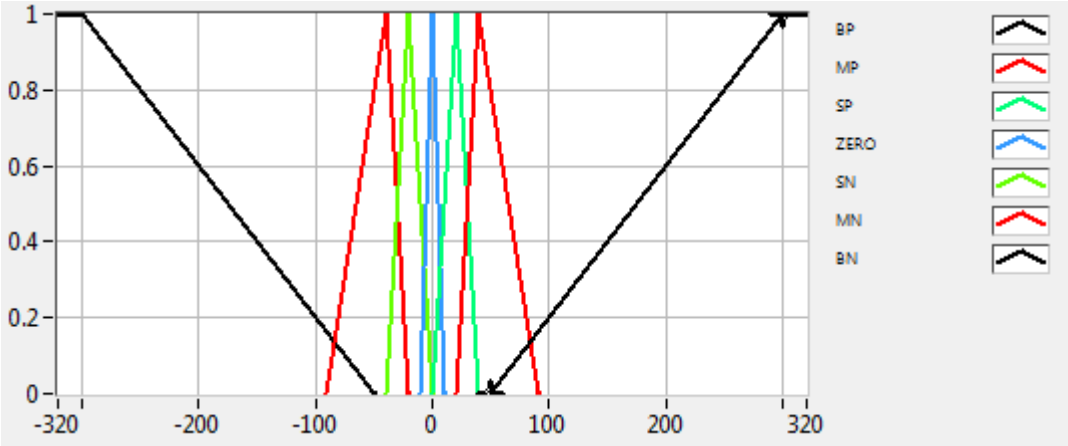


Figure 5.10: Error Membership Functions

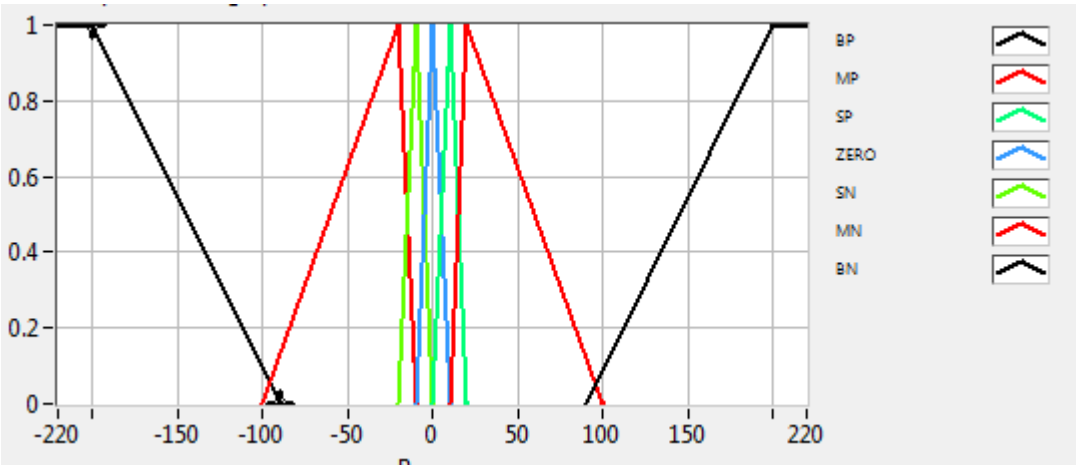


Figure 5.11: Change Of Error Membership Functions

Three output sets, PID parameters, are used as shown in Figure(5.12) , Figure(5.13) and Figure(5.14). The range of each set is identified by using manual tuning PID method:

1) KP: Range [0,.28]

2) KI : Range [0,.028]

3) KD: Range [0,.0008]

Memberships of each output are {S, MS, M, MB, and L}. Triangular member ships are used for MS, M and MB. SMF membership is used for L. ZMF membership is used for S. Two motors have the same memberships. Types of memberships are identified experimentally.

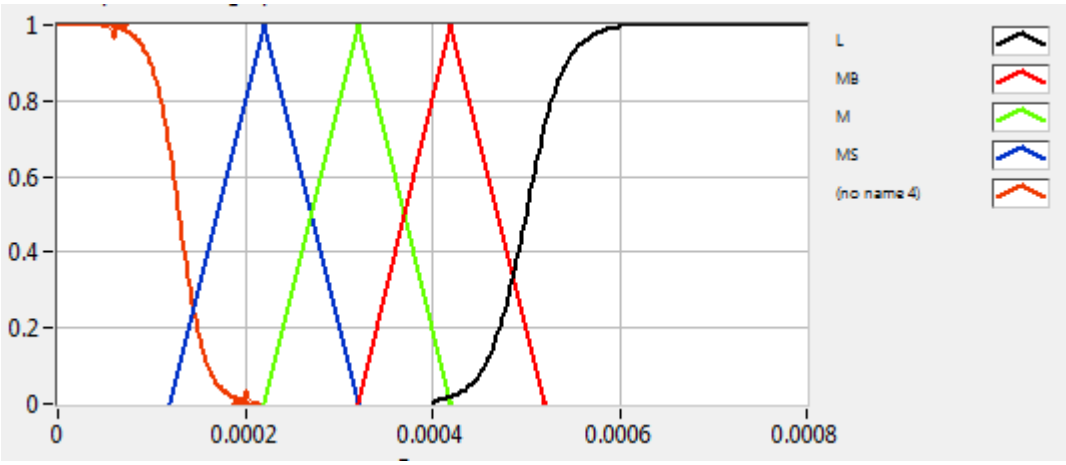


Figure 5.12: Kd Membership Functions

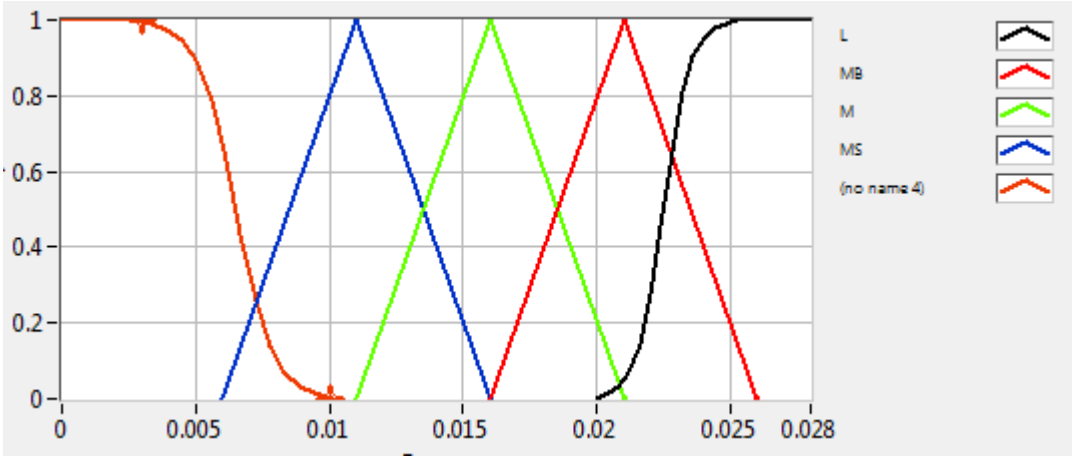


Figure 5.13:Figure 20: Kp Membership Functions

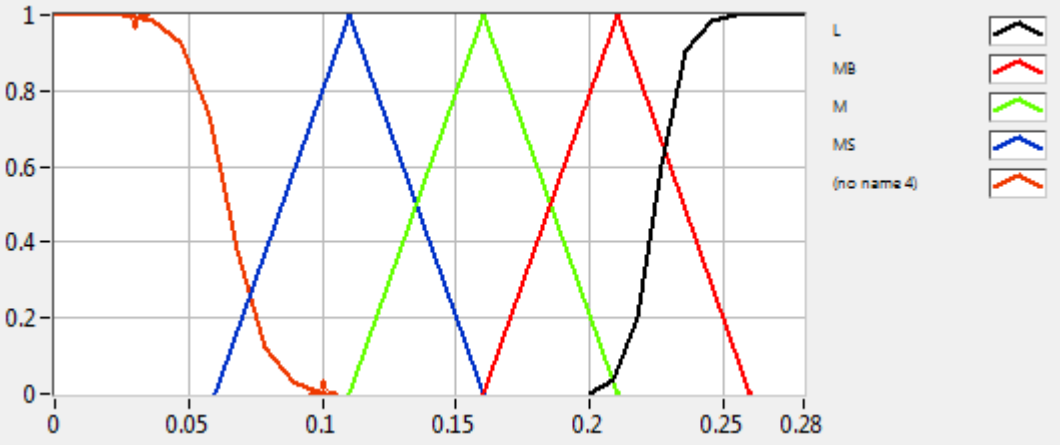


Figure 5.14: Ki Membership Functions

3- Rule base design:

After reviewing previous studies such as [13][32], the researcher acquires experience about fuzzy control rules and then the fuzzy control rules of the proposed algorithm have been derived experimentally. All rule bases derived in the same way. The full sets of rules are summarized in the following Kmaps:

Table 5.1: Kp Kmap

K_p		CHANGE OF ERRORR						
		BN	MN	SN	ZERO	SP	MP	BP
ERORR	BN	MB	MS	S	S	S	MS	MB
	MN	MB	M	MS	S	MS	M	MB
	SN	L	MB	M	MS	M	MB	L
	ZERO	L	L	MB	M	MB	L	L
	SP	L	MB	M	MS	M	MB	L
	MP	MB	M	MS	S	MS	M	MB
	BP	MB	MS	S	S	S	MS	MB

Table 5.2: Ki Kmap

K_i		CHANGE OF ERRORR						
		BN	MN	SN	ZERO	SP	MP	BP
ERORR	BN	MS	MB	B	B	B	MB	MS
	MN	MS	M	M	B	M	M	MS
	SN	S	MS	MS	M	MS	MS	S
	ZERO	S	S	S	MB	S	S	S
	SP	S	MS	MS	MS	MS	MS	S
	MP	MS	M	M	B	M	M	MS
	BP	MS	MB	B	B	B	MB	MS

Table 5.3: Kd Kmap

K_D		CHANGE OF ERRORR						
		BN	MN	SN	ZERO	SP	MP	BP
ERORR	BN	MS	MB	B	B	B	MB	MS
	MN	MS	M	M	B	M	M	MS
	SN	S	MS	MS	M	MS	MS	S
	ZERO	S	S	S	MB	S	S	S
	SP	S	MS	MS	MS	MS	MS	S
	MP	MS	M	M	B	M	M	MS
	BP	MS	MB	B	B	B	MB	MS

4- Defuzzification process:

At the final step, the determination of both the applicable rules for any fuzzy controller inputs (E, CE) and the fuzzy control outputs are performed by inference engine strategy.

The fuzzy results obtained from the above rules should be defuzzified to convert them into meaningful crisp values. In general, there are several methods used for defuzzification such as centroid of area (COA), maximum method (MM), mean of maximum (MOM), and bisector of area (BOA). In this theses, the center of area method is used. It is expressed as:

$$UCOA = \frac{\sum_{i=0}^{N} \mu_i(u).u}{\sum_{i=0}^{N} \mu_i(u)} \quad (5.3)$$

5.3 Feedback of the system:

5.3.1 Optimized bootstrap particle filter (OBPF):

Bootstrap filter predicts the new position of particle in one step using this model $p(xk|\ xk-1)$ as important distibution. In the OBPF, there are two steps for prediction. First step uses the same relation $p(xk|\ xk-1)$, but second step uses a new model $p(xk|yk)$ to predict particle postion. New model depends on adaptive Chemostic step proposed in section 3.5, where particles move in random directions, but in Known magnitude. New relation $xk{\sim}p(xk|yk)$ is implanted by using following step:

1 . Weigh particles.
2. Update θ_{best} andθ_{gbest}.
3. Update particle using eq. (2.7) and eq. (2.8)

$$C(i) = w * norm(\Delta(i)) + c_1 * norm(\theta_{best} - \theta_i) + c_2 * norm(\theta_{gbest} - \theta_i)$$

$$\theta^i(j+1,k,l) = \theta^I(j,k,l) + c(i) * \frac{\Delta(i)}{\sqrt{\Delta(i)^T * \Delta(i)}}$$

Besides, PSO is used in resampling step **.** In order to construct the OBPF, follow next steps:

1. Initialization: Generate the initial state ,$x_i(0)$
   Generate the process noise, $w_i(t)$
2. Prediction I: give positions for particles according to the model and previous position as in eq.(3.13).

$$xi(k) = A(x_i(k-1), u(k-1), w_i(k-1))$$

3. Prediction II : give positions for particles according to the weights of particle

$$xk \sim p(xk|yk)$$

4. Weights of particle (update state): generate the likelihood, using the current particle and measurement
5. Resample particles: resample the set of particles using PSO algorithm.
6. Estimation: estimate the best value using eq.(3.11).

5.3.2 Video object tracking using OBPF:

The algorithm of using OBPF in video object tacking is shown below:

- Initialization: Generate the initial states ,X
  Generate the initial weights, W
  Generate the initial particle velocity, V

For j=1……..the number of frame

- Prediction I : give positions for particles using eq.(4.1)
- Prediction II : $xk \sim p(xk|yk)$

For k: 1…N_i

- Weights of particle (update state): using the current particle and measurement using eq.(4.3), eq.(4.4), eq.(4.5), eq.(4.6), and eq.(4.7).
- Resample particles:
- Update x_{best} andx_{gbest}.

1. Find new velocities of particles used eq. (2.1).
2. Find new positions of particles used eq. (2.2).

End for

- Estimation: estimate the best value using eq.(3.11).

End for

5.3.3 AOBPF for real video object tracking:

In real time, Previous algorithm has problem with object has variable speeds. To solve this problem, the first prediction step is improved to update its length every frame using following equations:

$$X_k = A * X_{k-1} + B(k) * U_{t-1} \quad (5.4)$$

$$B(k) = g * Error(k-1) \quad (5.5)$$

$$Error(k-1) = center\ of\ imag\ e\ frame - previous\ postion\ of\ interested\ object\ tracking \quad (5.6)$$

Where g is constant.

5.4 Experiments and results

5.4.1 OBPF test:

Comparison among OBPF , PF-PSO and PF-BFO in object tracking are made . The color of the skin of the object is tracked in sample video which is taken by digital camera and has strong noise , high variance elimination and high similarity between color as shown in Figure (5.15) .

Figure 5.15: Sample Frame Of Video Test

In contrast with Simple approach which proposes solving the problem of degeneracy and impoverishment through using a great number of particle, the researcher proposes the use of a less number of particles for tracking. The researcher believes that is a mean for measuring the strength of tracking technique. In addition, less cost time means better technique for real time. Color standard deviation is used as a measurement tool which introduces less variance more accuracy

Firstly, OBPF and PF-PSO are used with one iteration for frame and PF-PFO is used with following parameter: Nr=1,Nc=1and Ns=1 for frame. The results are shown in Table 5.4 .

Table 5.4 : Result Of OBPF Test With 7 Dtandard Diviation

STANDARD DIVIATION	FILER TYBE	Min number of particles hold tracking	TIME (S)
7	PF_PSO	50	0.0045
	PF_BFO	35	0.0263
	OBPF	20	0.0018

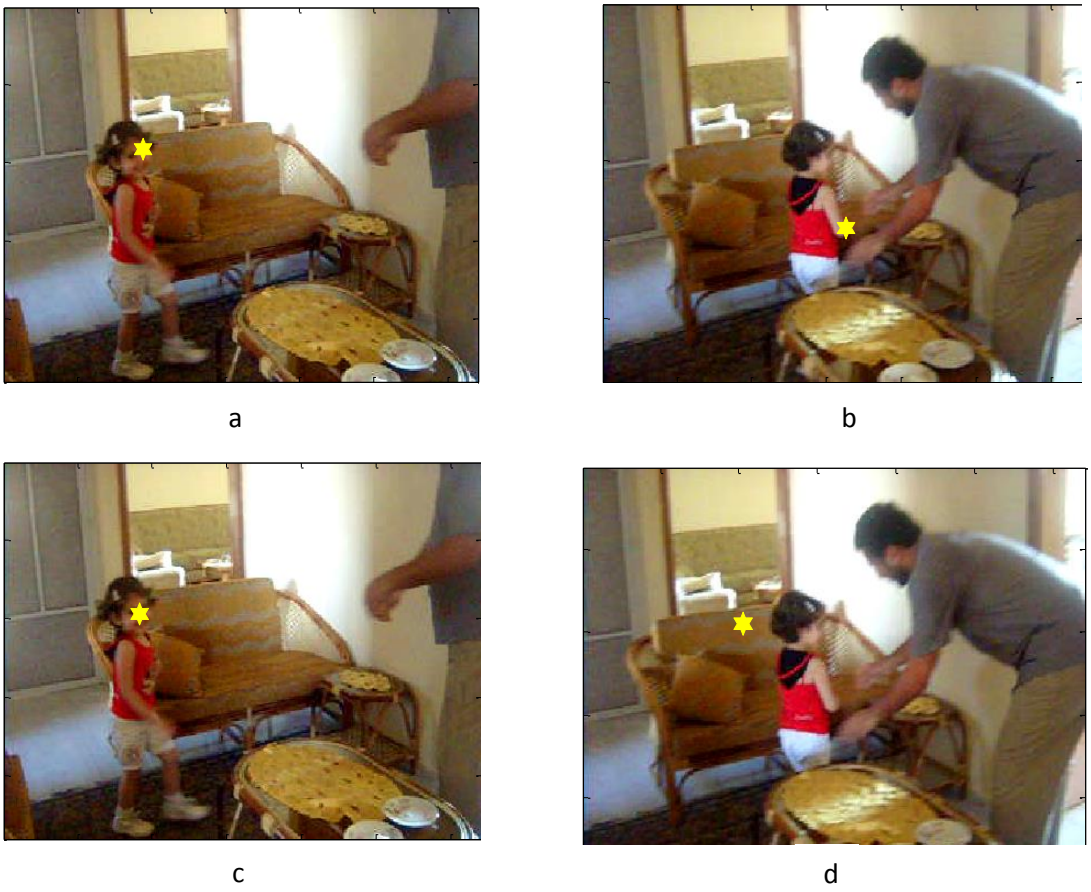
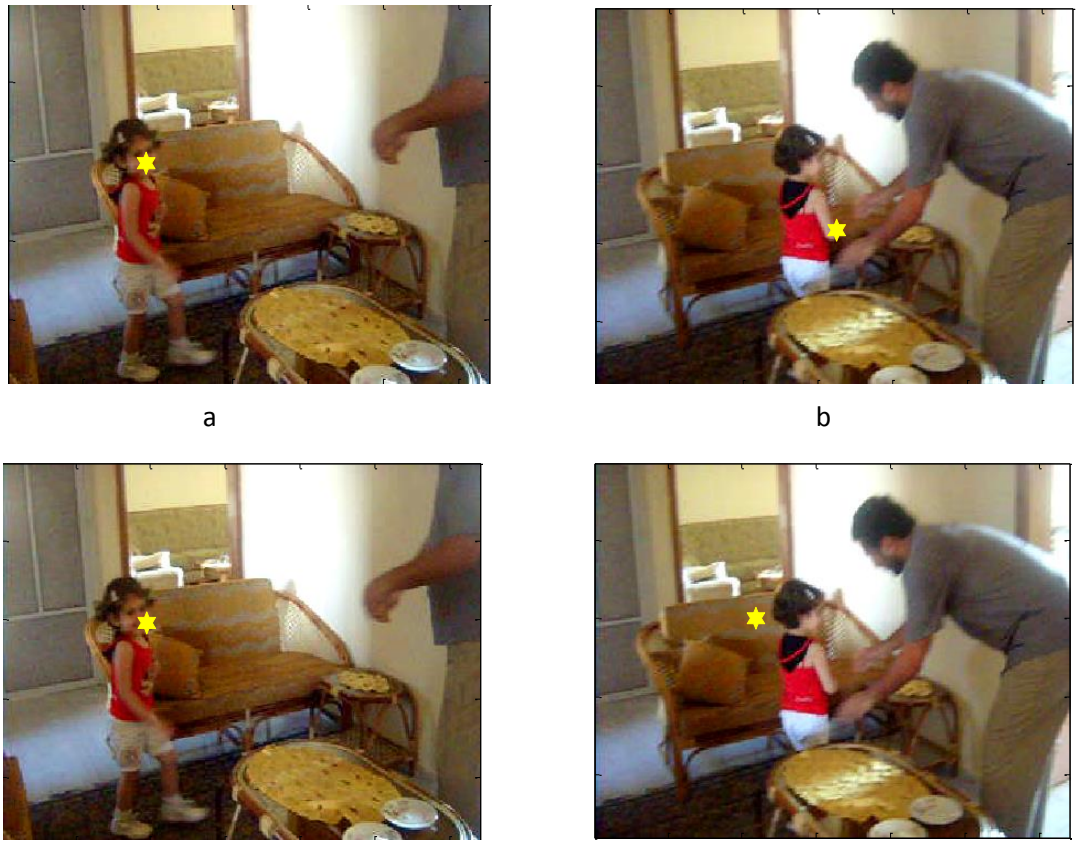


Figure 5.16: The Tesult Of First Test With PF_ PSO

Table (5.4) shows the PF_PSO is fast, but it is the weakest in tracking. Although PF_BFO is stronger than PF_PSO in tracking, it is the slowest. OBPF has the best result. It has both the best time and the strongest tracking.

Figure (5.16) (a) and (b) show that the PF-PSO can track the object in the frames 30 and 60 with 50 particles and color standard variation 7. Although PF-PSO with 30 particles can track the object in frame 30 as shown in Figure (5.16) (c), it can't track in frame 60 as shown in Figure (5.16) (d).



a b

c d

Figure 5.17: The Tesult Of First Test With PF_ BFO

Figure (5.17) (a) and (b) show that the PF-BFO can track the object in the frames 30 and 60 with 35 particles and color standard variation 7. Although PF-BFO with 30 particles can track the object in frame 30 as shown in Figure (5.17) (c), it can't track in frame 60 as shown in Figure (5.17) (d).

a b

Figure 5.18: The Tesult Of First Test With OBPF

Figure (5.18) (a) and (b) show that the OBPF can track the object in frame 30 and 60 with 20 particles.

Secondly, the effect of changing color standard deviation on the number of particle is measured to show the robust of techniques used in tracking. When Color standard deviation is reduced from 7 to 5, we have the result shown in Table (5.5)

Table 5.5: Result of OBPF test with 5 standard deviation

STANDARD DEVIATION	FILER TYBE	Min number of particles hold tracking	TIME (S)
5	PF_PSO	63	0.0048
	PF_BFO	45	0.0341
	OBPF	25	0.0020

Comparison between Table (5.4) and Table (5.5) shows the following. When Color standard deviation is reduced, the number of particle in PF_PSO is increased by 13 particles to hold tracking, the number of particle in PF_BFO is increased by 10 particles and the number of particle in OBPS is increased by only 5 particles. That proofs the strength of OBPF in video object tracking.

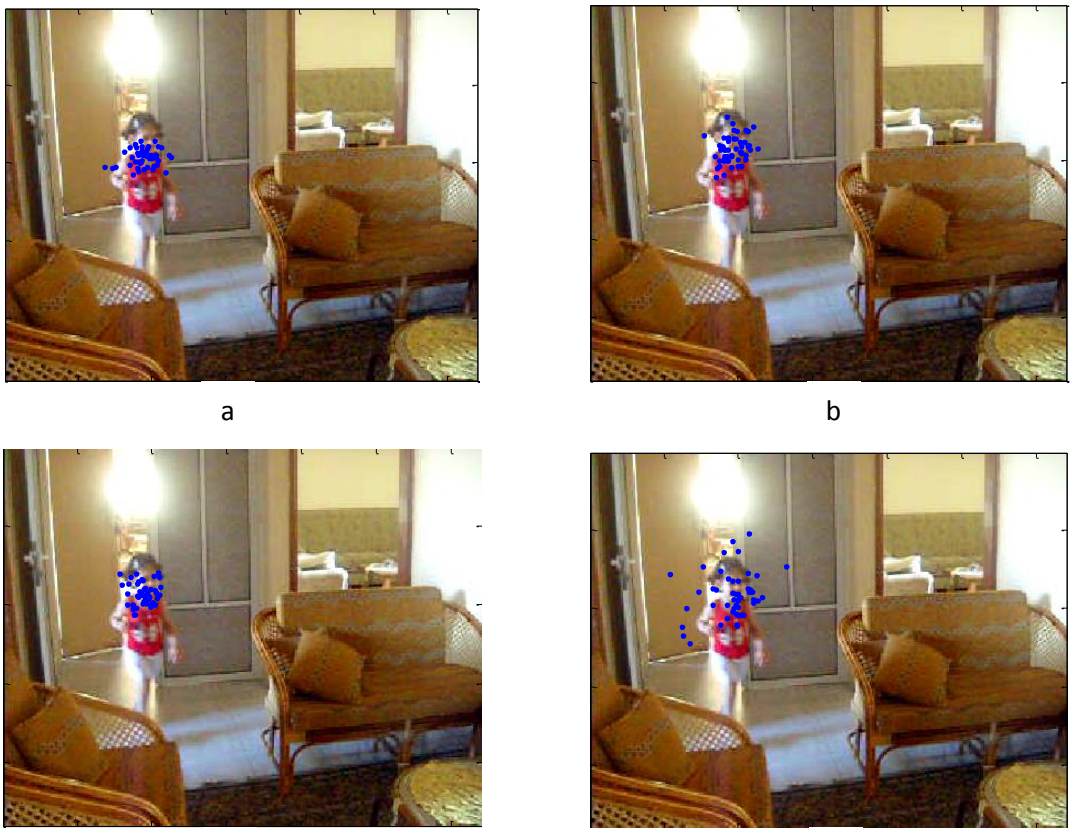


Figure 5.19: Predict Step For Different Particle Filters

Finally, Figure (5.19) shows why OBPF is the best for tracking. At predict step, particles in OPS cover area that is larger than area covered by particles in PF_PSO or PF_BFO. Figure (5.c) shows that at first predict step of OBPF, it covers area that is similar to area covered by PF_PSO as shown in (5.a) or PF_BFO as shown in (5.b). However, Figure (5.d) shows that at second step predict of OBPF, it covers a wider area.

5.4.2 AOBPF test:

Comparison between OBPF and AOBPF in real time object tracking is made. As shown in Figure (5.20), the blue head of pen is tracked, the dark blue circle means estimated position of interested object tracking and Orange circle indicates the blue circle.

Figure 5.20: Sample Frame For Real Time Vedio

Firstly, tracking fixed OBI is used for test. It means that camera moves from dormancy to movement and then return to dormancy. The velocity of camera depends on how OBI far from the center of frame. OBI is selected by using mouse. The test is applied for only horizontal movement. The result is shown in Table (5.6).

Table 5.6: Result Of AOBPF Test

Type of PF	Initial distance of OBI from center in pixel			
	55	110	174	202
OBPF	track	track	lose	lose
AOBPF	track	track	track	track

Table (5.6) shows that OBPF can hold tracking when initial distance error of OBI is 55 pixels or 110 pixels, but it loses tracking when initial distance error of OBI is 174 pixels or above. AOBPF can hold tracking for any initial distance error.

Sequence frames in Figure (5.21) proof that with small value of B (constant in eq. (5.1)), OBPF loses tracking when initial distance error of OBI is 174 pixels.

Sequence frames Figure (5.22) proof that with large value of B (constant in eq. (5.1)), when initial distance error of OBI is 194 pixels, OBPF holds track for short time then system becomes unstable.

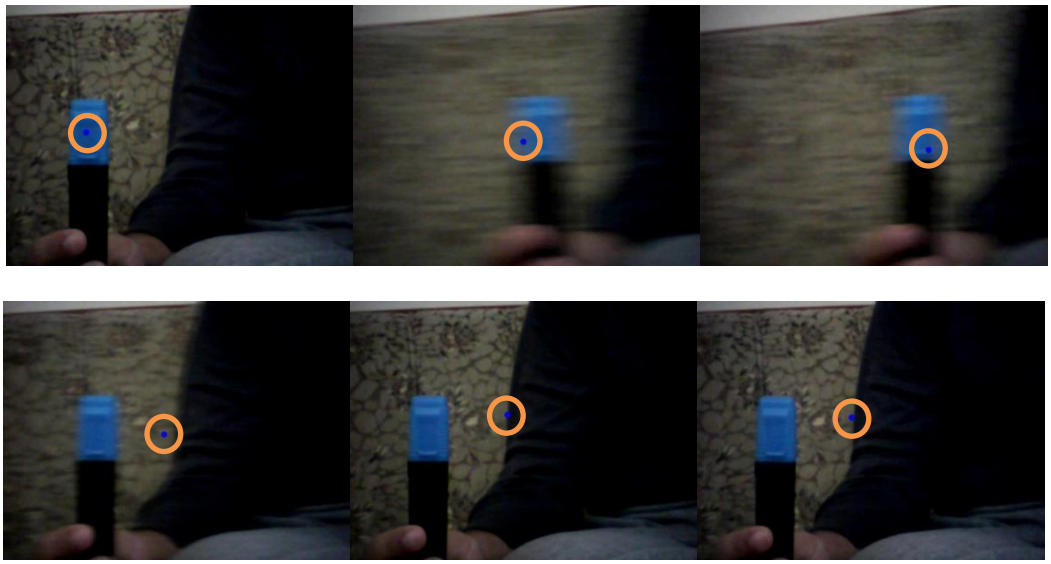


Figure 5.21: OBPF Lost Tracking With Fast Moving Object With Small B

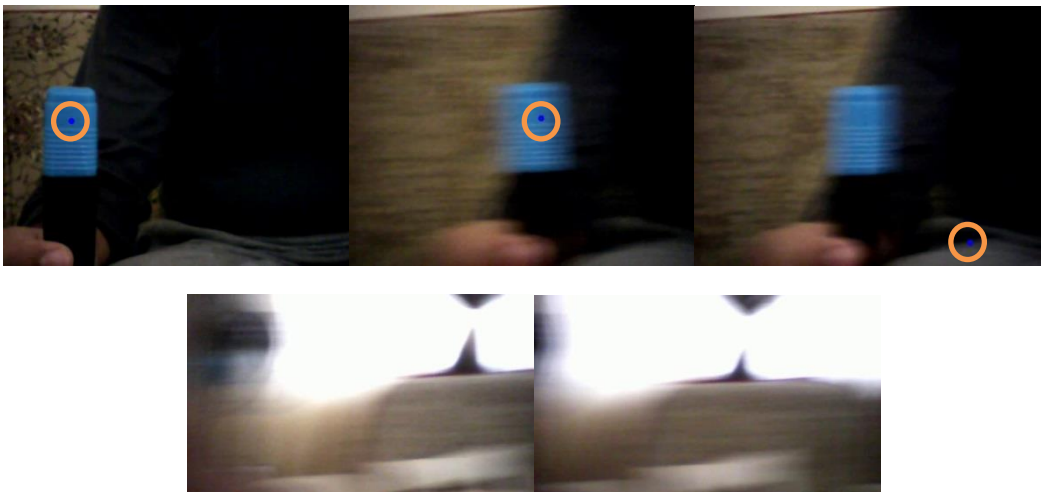


Figure 5.22: OBPF Lost Tracking With Fast Moving Object With Larg B

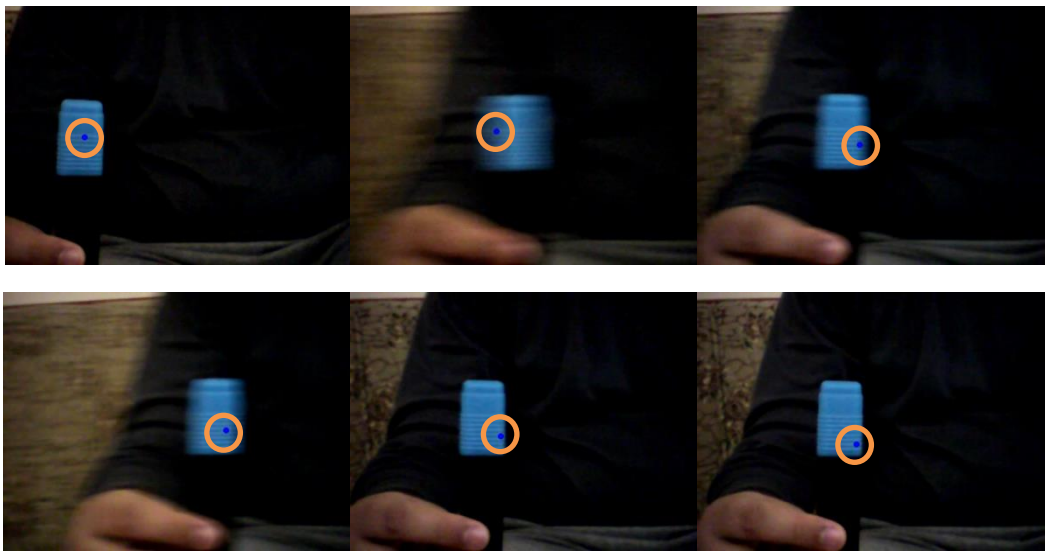


Figure 5.23: AOBPF holds Tracking With Fast Moving Object With Adaptive B

Sequence frames in Figure(5.23) proof that with adaptive value of B(constant in eq.(5.1)), AOBPF holds tracking when initial distance of OBI is 174.

Secondly , OBI is moved with randomly velocity to show which techniques can hold tracking. The result is shown in Figure(5.24) and Figure(5.25) :

Although sequence frames Figure (5.24) proof that with random velocity of camera, OBPF loses tracking, Figure (5.25) shows AOBPF can hold tracking.

Previous tests show how AOBPF more robust than OBPF in real time tracking.

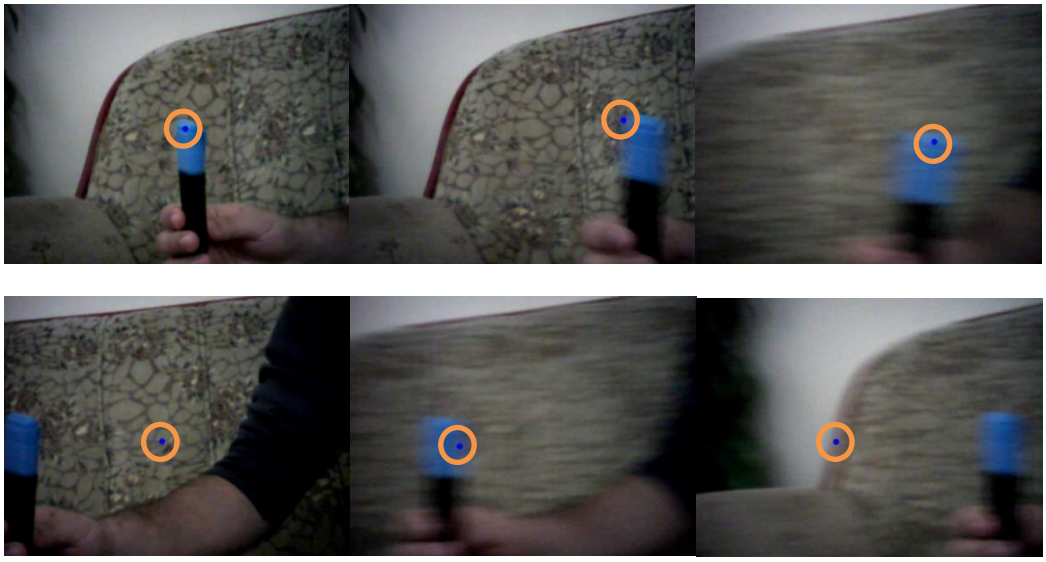


Figure 5.24: OBPF Lost Tracking For OBI Has Random Velocity

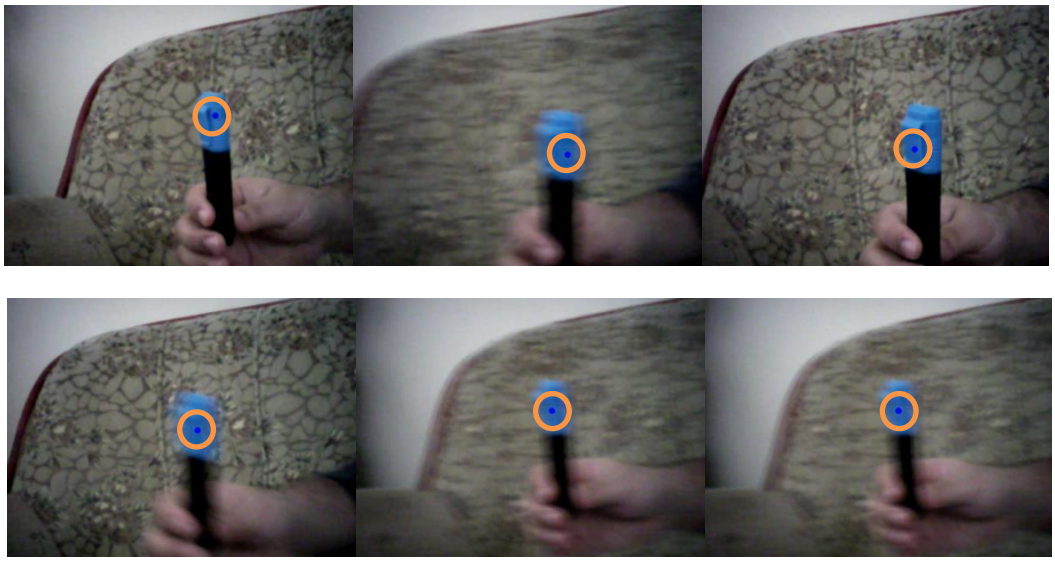


Figure 5.25: AOBPF Hold Tracking For OBI Has Random Velocity

5.4.3 FPID test:

Tracking fixed OBI is used for test. It is meant that camera moves from dormancy to movement suddenly and then return to dormancy to keep the object at the center of frame. OBI is selected by using mouse. After Many experiments, following hints are found:

- Because of large size of cross section of OBI and random change of particle position, the steady state of system is area with mean position at point(240,320) and variance 20 pixels.
- The range (1495 Hz-1505 Hz) of control signal doesn't have any effect. It means zero speed.

Because of the two servo motor have the same tuning of FPID, the behavior of two motor is the same. The test is only applied for horizontal movement (one motor). The result is shown in Table (5.7).

Table 5.7: The Results Of FPID Test

Type of controller	Initial Position of OBI (pixel)	Settling time (frame)
FPID	95	14
	176	14
PID	113	unstable
	181	17

The comparison between Figure (5.26) and Figure (5.27) and Table(5.7) shows the following. When initial position of OBI is 95 pixels, FPID control can lead the system to centralize OBI at frame 14 when PWM signal equals to 1502 HZ. However, when initial position of OBI is 113 pixels, PID control can't lead the system to centralize OBI and motor doesn't stop because PWM signal is higher than 1505 HZ. It is meant that the robot eye loses tracking and becomes unstable.

The comparison between Figure (5.28) and Figure (5.29) and Table(5.7) shows the following. When initial position of OBI is 164 pixels, FPID control can lead the system to centralize OBI at frame 14 when PWM signal equals to 1504 HZ. However, when initial position of OBI is 181 pixels, PID control can lead the system to centralize OBI at frame 17 when PWM signal equals to 1502 HZ . It is meant that FPID make system faster.

Figures show both PID control and FPID control have long rise time. Bad behavior of hacked servo motor causes this problem.

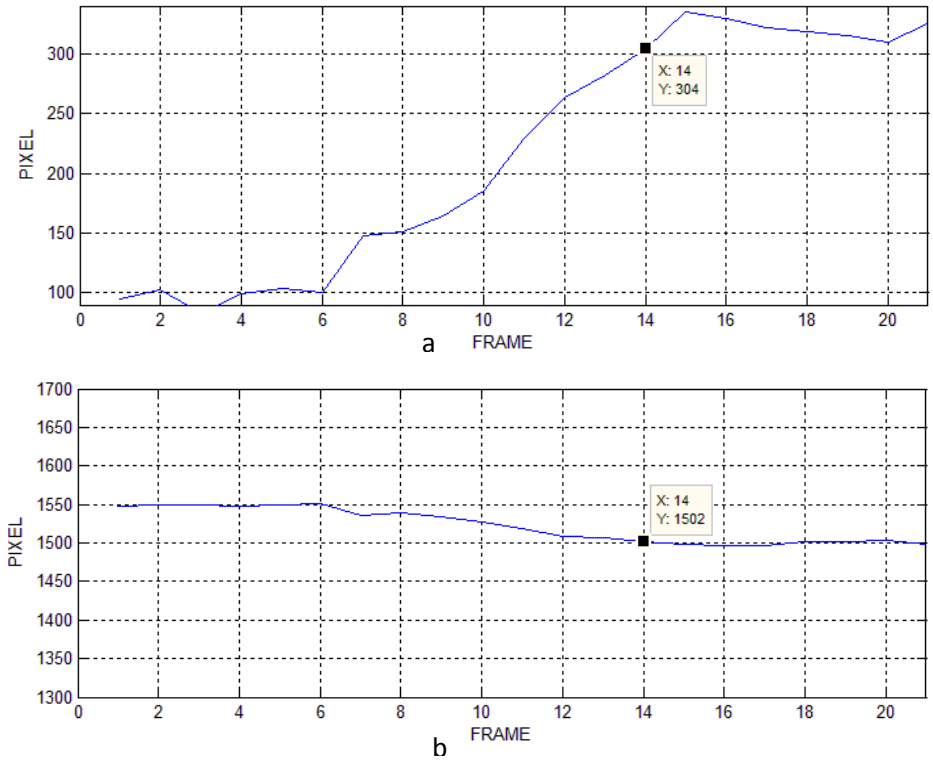


Figure 5.26: The Performance Of FPID When Initial Postion Is 95

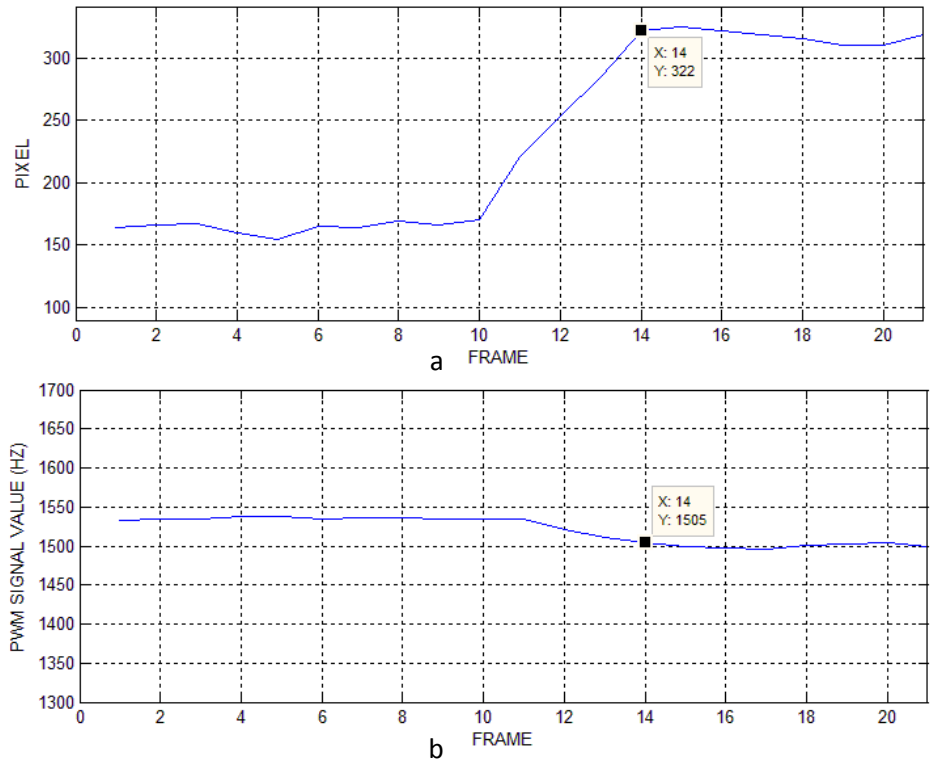


Figure 5.27: The Performance Of FPID When Initial Postion Is 161 Pixels

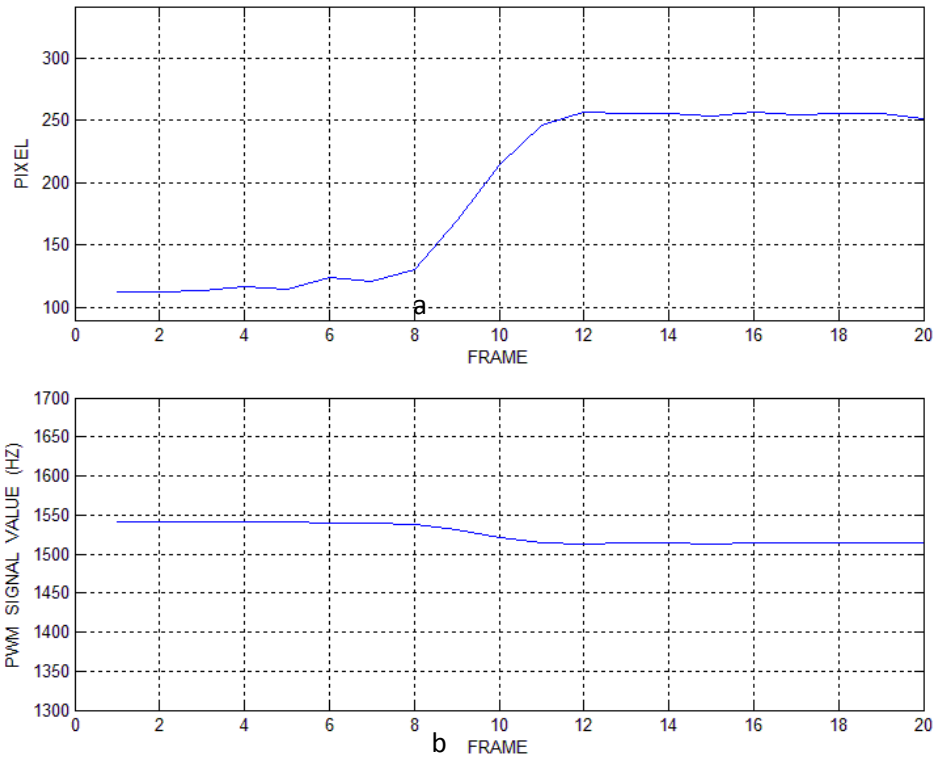


Figure 5.28: The Performance Of PID When Initial Postion Is 113 Pixels

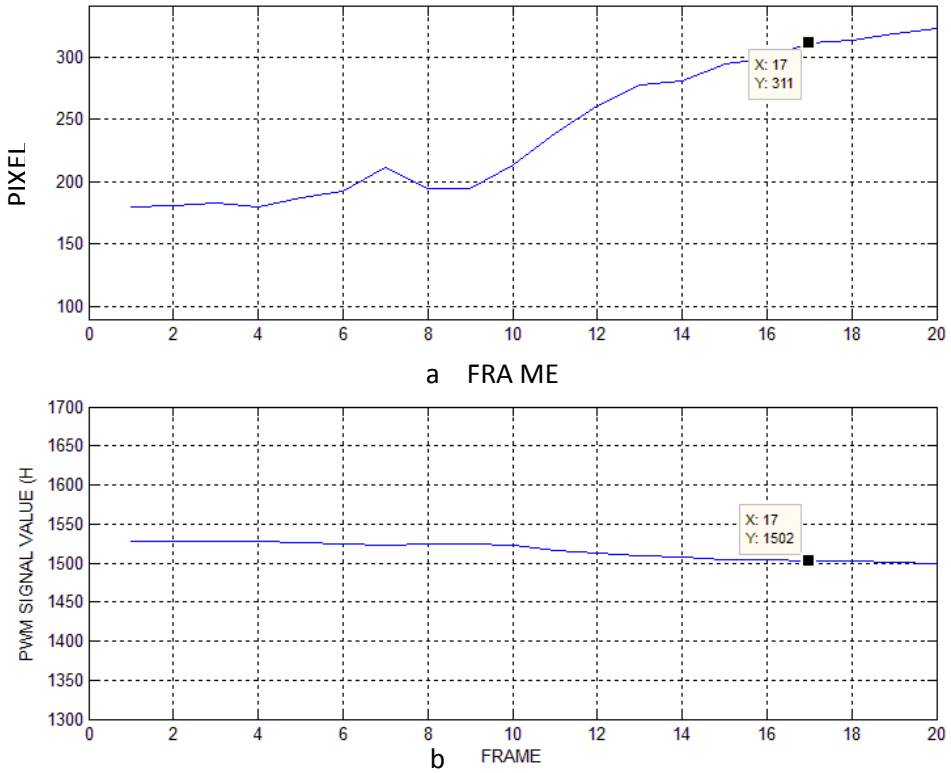


Figure 5.29: The Performance Of PID When Initial Position Is 180 Pixels

5.4.4 2DOF test

The Performance of system is tested when two motors work for both horizontal movement and vertical movement. As shown in Figure (5.30), the blue head of pen is tracked; the dark blue circle means estimated position of OBI.

Figure 5.30: Sample Frame of 2DOF test

OBI is moved with randomly directions to show if the robot eye can maintain OBI at the center of camera frame at point (240,320).

Figure 5.31: Robot Eye Maintain Approximately OBI At The Center Of Frame

Figure (5.31) shows sequence frames which proof that the robot eye can approximately centralize the OBI. If the features of motors and camera are better, the results will be better.

5.4.5 Summary of results:

Firstly, OBPF test proofs the following: PF_PSO is fast, but it is the weakest in tracking. Although PF_BFO is better than PF_PSO in tracking, it is the slowest. OBPF has the best result. It has the best time and the most robust tracking.

Secondly, AOPF test proofs the following: AOPF can hold tracking for object that has variable speed. Although OBPF can hold tracking for object that has rigid motion, it loses tracking for object that has variable speed.

Thirdly, FPID test proofs the following: The computational time required to maintain the object in the center is minimized by using FPID controller. Although FPID takes 14 frames to centralize OBI, PID takes 17 frames to centralize the same OBI. Besides, FPID makes system more stable. When initial speed of system is high, PID makes system unstable.

Finally, 2DOF test proofs that the robot eye can approximately centralize the OBI. If the features of motors and camera are better, the results will be better.

CHAPTER 6: CONCLUSIONS

In this thesis, the robot eye is designed using webcam and two servo motors. Arduino is used as driver for motors. Software of robot is implemented in visual C++ using Open CV tool and fuzzy control libarary programed by resacher. The robot aim to put OBI at the center of camera frame.

OPBF and AOPBF, unprecedented methods are provided in video object tracking. Comparsion among OBPF, PF_BFO and PF_PSO is made for tracking object in saved movie. OBPF outperfored other methods in term of accuracy, stability and speed. It solves both degeneracy phenomenon and impoverishment problem which PF has by improving predict step in PF. OBPF has two predect steps. First predict step is similalr to predict step in BPF. Second predict step depends on adaptive Chemostic step proposed in section 3.5.

However, OPBF has problem when it is used in real time. It can't hold tracking for object with variable speeds. To solve this problem, AOBPF is proposed. AOPF makes the length of first predict step adaptive to hold tracking object with variable speeds. Comparsin between OPBF and AOBPF for tracking video object in real time is made. The results show how AOPF is robust.

Because of nonlinearity of system, PID is not optimal contro . FPID control with 49 rules is desigened for robot. FPID is MIMO control. It has two inpust and two outps. Each motor has its output signal. Two motors have the same memberships and rules. Comparsion between FPID and PID is made for real video object tracking. The results show FPID made system faster and more stable.

However, the rise time of system is not good , because the bad performance of hacked low cost servo motors. Due to the seige imposed on Gaza strip by Israeli occupation, high performance components could not be imported.

Althogh the robot can hold traking and centralize the OBI, the accuracy of steady state is not high. Random movements of particle and large size of cross section of OBI made steady state to be area without fixed value. To solve this problem in future, two stage of tracking may be designed. First stage uses AOPBF to detect region of interesting and second stage uses another method to achieve fixed steady state.

REFERENCES

[1] H. M. Dee and S. A. Velastin, "How close are we to solving the problem of automated visual surveillance," Machine Vision and Applications, vol. 19, pp. 329-343, 2008.

[2] J. Klein, C. Lecomte, and P. Miche, "Preceding car tracking using belief functions and a particle filter," 19th International Conference on Pattern Recognition, ICPR 2008,pp. 1-4, 2008.

[3] L. Mejias, P. Campoy, S. Saripalli, and G. Sukhatme, "A visual servoing approach forn tracking features in urban areas using an autonomous helicopter," Proceedings 2006 IEEE International Conference on Robotics and Automation, ICRA 2006, pp. 2503-2508,2006.

[4] M. Li, B. Pang, Y. He, and F. Nian, "Particle Filter Improved by Genetic Algorithm and Particle Swarm Optimization Algorithm," Journal of Software, vol. 8, 2013.

[5] L. Ye, J. Wang, C. Li, H. Wang, and Q. Zhang, "A Novel Particle Filtering Framework Using Genetic Monte Carlo Sampling," International Conference onManagement and Service Science, pp. 1-4, 2009.

[6] G. Zhang, Y. Cheng, F. Yang, and Q. Pan, "Particle filter based on PSO," International Conference on Intelligent Computation Technology and Automation (ICICTA), pp. 121-124,2008.

[7] Mohammed Alhanjouri, Ahmed Alostaz "BFO vs. BSO for video object tracking using particle filter (PF)," Journal of Emerging Trends in Computing and Information Sciences, vol. 4, p. 8, 2013.

[8] Z. Xiaowei, L. Hong, and S. Xiaohong, "Object Tracking with an Evolutionary Particle Filter Based on Self-Adaptive Multi-Features Fusion," International Journal of Advanced Robotic Systems, vol. 10, 2013.

[9] Z. Hao, X. Zhang, H. Li, and J. Li, "Video object tracking based on swarm optimized particle filter," 2nd International Conference on Industrial Mechatronics and Automation (ICIMA), pp. 702-706,2010.

[10] Z. Hao, X. Zhang, P. Yu, and H. Li, "Video object tracking based on particle filter with ant colony optimization," 2nd International Conference on Advanced Computer Control (ICACC), pp. 232-236, 2010.

[11] K. jia Bai and W. Liu, "Improved object tracking with particle filter and mean shift," IEEE International Conference on Automation and Logistics, pp. 431-435,2007.

[12] A. Qadir, W. Semke, and J. Neubert, "Vision Based Neuro-Fuzzy Controller for a Two Axes Gimbal System with Small UAV," Journal of Intelligent & Robotic Systems, pp.1-19, 2013.

[13] M. El-Bardini, E. Elsheikh, and M. Fkirin, "Real Time Object Tracking Using Image Based Visual Servo Technique." International Journal of Computer Science & Emerging Technologies, Volume 2, pp.252-257, April 2011.

[14] O. Chang, P. Campoy, C. Martinez, and M. A. Olivares-Méndez, "A robotic eye controller based on cooperative neural agents," The 2010 International Joint Conference on Neural Networks (IJCNN), pp. 1-6,2010.

[15] J. Kennedy and R. Eberhart, "Particle swarm optimization," in Proceedings of IEEE international conference on neural networks, pp. 1942-1948,1995.

[16] C. W. Reynolds, "Flocks, herds and schools: A distributed behavioral model," in Proceedings of the 14th annual conference on Computer graphics and interactive techniques, pp. 25-34, 1987.

[17] S. S. Rao and S. Rao, *Engineering optimization: Theory and practice*, John Wiley & Sons, 2009.

[18] K. M. Passino, *Biomimicry for optimization, control, and automation*, Springer, 2005.

[19] E. S. Ali and S. Abd-Elazim, "Optimal PID tuning for load frequency control using bacteria foraging optimization algorithm," ACE, vol. 2, p. 1, 2010.

[20] B. Niu, H. Wang, L. Tan, and L. Li, "Improved BFO with adaptive chemotaxis step for global optimization," Seventh International Conference on Computational Intelligence and Security (CIS), pp. 76-80,2011.

[21] X. Xu, Y.-h. Liu, A.-m. Wang, G. Wang, and H.-l. Chen, "A new adaptive bacterial foraging optimizer based on field," EighthInternational Conference on Natural Computation (ICNC), pp. 986-990,2012.

[22] K. M. Bakwad, S. S. Pattnaik, B. Sohi, S. Devi, K. Panigrahi, S. Das, and M. Lohokare,"Hybrid Bacterial Foraging with parameter free PSO," World Congress on Nature & Biologically Inspired Computing, NaBIC 2009, pp. 1077-1081,2009.

[23] D. H. Kim, A. Abraham, and J. H. Cho, "A hybrid genetic algorithm and bacterialforaging approach for global optimization," Information Sciences, vol. 177, pp.3918-3937, 2007.

[24] Ahmed Alostaz, Mohammed Alhanjouri "A New Adaptive Bfo Based On Pso For Learning Neural Network," I-manager ’s Journal of Computer Science, vol. 1, p. 8, 2013.

[25] P. Aggarwal, Z. Syed, and N. El-Sheimy, *MEMS-based integrated navigation*, Artech House, 2010.

[26] J. V. Candy, "Bayesian signal processing," Classical, modern and particle filtering methods vol. 54, John Wiley & Sons, 2011.

[27] S. Särkkä, " Bayesian estimation of time-varying systems: discrete-time systems," unpublished|,2012, http://www.lce.hut.fi/~ssarkka/course_k2012/.

[28] Z. Chen, "Bayesian filtering: From Kalman filters to particle filters, and beyond," Statistics, vol. 182, pp. 1-69, 2003.

[29] K.-Y. Liu, S.-Q. Li, L. Tang, L. Wang, and W. Liu, "Fast face tracking using parallel particle filter algorithm," IEEE International Conference on Multimedia and Expo, 2009, ICME 2009, pp. 1302-1305,2009.

[30] J. Ma, C. Han, and Y. Chen, "Efficient visual tracking using particle filter," 10th International Conference on Information Fusion , pp. 1-6, 2007.

[31] R. Chen, Z. Zhang, H. Lu, H. Cui, and Y. Yan, "Particle-filter-based object tracking with color and texture information fusion," Sixth International Symposium on Multispectral Image Processing and Pattern Recognition, pp. 74952F-74952F-8, ICME 2009.

[32] X. Huang and L. Shi, "Simulation on a fuzzy-PID position controller of the CNC servo system," Sixth International Conference on Intelligent Systems Design and Applications, ISDA'06, pp. 305-309, 2006.

[33] Willow Garage and Itseez, "Welcome to OpenCV documentation!," http://docs.opencv.org/, access: July 6, 2014

[34] K. M. Passino, S. Yurkovich, and M. Reinfrank, *Fuzzy control* , Citeseer, 1998.