DIM POINT TARGET TRACKING IN INFRARED IMAGE SEQUENCES WITH LOW SNR

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ZAHİDE SELİN GÜLER

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICS AND ELECTRONICS ENGINEERING

FEBRUARY 2016

Approval of the thesis:

DIM POINT TARGET TRACKING IN INFRARED IMAGE SEQUENCES WITH LOW SNR

submitted by **ZAHİDE SELİN GÜLER** in partial fulfillment of the requirements for the degree of **Master of Science in Electrics and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Gülbin DURAL ÜNVER _____

Dean, **Graduate School of Natural and Applied Sciences**

Prof. Dr. Gönül TURHAN SAYAN _____

Head of Department, **Electrical and Electronics Eng.**

Assist. Prof. Elif VURAL _____

Supervisor, **Electrical and Electronics Eng.**

Examining Committee Members:

Prof. Dr. Aydın ALATAN _____

Electrical and Electronics Engineering Dept., METU

Assist. Prof. Elif VURAL _____

Electrical and Electronics Engineering Dept., METU

Prof. Dr. Çağatay CANDAN _____

Electrical and Electronics Engineering Dept., METU

Assoc. Prof. Umut ORGUNER _____

Electrical and Electronics Engineering Dept., METU

Assoc. Prof. Murat EFE _____

Electrical and Electronics Engineering Dept., Ankara University

Date: February 2nd, 2016

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name : ZAHİDE SELİN GÜLER

Signature :

ABSTRACT

DIM POINT TARGET TRACKING IN INFRARED IMAGE SEQUENCES WITH LOW SNR

Güler, Zahide Selin

M.S., Department of Electrical and Electronics Engineering

Supervisor: Assist. Prof. Elif Vural

February 2016, 90 pages

Dim point target tracking in infrared (IR) images has been an active research area especially in military fields. Due to the long distance from IR sensors, target appears as a dim moving point hidden in a heavily cluttered background and causes the Signal to Noise Ratio (SNR) to be very low. We present a tracking algorithm based on Particle Filters (PF), which estimates the target position by using both brightness level and motion model measurements. A target candidate list for the presented PF algorithm is generated by Top-Hat background subtraction and statistical information of frame sequences. An important advantage of the proposed algorithm is that it does not use any a priori knowledge of the target properties. The performance of the proposed algorithm is evaluated on different types of IR image sequences generated from different detector types, varying SNR levels and different target motions. The experimental results demonstrate that the algorithm can successfully track dim moving point target in low SNR environment and accurately estimate its trajectory.

Keywords: IR Image Sequences, Particle Filter, Dim Point Target Tracking, Low SNR, Top-Hat Morphological Filter, TBD.

ÖZ

DÜŞÜK SİNYAL-GÜRÜLTÜ ORANINA SAHİP KIZILÖTESİ GÖRÜNTÜ SEKANSLARINDA SÖNÜK NOKTA HEDEF İZLEME

Güler, Zahide Selin

Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü

Tez Yöneticisi: Yrd. Doç. Dr. Elif Vural

Şubat 2016, 90 sayfa

Kızılötesi görüntülerde sönük nokta hedef izleme, özellikle askeri sahada aktif bir araştırma alanı olarak yer almaktadır. Kızılötesi sensörlerden uzakta olmasından kaynaklı olarak, hedef, yoğun karışıklığa sahip arka plan içerisinde gizlenmiş bir hareket eden sönük nokta olarak belirmekte ve sinyal-gürültü oranının çok düşük olmasına sebep olmaktadır. Sunulan Parçacık Filtre Hedef İzleme algoritması hareket modeli ve parlaklık seviyesi ölçümlerini kullanarak hedef pozisyonu kestirimi sunmaktadır. PF algoritması için hedef aday listesi, Top-Hat arkaplan çıkarılması ve çerçeve sekansının istatistiksel bilgileri kullanılarak oluşturulmaktadır. Önerilen algoritmanın önemli bir avantajı ise hedef özelliklerine dair hiçbir önsel bilgi kullanmıyor olmasıdır. Çalışmada önerilen algoritmanın performansını incelemek için farklı dedektörlerden elde edilen farklı görüntü sekansları, değişken sinyal-gürültü seviyeleri ve farklı hedef hareketleri kullanılmıştır. Deney sonuçlarına göre algoritma, düşük gürültü-sinyal oranındaki bir çevredeki hareket eden sönük nokta hedefini başarılı bir şekilde takip edebilmekte ve hassas olarak yörüngesini kestirebilmektedir.

Anahtar Kelimeler: Kızılötesi Görüntü Sekansı, Parçacık Filtre, Sönük Nokta Hedef İzleme, Düşük Sinyal-Gürültü Oranı, Top-Hat Morfolojik Filtre, İzle-Bul.

To my mother,

ACKNOWLEDGEMENTS

First and foremost, I would like to express my deepest gratitude to my supervisor, Assist. Prof. Dr. Elif VURAL for her guidance and endless support throughout the process of this study. She was always available for my questions and gave generously of her time. She always knew where to look for the answers to my problems while leading me to the right theory and perspective. Without her enthusiasm, encouragement and faith in me, this work would not be completed.

I dedicate this thesis to my mother who has always been there for me. Her everlasting support and unconditional love have given me the strength to overcome obstacles throughout my life. I would also like to thank my father and my elder sister who have always been proud of me and my achievements. Without my family, I would not be who I am and where I am today.

I would like to offer my sincere thanks to my friends and colleagues for their support and trust. I would like to thank Gamze KARAMAN and Halilcan TOKSÖZ for their friendship and support, Kemal Arda ÖZERTEM for providing me the dataset and always being helpful to me, Doruk KÜÇÜKÇELEBİ for his kindness and making time to retest for me and also his effort to solve my problems. Finally special thanks to Ufuk IRMAK, who has always been there for me to solve my problems. He is the one with magic and luck when my code was not working.

I would like to express my sincere thanks to Ali Ünver SEÇEN. Without him I couldn't be able to finish my courses. He has always been helpful and supportive throughout my study. He has always more than a friend to me. I also would like to thank my special girl and new team member Hande JANE who has also been supportive and helpful, more like a sister to me.

Another special thanks to my best man Barış BEKMEZ who has given me advices to give up when I was complaining, but never refused me when I need his help. He has always been there for me. I'm grateful having him in my life.

For the most special one, I would like to offer my sincere thanks to Ümit ÖZTÜRK, for his patience, tolerance and support. He has always trusted in me and always been proud of me. He has always given me motivation and helped me to look on the bright side.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

DBT:	Detect Before Track
DP:	Dynamic Programming
HT:	Hough Transform
IR:	Infrared
MHT:	Multi-dimensional Hough Transform
MSE:	Mean Square Error
PDF	Probability Distribution Function
PF:	Particle Filtering
PSNR:	Peak Signal to Noise Ratio
RHT:	Randomized Hough Transform
SHT:	Standard Hough Transform
SIR:	Sampling Importance Resampling
SIS:	Sequential Importance Sampling
SNR:	Signal to Noise Ratio
SCNR:	Signal to Clutter plus Noise Ratio
TBD:	Track Before Detect

CHAPTER 1

INTRODUCTION

Object detection and tracking are important research topics of today. They find applications in both civilian and military fields such as surveillance, human identification, rescue operations, security services, diagnosis of diseases, robot navigation and guidance in weapon systems. Considering these application areas, object detection and tracking over a sequence of images could be thought of as an important and challenging task due to the complex background environments.

Object detection involves extracting the location of objects of interest in the frame of a video sequence. Object extraction is the most critical step and the type of the application area serves to generate its requirements but also specifies its limitations. The type of imaging wavelength, object detection range, size of the object and signal to noise ratio (SNR) or signal to clutter plus noise ratio (SCNR) are the most known limitations of object detection algorithms. Statistical models, background subtraction, optical flow, are some of the approaches used to deal with these obstacles.

Object tracking can be defined as the problem of estimating the trajectory of an object within image sequences as it moves between frames. In other words, a tracker assigns the target as the consistent parts of the image in different frames of a video [1]. It is known that object tracking needs both spatial and temporal information. Technological achievements like high powered computers, high quality video cameras and the increasing need for automated video analysis are the main reasons why object tracking has become this popular [2].

In literature, detection and tracking algorithms are categorized into two broad classes: detect-before-track (DBT) and track-before-detect (TBD). These two classes differ in

their manners of using the intensity and motion information. These two approaches will be detailed in Chapter 2.

1.1. DIM POINT TARGET TRACKING

“"Dim" will be taken to mean low contrast with the background” [3]. It is more difficult to detect and track dim point targets than brighter ones and distinguishing them from heavy clutters is a challenging task in infrared search and track systems [27].

Recently the detection and tracking of dim point moving targets in IR images in cluttered background has been an active research area in military applications, for example early warning systems, air and missile defense systems, etc. Fast and reliable automatic target detection and tracking is a very important task especially when the distance between the target and the imaging system is too high which causes the target to occupy several pixels or even one pixel in each frame of the image sequence. Due to its long distance from the IR sensors, the target appears as a dim moving point hidden in a heavily cluttered background in an IR image sequence. Thus its SNR is very low, which results in limited information for performing detection or tracking tasks [39]. “The target may also appear and disappear at unknown points in time” [4]. “As a result of small pixel size, poor contrast and lack of texture information of the target, it is difficult to make an effective distinction between the IR dim target and the background” [28].

In the situations where the Signal to Noise Ratio (SNR) is so low that the target cannot be detected in a single frame, the best way to enhance the SNR is to apply track-before-detect (TBD) techniques to determine the target information through image frames [5]. Many TBD algorithms have been proposed such as Hough transform (HT) [6], dynamic programming [7], [8] and the particle filtering method [9], [10]. These algorithms will be explained in detail in the following sections.

1.2. MOTIVATION

In order to leave the defense system enough time to respond, the target must be found as soon as possible and from a long distance. The need for detection and tracking from

long distance makes the tracking capability the key component of the whole defense system. Therefore, the study of low contrast small target detection and tracking algorithms is quite meaningful for increasing the attack distance and the response speed of the weapons. Because of the lack of prior information on the small target image such as shape, texture and type, the information on a small target available to the system is comparatively less and the target can be easily covered by clutter and noise when the SNR is low [11].

In order to overcome this problem, this thesis aims to develop and use a TBD method based on the Particle Filter, which is preferred due to its strong capability to handle low SNR. The aim of the study is to use this TBD algorithm in image sequences with different noisy backgrounds and different SNR values for the detection and tracking of dim point moving target. The proposed method takes the advantage of the TBD approach but also has the capability of eliminating unlikely target candidates according to the statistical information within frames. To gather the statistical information, algorithm utilizes some preprocessing steps on first few image frames. The aim of this approach is to detect all candidate points that can possibly be the true target. Then, the algorithm eliminates the initial candidates progressively, by considering their motion consistency to find the true target. Furthermore the proposed method processes the frames of the image sequence in a sequential manner. This ensures the causality of the algorithm, which potentially allows its usage in real-time tracking applications.

1.3. SCOPE AND OUTLINE OF THE THESIS

In this thesis a Particle Filter based method is presented for the tracking of a dim point moving target in low SNR.

Next in Chapter 2 a literature review about object detection and tracking algorithms is given. Theoretical details of the proposed algorithm are also discussed.

In Chapter 3 the explanation and implementation procedure of the proposed method is presented.

Chapter 4 focuses on the details of data generation process, algorithm implementation in MATLAB and simulation results of proposed algorithm.

Finally, the conclusions and future directions are mentioned in Chapter 5.

CHAPTER 2

DIM POINT TARGET DETECTION AND TRACKING ALGORITHMS

In this chapter, dim point target detection and target tracking will be covered in detail separately, and related algorithms will be introduced and discussed.

2.1. INTRODUCTION

Dim point target detection and tracking in IR images are currently attracting great interest in both civil and military applications.

In IR image sequences, the interested target appears as a dim point in cluttered background because of the distance between the target and the sensor, and has no feature information like shape, texture or target type. Furthermore, aerodynamical effects and disturbances decreases the SNR in a very low levels in real environments. “The only information for the detection and tracking of a dim target is its unknown intensity and velocity” [12].

2.2. LITERATURE REVIEW

Tracking algorithms can be classified as: detect-before-track (DBT) and track-before-detect (TBD). In DBT, target intensity is used first and target motion is used after the detection. On the other hand in TBD, target motion is used before the target intensity [12].

Standard DBT techniques detect the target at each measurement and then use these detections to estimate the trajectory of the target. DBT algorithms are adequate for applications where the targets are bright compared with the background. However, in many real cases, the target has an intensity below the detection threshold for many successive frames [34]. As a result, they perform poorly in long range surveillance where the target size is a few pixels or even a single pixel, and it can be easily lost in noise and clutter [4][39]. “DBT algorithms have two disadvantages:

1. They exhibit poor performance when the SNR is low,
2. Much of the information contained in the measurements is completely discarded due to the application of a detection threshold at each frame" [13].

TBD algorithms such as dynamic programing, Hough Transform, Particle Filter, etc. on the other hand, have fairly good advantages in the area of tracking dim point targets in low SNR environments. Instead of thresholding each frame, data is processed over a number of frames then the algorithm decides whether it is target or not.

TBD is specifically used in very low SNR scenarios. Latest research highlighted the performance of TBD is for low SNR more than DBT. But the performance of TBD algorithms can degrade in the presence of a velocity mismatch or a target maneuver [12].

A main challenge for any DBT and TBD algorithm is, the huge amount of data to be handled. This increases the computational complexity and the computation time. Therefore the realization and implementation of real-time tracking systems are either costly or they sacrifice on accuracy [4].

2.3. DETECT BEFORE TRACK

DBT algorithms adopt a "single-frame detection and multi-frame confirmation" strategy, and work well with image sequences of high SNR by focusing on the target's spatial character rather than the temporal character during the detection procedure. Compared with TBD, the detection based on DBT is faster, simpler and easier to implement in real-time, but may fail in the case of low SNR and the contrast between target and background. "*Figure 2.1* shows the flow chart of DBT algorithms" [14]. Various techniques for detection can be used such as Mathematical Morphologic Method, Statistical Modelling, Genetic Algorithms, Wavelet Transformation [43], Higher Order Correlation Method [44], Empirical Mode [44][28], etc.

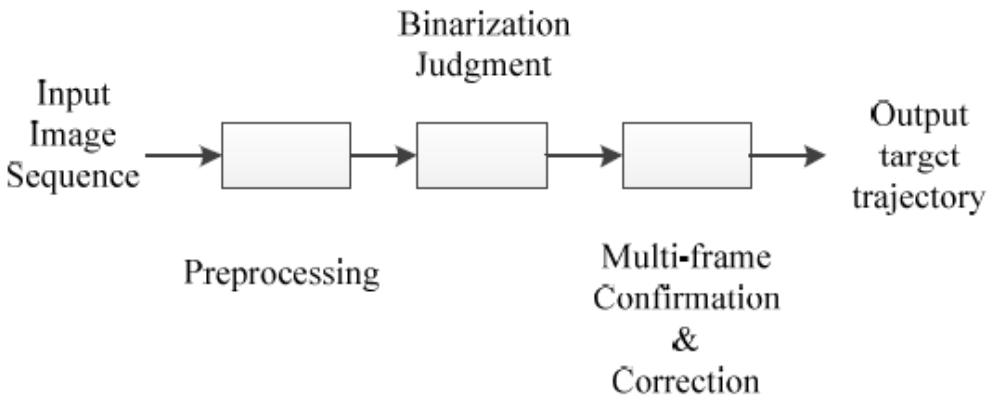


Figure 2.1 “DBT Algorithm Flow Chart” [14]

The critical issue of DBT algorithms is to pre-detect the target efficiently from a single frame. “The robustness of DBT algorithms is directly related to the characteristics of both small targets and backgrounds” [45]. In conclusion the key task of the target detection in cluttered background is distinguishing the real target region from the clutter background fast and efficiently [43].

2.3.1. PREPROCESSING

Preprocessing is a common name for low-level abstraction operations on images. The aim of these operations is to improve the image data in order to suppress unwanted distortions or to enhance image features which are important for further processing steps. Preprocessing operations can be pixel brightness transformations, geometric transformations, utilizing local neighborhood properties, etc. If some a priori knowledge is known, such as the nature of degradation, the nature of noise, and target properties (shape, size, etc.), preprocessing operations may be simplified considerably.

2.3.2. BINARIZATION JUDGEMENT

A brightness based threshold is chosen for detection; if the intensity value of a pixel is above this threshold it is probably the target, then then the pixel value is assigned as 1; and if the intensity value of a pixel is below this threshold, it is probably not target, then the pixel value is assigned as 0.

$$\begin{cases} c[x,y] \geq \text{Threshold}, & c[x,y] = \text{target} = 1 \\ c[x,y] \leq \text{Threshold}, & c[x,y] = \text{no target} = 0 \end{cases}$$

where [x, y] represents the pixel coordinates [25].

2.3.3. MULTI-FRAME CONFIRMATION AND CORRECTION

In the final step, because of not applying the appropriate threshold, there can be more than one trajectories. At this stage of the DBT algorithm the spatial information obtained from previous stages is compared with the temporal behavior of the target and the most consistent trajectory is determined.

2.4. TRACK BEFORE DETECT

Meanwhile, TBD algorithms adopt a "multi-frame detection" strategy to detect the target, by using both spatial and temporal information. The algorithm keeps tracking more than one candidate trajectories initialized in the detection process, and estimates a posterior probability for each one, which is compared with some kind of threshold at the end of the process. If one's posterior probability exceeds the threshold, it will be predicted as a target trajectory. TBD is preferred as an efficient and robust signal processing algorithm which uses extended observation time and reduces the probability of false alarm rate [46]. Previous studies suggest that TBD algorithms have a more complex structure and need more computation and storage than DBT algorithms, but are extremely effective in low SNR environments. *Figure 2.2* shows the flow chart of TBD algorithms [14].

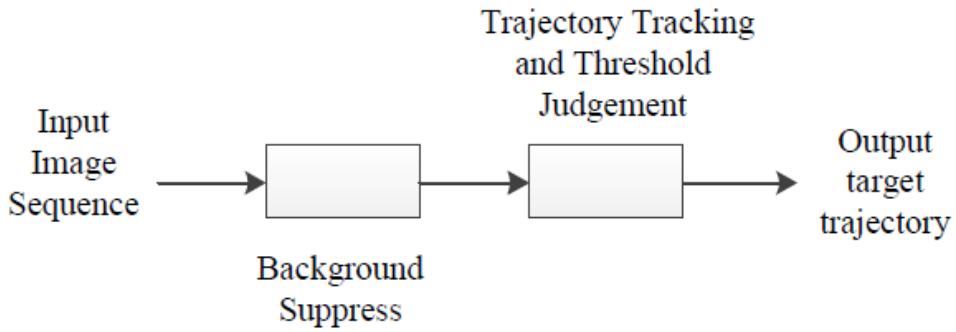


Figure 2.2 “TBD Algorithm Flow Chart” [14]

2.4.1. BACKGROUND SUPPRESSION

A background suppression algorithm, aims to estimate the background image distribution as accurately as possible, such that the residual information follows the Gaussian distribution. Then, it subtracts estimated background from the image sequence, while trying not to remove the information of the target from the sequence.

2.4.2. TRAJECTORY TRACKING AND THRESHOLD JUDGEMENT

In this stage of the algorithm, target candidates are tracked via the motion model representation of the target that is searched. Then a posterior probability based threshold is applied and the most probable trajectory is assigned as the target's true trajectory.

2.5. TRACK BEFORE DETECT ALGORITHMS

Many TBD algorithms have been proposed for dim point target tracking [26] like Probability Hypothesis Density [29], Viterbi Algorithm [32], Recursive Bayesian Algorithm [39], mean-shift algorithm [30], Probabilistic Data Association [35], non-linear filtering [42] etc. But the most famous TBD tactics involve Hough Transform (HT) [46], [41], Dynamic Programming (DP) [7], [8], [16] and Particle Filtering (PF) [31], [33], [36], [38]. These three algorithms are explained below.

The detection of straight lines in a clutter is the most common usage of HT in image processing. HT is generally applied on single images. The algorithm is based on representing the coordinates of points with a line equation defined by ρ and θ. The equation is given in *Figure 2.3.*

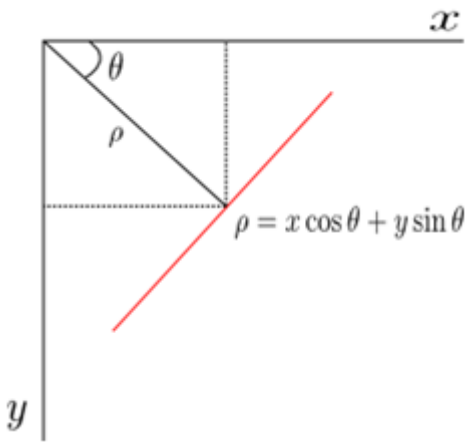


Figure 2.3 Hough Transform Equation for Representation of a Line

"HT has been widely used in image processing because of its robustness against noise. But HT has a long computation time and large memory requirements" [15]. The evolution of the Standard Hough transform (SHT), has led to some modified algorithms, such as randomized Hough transform (RHT) or Multi-dimensional Hough Transform (MHT) for tracking in clutter with higher probability of detection and lower computational cost. The HT uses the idea of defining a mutual constraint between image points and the parameters ρ and θ, based on basic line equation. HT can be described as a one to many mapping from a point on the image to a set of parameter values. In other words, HT calculates parameters of all straight lines, which belong to the set that represents a line which passes through a given image point (x,y).

Dynamic Programming is also a TBD algorithm which searches all the possible states, scans each pixel and marks probable tracks in each frame, and determines where it was in the beginning of its motion in the previous images. Each transition receives a score and the scores are related to target's intensity, velocity, direction and are given considering their surrounding and a priori restrictions [16].

Both DP and HT algorithms use score-based methods to detect the true target trajectory, that need to trace back and reconstruct the best path. Therefore, these algorithms cannot be applied if there is a requirement for on-line target tracking.

Particle Filter is another TBD technique which has the advantages of dealing with nonlinear/non-Gaussian problems. Recently, it has received more and more attention

as a track-before-detect algorithm [17]. “PF is based on Monte Carlo simulation and recursive Bayesian estimation” [18]. *Figure 2.4* shows how PF works in each frame.

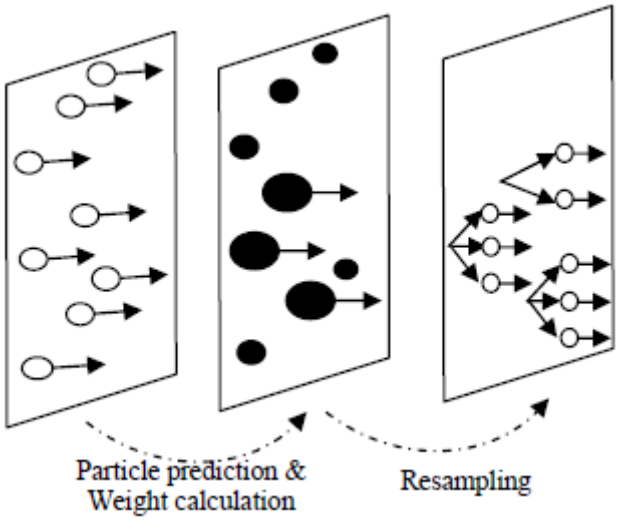


Figure 2.4 Particle Filter representation in each frame [40]

PF is at the core of the algorithm proposed in this thesis and is explained in detail in the following section.

2.6. PARTICLE FILTER ALGORITHM

In order to analyze a dynamic system, two models are known to be required: the system model which describes the change in the state with time and the measurement model which relates the noisy measurements to the state.

In the Bayesian model, to define the posterior probability density function (pdf) of the state one should gather all available information that also includes the set of measurements. An optimal estimate of the state and a measure of the estimation accuracy may be obtained from this posterior pdf. In many problems, an estimate is required every time that a measurement is received. For this case, a recursive filter has become a convenient solution.

A recursive filtering approach defines as an algorithm which processes received data sequentially rather than as a batch so that storing the complete data set is not a necessity if a new measurement becomes available [19].

"Recursive filters consist of two main stages: prediction and update. In the prediction stage, the algorithm uses the system model to predict the next state from the new measurement. The state prediction process is generally subjected to an unknown disturbances that results in a deformation and spread into the state pdf. The update stage, uses the latest measurement to modify and correct the prediction pdf. This is accomplished using Bayes theorem, which is the algorithm for updating the knowledge about the target state with using the information from new data" [19].

The state of a system evolves in time and information about the state is obtained from noisy measurements at each time step. In a discrete-time state-space model, the state of a system evolves according to:

$$x_k = f_k(x_{k-1}, v_{k-1}) \tag{2.1}$$

where $\boldsymbol{x_k}$ is a vector representing the state of the system at time $\boldsymbol{k}$, $\boldsymbol{v_{k-1}}$is the state noise vector, $\boldsymbol{f_k}$ is a non-linear and time-dependent function describing the evolution of the state vector.

Information about $\boldsymbol{x_k}$ is obtained only through noisy measurements of it, $\boldsymbol{z_k}$, which are governed by the equation

$$z_k = h_k(x_k, n_k) \tag{2.2}$$

where $\boldsymbol{h_k}$ is a possibly non-linear and time-dependent function describing the measurement process and $\boldsymbol{n_k}$ is the measurement noise vector.

The tracking problem can be defined as recursively calculating up to some degree of belief in the state $\boldsymbol{x_k}$ at time $\boldsymbol{k}$, given the data $\boldsymbol{z_{1:k}}$ up to time $\boldsymbol{k}$. The relation between $\boldsymbol{x_k}$ and $\boldsymbol{z_{1:k}}$ is captured through the pdf $\boldsymbol{p(x_k|z_{1:k})}$. It is assumed that the initial pdf $\boldsymbol{p(x_0|z_0) \equiv p(x_0)}$, which is also known as the prior distribution where $\boldsymbol{z_0}$ is the set of no measurements. Then, the pdf $\boldsymbol{p(x_k|z_{1:k})}$ may be obtained, recursively, in two stages: prediction and update[19].

Suppose that the required pdf $\boldsymbol{p(x_{k-1}|z_{1:k-1})}$ at time $\boldsymbol{k-1}$ is available. The prediction stage involves using the system model (2.1) to obtain the prior pdf of the state at time $\boldsymbol{k}$ via the Chapman–Kolmogorov equation [19]

$$p(x_k|z_{1:k-1}) = \int p(x_k|x_{k-1})\, p(x_{k-1}|z_{1:k-1}) dx_{k-1} \tag{2.3}$$

where $\boldsymbol{p(x_k|x_{k-1}, z_{1:k-1}) = p(x_k|x_{k-1})}$ as (2.1) describes a Markov process of order one[19].

At time step $\boldsymbol{k}$, a measurement $\boldsymbol{z_k}$ becomes available, and this may be used to update the prior (update stage) via Bayes' rule [19]

$$p(x_k|z_{1:k}) = \frac{p(z_k|x_k)p(x_k|z_{1:k-1})}{p(z_k|z_{1:k-1})} \tag{2.4}$$

where

$$p(z_k|z_{1:k-1}) = \int p(z_k|x_k)\, p(x_k|z_{1:k-1}) dx_k \tag{2.5}$$

depends on the likelihood function $\boldsymbol{p(z_k|x_k)}$ defined by the measurement model (2.2) and the known statistics of $\boldsymbol{n_k}$. In the update stage (2.4), in order to modify the prior density, the measurement z_k is used to obtain the required posterior density of the current state[19].

The relation between (2.3) and (2.4) form the basis for the optimal Bayesian solution. This recursive propagation of the posterior density represents only a conceptual solution, unfortunately cannot be determined analytically. Solutions do exist in a restrictive set of cases which will be detailed[19].

2.6.1. SEQUENTIAL IMPORTANCE SAMPLING (SIS)

Sequential importance sampling (SIS) is the most basic Monte Carlo method used for when the prediction and update steps cannot be analytically computed. In deriving the SIS algorithm, it is useful to consider the full posterior distribution at time $\boldsymbol{k}$, $\boldsymbol{p(x_{0:k}|z_{1:k})}$, rather than the filtering distribution, $\boldsymbol{p(x_k|z_{1:k})}$ which is just the

marginal of the full posterior distribution with respect to $\boldsymbol{x_k}$. "The idea in SIS algorithm is to approximate the posterior distribution at time, $\boldsymbol{p(x_{0:k}|z_{1:k})}$ **,** with a weighted set of samples $\{\boldsymbol{x_{0:k}^i}, \boldsymbol{\omega_k^i}\}_{\boldsymbol{i=1}}^{\boldsymbol{N}}$, also called particles. The weights are normalized as $\sum_{\boldsymbol{i}} \boldsymbol{\omega_k^i} = \mathbf{1}$. The particles are recursively updated to obtain an approximation to the posterior distribution as" [19]

$$p(x_{0:k}|z_{1:k}) \approx \sum_{i=1}^{N_s} \omega_k^i \delta(x_{0:k} - x_{0:k}^i) \tag{2.6}$$

Another property of the SIS algorithm is its dependence on importance sampling. In importance sampling, target distribution $\boldsymbol{p(x)}$ is approximated with using samples from a proposal distribution $\boldsymbol{q(x)}$. It is generally used when there is a difficulty in sampling directly from the target distribution itself. It is accepted that sampling from the proposal distributions is more convenient. To compensate for the discrepancy between the target and proposal distributions, one has to weight each sample $\boldsymbol{x^i}$ by [19]

$$\omega_i \propto \pi(x^i)/q(x^i) \tag{2.7}$$

where $\boldsymbol{\pi(x)}$ is a function that is proportional to $\boldsymbol{p(x)}$ (i.e. $\boldsymbol{p(x) \propto \pi(x)}$) and that we know how to evaluate. Then, a weighted approximation to the density $\boldsymbol{p(x)}$ is given by [19]

$$p(x) \approx \sum_{i=1}^{N_s} \omega^i \delta(x - x^i) \tag{2.8}$$

Therefore, if the samples $\boldsymbol{x_{0:k}^i}$ are drawn from an importance density $\boldsymbol{q(x_{0:k}^i|z_{1:k})}$**,** then the weights in (2.6) are defined by (2.7) to be [19]

$$\omega_k^i \propto \frac{p(x_{0:k}^i|z_{1:k})}{q(x_{0:k}^i|z_{1:k})} \tag{2.9}$$

Returning to the sequential case, at each iteration, one could have samples constituting an approximation to $\boldsymbol{p(x_{0:k-1}|z_{1:k-1})}$ and want to approximate $\boldsymbol{p(x_{0:k}|z_{1:k})}$ with a new set of samples. If the importance density is chosen to factorize such that [19]

$$q(x_{0:k}|z_{1:k}) = q(x_k|x_{0:k-1}, z_{1:k})q(x_{0:k-1}|z_{1:k-1}) \tag{2.10}$$

then one can obtain samples $\boldsymbol{x^i_{0:k}} \sim \boldsymbol{q(x_{0:k}|z_{1:k})}$ by evolving each of the existing samples $\boldsymbol{x^i_{0:k-1}} \sim \boldsymbol{q(x_{0:k-1}|z_{1:k-1})}$ with the new state $\boldsymbol{x^i_k} \sim \boldsymbol{q(x_k|x_{0:k-1}, z_{1:k})}$**.** [19]

The weight update equation can be shown to be

$$\begin{aligned} \omega_k^i &\propto \frac{p(z_k|x_k^i)\, p(x_k^i|x_{k-1}^i) p(x_{0:k-1}^i|z_{1:k-1})}{q(x_k^i|x_{0:k-1}^i, z_{1:k}) q(x_{0:k-1}^i|z_{1:k-1})} \\ &= \omega_{k-1}^i \frac{p(z_k|x_k^i)\, p(x_k^i|x_{k-1}^i)}{q(x_k^i|x_{0:k-1}^i, z_{1:k})} \end{aligned} \tag{2.11}$$

Furthermore, if $\boldsymbol{q(x_k|x_{0:k-1}, z_{1:k}) = q(x_k|x_{k-1}, z_k)}$, then the importance density becomes only dependent on $\boldsymbol{x_{k-1}}$ and $\boldsymbol{z_k}$. This is useful when only a filtered estimate of $\boldsymbol{p(x_k|z_{1:k})}$ is required at each time step. In such scenarios, only $\boldsymbol{x^i_k}$ need to be stored; therefore, one can discard the path $\boldsymbol{x^i_{0:k-1}}$ and the history of observations $\boldsymbol{z_{1:k-1}}$**.** The modified weight is then[19]

$$\omega_k^i \propto \omega_{k-1}^i \frac{p(z_k|x_k^i)\, p(x_k^i|x_{k-1}^i)}{q(x_k^i|x_{k-1}^i, z_k)} \tag{2.12}$$

and the posterior filtered density $\boldsymbol{p(x_k|z_{1:k})}$ can be approximated as

$$p(x_k|z_{1:k}) \approx \sum_{i=1}^{N_s} \omega_k^i \delta(x_k - x_k^i) \tag{2.13}$$

where the weights are defined in (2.12). It can be shown that as $\boldsymbol{N_s} \to \infty$, the approximation (2.13) approaches the true posterior density $\boldsymbol{p(x_k|z_{1:k})}$[19].

The SIS algorithm thus consists of recursive propagation of the weights and support points as each measurement is received sequentially. A pseudo-code description of this algorithm is given by **Algorithm 1**.

Algorithm 1 Pseudo-code for SIS Particle Filter

for *i=1:Ns*

$x_k^i \sim q(x_k^i | x_{k-1}^i, z_k)$

Assign the particle a weight, ω_k^i*, according to (2.12)*

end for

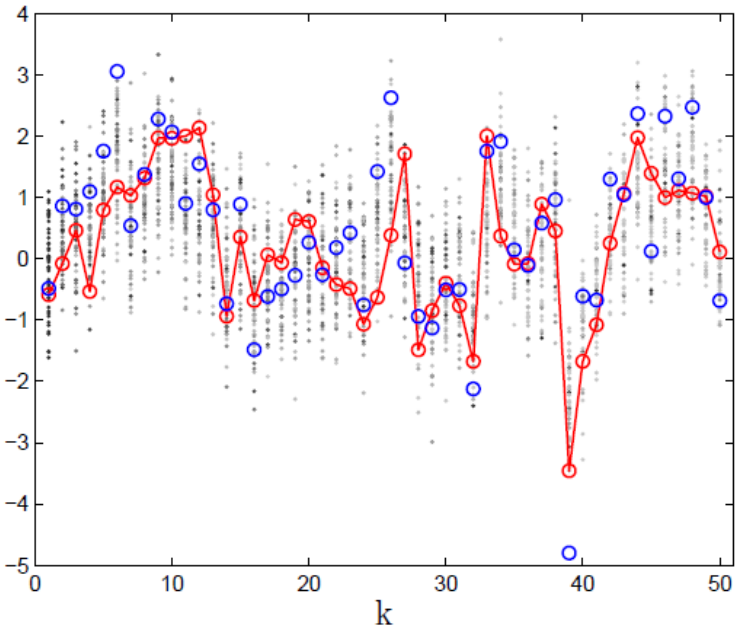


Figure 2.5 Demonstration of the SIS filtering algorithm on a linear Gaussian model [20]

In *Figure 2.5* the red line shows the state, $\boldsymbol{x_k}$, of the system at each time step $\boldsymbol{k}$; the blue circles show the measurements, $\boldsymbol{z_k}$, at the corresponding time steps. The gray dots represent the particles generated by the SIS algorithm. Darker colors correspond to a larger weight for the particle. In this example, N = 50 particles is used [20].

2.6.1.1. DEGENERACY PROBLEM

The iterations of **Algorithm 1** leads to a degeneracy problem where only a few of the particles have a significant weight, and all the others have very small weights. Degeneracy is typically measured by an estimate of the effective sample size:

$$N_{eff} = \frac{N_s}{1 + var(\omega_k^{*i})} \tag{2.14}$$

where $\boldsymbol{\omega_k^{*i} = p(x_k^i|z_{1:k})/q(x_k^i|x_{k-1}^i, z_k)}$ is referred to as the "true weight". A smaller $\boldsymbol{N_{eff}}$ means a larger variance for the weights, hence more degeneracy. (2.14) cannot be evaluated exactly, but an estimate $\widehat{\boldsymbol{N_{eff}}}$ of $\boldsymbol{N_{eff}}$ can be obtained by

$$\widehat{N_{eff}} = \frac{1}{\sum_{i=1}^{N_s}(\omega_k^i)^2} \tag{2.15}$$

where $\boldsymbol{\omega_k^i}$ is the normalized weight obtained using (2.11).

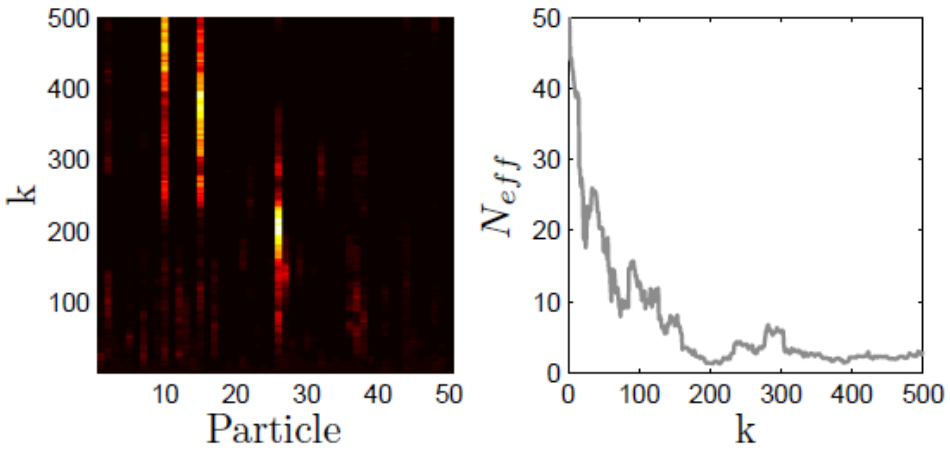


Figure 2.6 Particle weights and Neff change with respect to k

"The weights of all 50 particles (x-axis) at each time step k (y-axis) (in the left).In *Figure 2.6* brighter colors represent larger weights. The effective sample size called N_{eff} is shown as a function of time step k (in the right)" [20].

As one can see from *Figure 2.6* as $\boldsymbol{k}$ increases, $\boldsymbol{N_{eff}}$ drops very quickly to values smaller than 5. Thus for large $\boldsymbol{k}$, only a few of the particles have significant weights.

2.6.1.2. RESAMPLING

A possible solution to deal with the degeneracy problem is resampling. The idea of resampling is to eliminate particles that have small weights and to concentrate on particles with large weights. The resampling step involves generating a new set

$\{x_k^{*i}\}_{i=1}^{N_s}$ by resampling, N_s, times from an approximate discrete representation of $p(x_k|z_{1:k})$ given by [19]

$$p(x_k|z_{1:k}) \approx \sum_{i=1}^{N_s} \omega_k^i \delta(x_k - x_k^i) \tag{2.16}$$

so that $\mathbf{Pr}(x_k^{i*} = x_k^j) = \omega_k^j$. The resulting sample is in fact an i.i.d. sample from the discrete density (2.16); therefore, the weights are now reset to $\omega_k^i = 1/N_s$. Pseudo-code descriptions of resampling and Particle Filter algorithms are given respectively in **Algorithm 2** and **Algorithm 3**.

2.6.2. SAMPLING IMPORTANCE RESAMPLING (SIR)

Particle filtering algorithms mostly describe a variant of the SIS algorithm mentioned above. The SIR algorithm also can be seen as a variant of SIS. But in this case the proposal distribution $q(x_k|x_{k-1}^i, z_k)$ is taken to be the state transition distribution $p(x_k|x_{k-1}^i)$ and resampling is applied at every iteration. Thus, in the SIR algorithm, the update equations for the particles become [19]

$$x_k^i \sim p(x_k|x_{k-1}^i) \tag{2.17}$$

$$\omega_k^i \propto p(z_k|x_k^i) \tag{2.18}$$

The resampling step follows the update equations. Note that the ω_{k-1}^i term disappears in the weight update equation in (2.18) because after resampling at time ***k-1,*** all weights ω_{k-1}^i become equal. [19]

Algorithm 2 Pseudo-code for Resampling

Initialize the CDF $c_1=0$

for *i=2:Ns*

Construct CDF: $c_i = c_{i-1} + \omega_k^i$

end for

Start CDF: i=1

Draw a starting point: $u_1 \sim U[0, N_s^{-1}]$

for *j=1:Ns*

move along the CDF: $u_j = u_1 + N_s^{-1}(j-1)$

while $u_j > c_i$

$i=i+1$

end while

assign sample: $x_k^{j*} = x_k^i$

assign weight: $\omega_k^j = N_s^{-1}$

assign parent: $i^j = i$

end for

Algorithm 3 Pseudo-code for Generic Particle Filter

for *i=1:Ns*

$x_k^i \sim q(x_k^i | x_{k-1}^i, z_k)$

Assign the particle a weight, ω_k^i*, according to (2.12)*

end for

Calculate total weight: $t = SUM\left[\{\omega_k^i\}_{i=1}^{N_s}\right]$

for *i=1:Ns*

normalize: $\omega_k^i = t^{-1}\omega_k^i$

end for

If $\widehat{N_{eff}} < N_T$

Resample using ***Algorithm 2***

end if

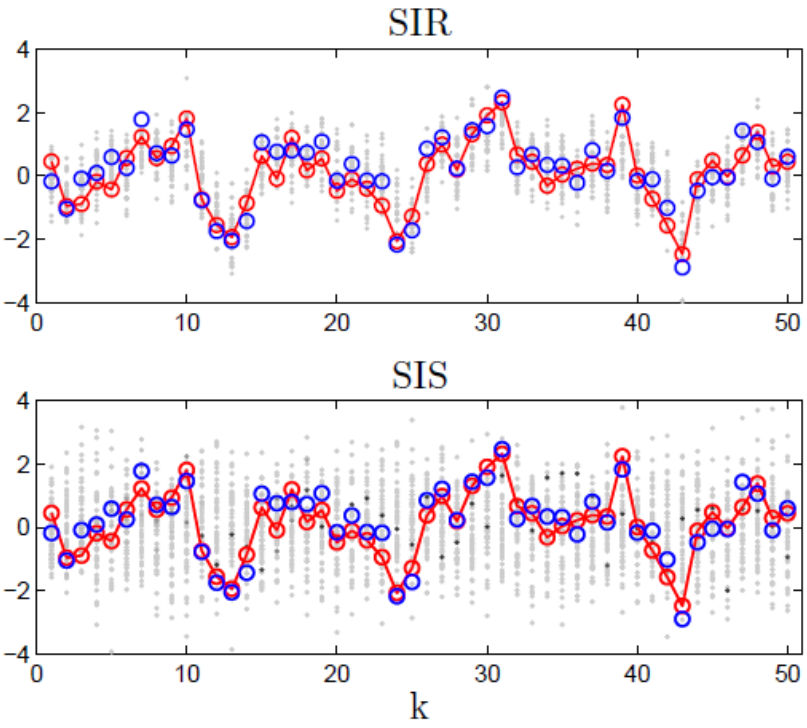


Figure 2.7 “Comparison of SIR and SIS algorithms” [20].

In *Figure 2.7* an SIR filter is applied to a linear Gaussian system. The red line shows the state, $\boldsymbol{x_k}$, of the system at each time step $\boldsymbol{k}$; the blue circles show the measurements, $\boldsymbol{z_k}$, at each time step. The gray dots represent the particles generated by the SIR algorithm. An SIS filter is applied to the same system. The SIS filter uses the same proposal distribution as the SIR filter, but does not involve resampling step. In this example, there are N = 50 particles in both cases [20].

The main advantage of the SIR algorithm is that it is extremely simple to implement, since it only requires sampling from the distribution $\boldsymbol{p(x_k|x_{k-1}^i)}$ and evaluating $\boldsymbol{p(z_k|x_k^i)}$. Its disadvantage is the independence of the proposal distribution $\boldsymbol{p(x_k|x_{k-1}^i)}$ from the observations $\boldsymbol{z_k}$ (this means that the states are updated without directly taking into account the information provided by the observations).

In order to overcome this, a motion model is used for the state update equation. The motion model is estimated from the measurements and the motional behavior of the target in a feedback loop. The motion model assumption and its usage in the algorithm will be detailed in Chapter 3.

CHAPTER 3

PROPOSED ALGORITHM

As mentioned in Chapter 1, the aim of the thesis is to track dim targets in low SNR. The main properties of the proposed algorithm can be listed below.

- A preprocessing step is applied for eliminating the points which are below some threshold that is adaptively calculated considering the statistical properties of the image frames.
- A separate PF algorithm is initiated to track each one of the candidate points that stay above the threshold after the preprocessing step. Initiating the PF algorithm with a list of candidate points rather than searching for the whole image sequence reduce the computational complexity.
- The PF algorithm generates particles for each candidate between some predefined windows of frames and tracks them separately.
- Candidate points are eliminated with respect to an error measure based on motion consistency, which also reduces the computational complexity.
- The PF algorithm is also fed with the estimates of the target candidates measure calculated from the current trajectory estimates. The velocity information is used in addition to the intensity information in order to define a hybrid weighting strategy to update particle weights. This provides robustness when the SNR is very low and intensity measurements cannot track the target.

In *Figure 3.1* a flow chart of the proposed algorithm is given.

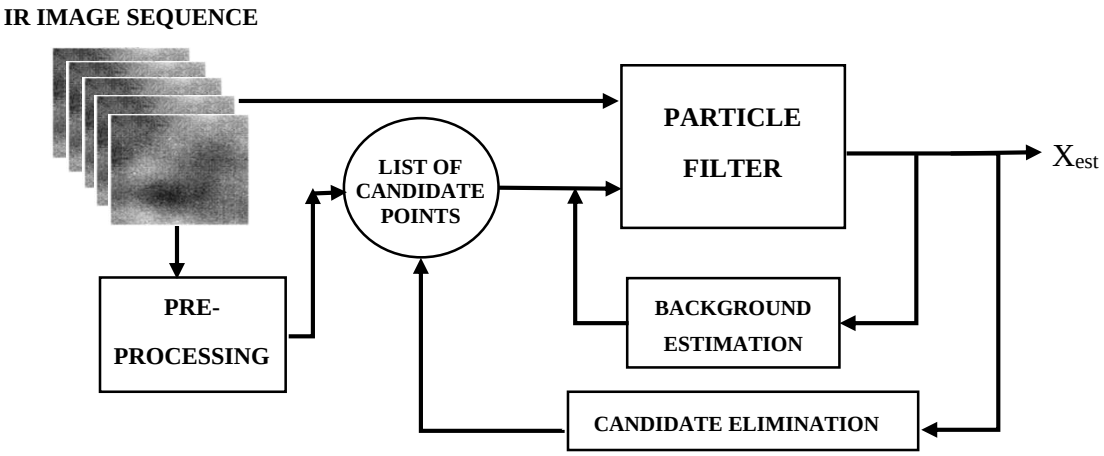


Figure 3.1 Proposed Algorithm Flow Chart

In order to test the performance of the proposed algorithm, different SNR levels and different data types are used. The image sequence generation process and the details of the algorithm are given in this chapter.

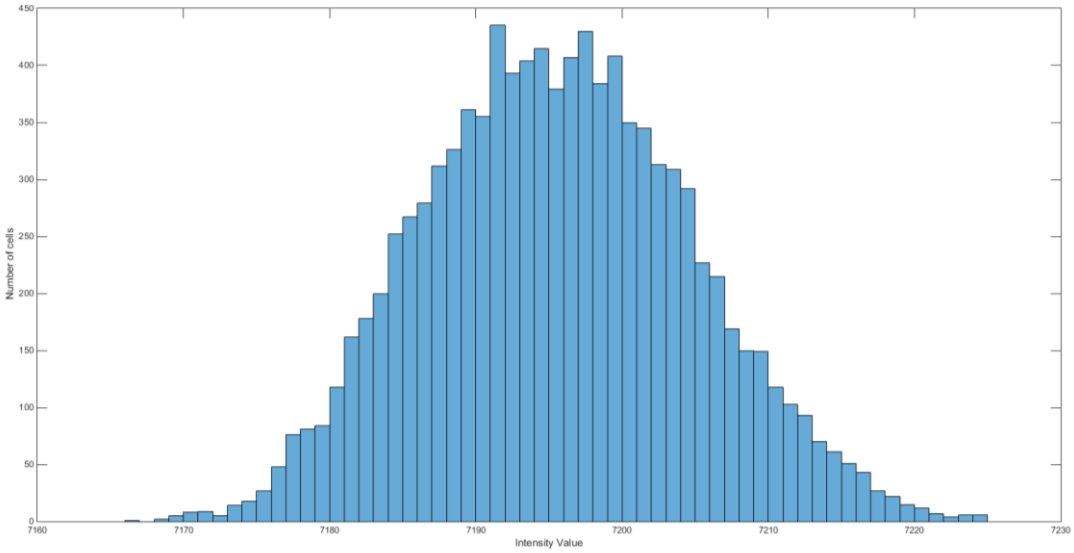
3.1. IMAGE SEQUENCE GENERATION

The need for dim target tracking occurs in a scenario where a target must be tracked within a long distance range and under low SNR conditions. In order to realize this scenario a series of image sequences are generated including different intensity values for target representation and different types of sensor recorded video sequences.

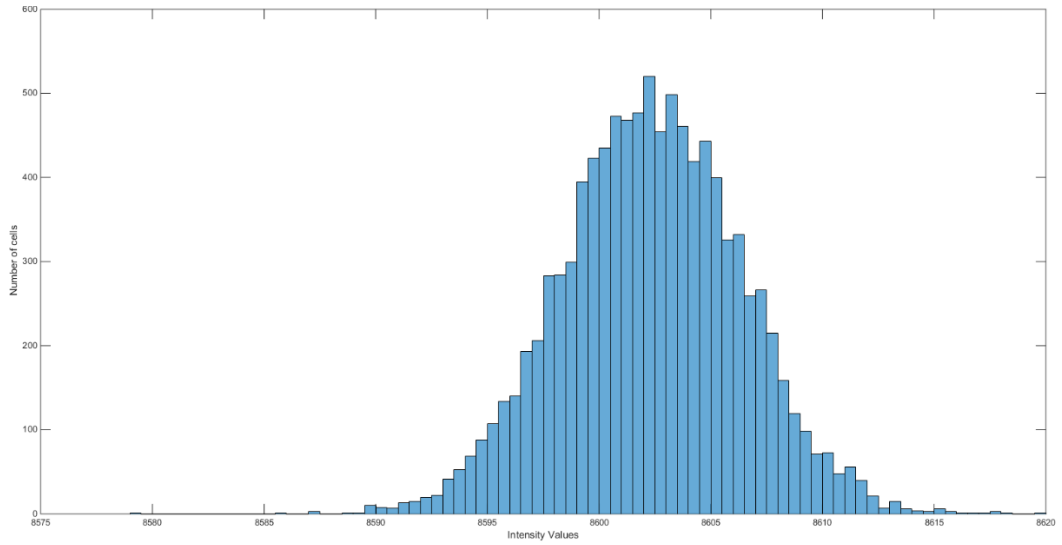
Two detector data (SENSOR TYPE-1 and SENSOR TYPE-2) are used for generating the dataset. The number of frames in the image sequences change but they are all sampled at 100 Hz. Each frame has the size of 100x100 pixels. A chosen frame of each data is given in *Figure 3.3*.

A representative target model is integrated into the image sequences with different intensity values. In order to represent the target in each image sequence, the mean value of each frame in each image sequence is calculated and a chosen value between 10 and 100 is added into the frame in a specific coordinate which is determined from the motion model of the target. The reason for using a range of different target intensity values is to compare performance of the algorithm in different SNR conditions.

In order to visualize the intensity distribution of image sequences, histograms of a representative frame for each dataset are shown in *Figure 3.2.* As one can see, in the first frame of each dataset; in SENSOR TYPE-1, the intensity value changes between 7166 and 7225, and the mean value is 7195; while in SENSOR TYPE-2, the intensity value changes between 8579 and 8619, and the mean value is 8602. The values of pixel intensities depend on the detector image stream type. In this study both SENSOR TYPE-1 and SENSOR TYPE-2 have 14-bit streams for images.



a. Histogram of SENSOR TYPE-1 image in 1st frame



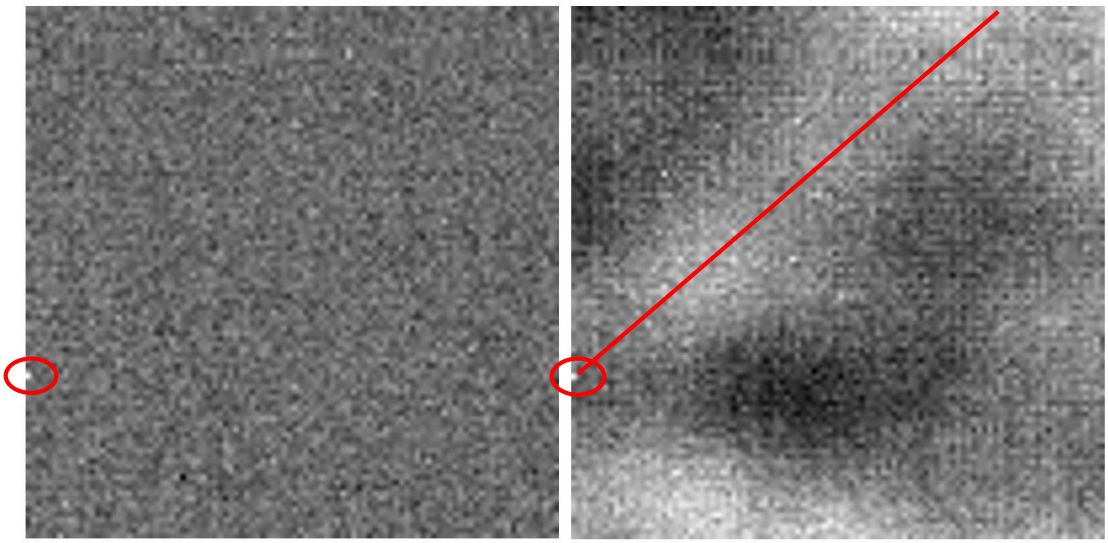
b. Histogram of SENSOR TYPE-2 image in 1st frame

Figure 3.2 Intensity Distributions of Image Sequences in 10th frame

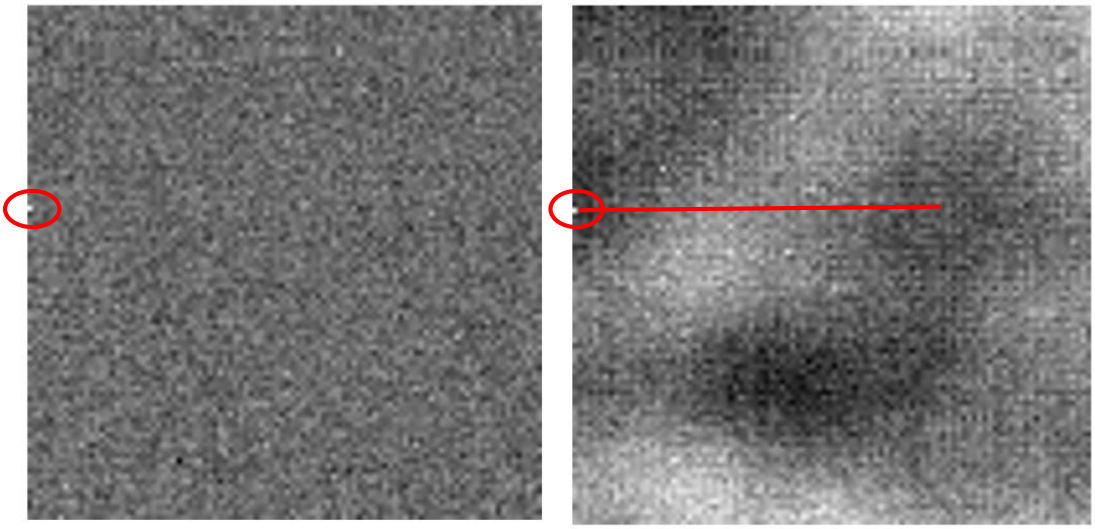
Two different motions are generated; the first one represents a target motion that enters the scene on the left side of the frame and moves horizontally, the second one enters the scene on the left corner and moves in a diagonal direction. The second motion model is chosen to generate a challenging data considering the pattern of the SENSOR TYPE-1 image sequence. In both motion models target enters in the first frame.

In *Figure 3.3* the first frames of each dataset where the target intensity is 30 above the mean intensity, is presented as an example for each image sequence pattern. The bright pixels in red circles represent the target which enters the scene in the coordinates [70,1] (a and b) and [40, 1] (c and d). Red lines represents the trajectory of target's movement in sequential frames.

From this point on, the motion of the target where it enters the scene in [70,1] is called "Motion 1" and the other motion which starts from the point [40,1] is called "Motion 2".



a. SENSOR TYPE-2 b. SENSOR TYPE-1



c. SENSOR TYPE-2 d. SENSOR TYPE-1

Figure 3.3 First frame of image sequences SENSOR TYPE-2 (a and c) SENSOR TYPE-1 (b and d)

3.2. OBSERVATION MODEL OF IR IMAGE SEQUENCE

The target will only occupy one or several pixels in the imaging plane when it is far away from the IR imaging system, although its actual diameter may be more than ten meters. In such situations, the noise amplitude is often quite similar to the intensity of dim targets and the targets are totally mixed up with noise in IR images [14].

In general, the observation model of IR image sequences can be described as follows:

$$I(x,y,k) = \begin{cases} B(x,y,k) + T(x,y,k) + N(x,y,k) & with\ target \\ B(x,y,k) + N(x,y,k) & without\ target \end{cases} \quad (3.1)$$

where $I(x,y,k)$ represents the overall image intensity, $B(x,y,k)$ is the intensity of background clutter, $N(x,y,k)$ is the intensity of noise and $T(x,y,k)$ is the target intensity. x and y represents the spatial coordinates and k represents the temporal coordinate of frames [4].

3.3. TARGET STATE TRANSITION MODEL

The dim target is usually considered as a point in a single IR image. The target motion model is expressed as follows

$$x_k = A\,x_{k-1} + B_k + C\,n_k \qquad k=1,2,...K \quad (3.2)$$

where $x_k = [x\ \dot{x}\ y\ \dot{y}]^T$, x is the target's coordinate in x-axis at time k, $\dot{x}$ is the target's speed along the x-axis, y is the target's coordinate in y-axis at time k, $\dot{y}$ is the target's speed along the y-axis. n_k represents the process noise. A is the state transition matrix

$$A = \begin{bmatrix} 1 & \partial T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \partial T \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

∂T is the time interval between consecutive frames. B_k is the vector $[\bar{x}\bar{\dot{x}}\ \bar{y}\bar{\dot{y}}]^T$ which represents the expected change in the displacement and speed.

The model described in (3.2) is in the form of general state transition model. In this study for the simplicity of the model, velocity indices are not included in the state model. The state model thus consists only of the target coordinates, nevertheless, thanks to the B_k parameter, the velocity information can still be captured in the model. The B_k vector is calculated from the estimated displacement over a predefined number of frames and updated regularly during the motion. This vector used for directing particles is thus computed based on the past motion information and will be detailed in Chapter 4.

3.4. PROPOSED ALGORTIHM AND IMPLEMENTATION PROCESS

It has been discussed in Chapter 2 that in low SNR tracking problems, TBD algorithms offer the best solutions for dim target tracking. Regarding its performance in nonlinear, non-Gaussian environments [17] the Particle Filter is chosen to solve the problem studied in this thesis.

The PF algorithm examines each pixel as a candidate. For datasets where the target intensity is relatively high compared to the background, a simple thresholding is enough to find the real target and there is no need to eliminate any target candidates. But in the case of lower intensities the threshold must be set sufficiently low to detect the real target, however, this results in many other candidates as well. It is assumed that the target appears in the scene from the first frame. The proposed algorithm detects the target candidates in the initial frames and reduces the number of candidate points,

which are given as input to the PF algorithm, gradually throughout the frames of the image sequence.

The applied algorithm will be detailed in the following sections. In the first part, background subtraction, morphological operations and their implementation processes are given. These algorithms serve to detect candidate points for the PF algorithm and are called as preprocessing operations. In the second part, tracking the candidate points with the PF algorithm is explained.

3.4.1. PREPROCESSING OPERATIONS FOR CANDIDATE DETECTION

A pre-processing method is adopted because of the low SNR and low contrast features in a single frame [37]. The purpose of the preprocessing algorithm is to gather statistical information about the image sequence in the first few frames and suppress noise and background regarding this information. After the thresholding and binarization steps candidate points will be generated. The aim of this step is to only reduce the number of candidates, not to change the properties of the input sequences to avoid any information loss. For each of these candidate points, the PF runs on the original input image sequences.

3.4.1.1. ELIMINATING THE NON-LINEARITY OF TEMPERATURE DISTRIBUTION

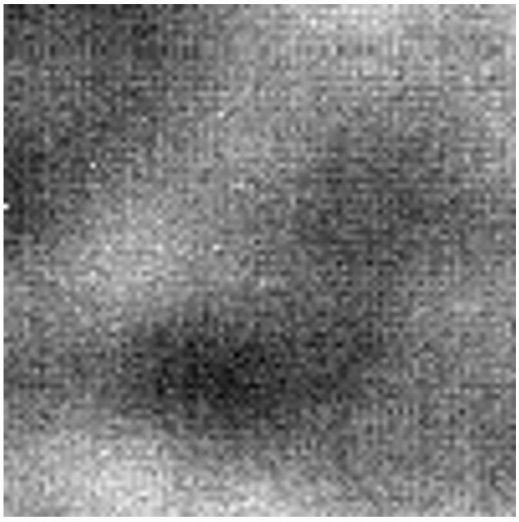
Detector image sequences are generated by recording while the detector is fixed in front of a wall, being exposed to temperature changes. According to Zhang. F et. al. the pixels in the same row of the image correspond to the same atmosphere temperature approximately. The intensity mean of *i*th row is given by [12]

$$\bar{u}_i = \sum_{j=1}^{n} u_{ij}/n \tag{3.4}$$

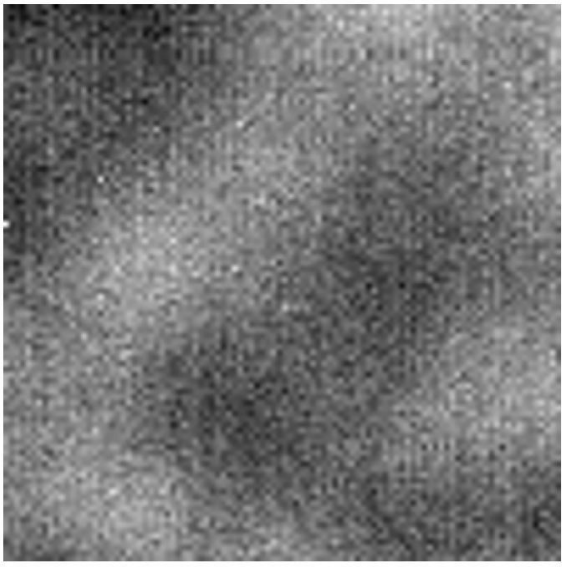
where $\bar{u}_i$ denotes the mean of the *i*th row, u_{ij} denotes intensity of the pixel lying on the *i*th row and the *j*th column, *n* is the number of pixels in one row. The non-linear

distribution of the temperature and the background are effectively suppressed by subtracting the mean intensity of its row from each pixel [12]

$$\tilde{u}_{ij} = u_{ij} - \bar{u}_i \tag{3.5}$$



(a) Before eliminating temperature distribution



(b) After eliminating temperature distribution

Figure 3.4 Temperature Distribution Elimination for SENSOR TYPE-1

Figure 3.4 shows the effect of temperature distribution elimination process. One can see that the intensity level of bright pixels is reduced.

3.4.1.2. TOP-HAT MORPHOLOGICAL FILTER

Morphological operations are commonly used in tiny target detection in IR images. The development of morphological research offers good results in dim target detection and thus morphological image processing has become a new trend in IR image processing [22].

The Top-hat operator in mathematical morphology can find the mass of bright pixels. After eliminating the temperature non-linearity, the Top-hat operator is utilized to reject background while preserving the target and enhancing the SNR of the images [12].

The morphological filter theory includes some basic operations. Given the input image $F = \{x, f(x) | x \in P, P \subseteq E^2\}$ and the structuring elements

$B = \{m, b(m) | m \in S, S \subseteq E^2\}$ where E^2 represents Euclidian space, P and S represent the sets of input and structuring elements, the dilation operation of F and B is defined by [22]

$$(F \oplus B)(x) = sup\{f(x-m) + b(m)\} \quad \text{where } m \in S, x-m \in P \tag{3.6}$$

$$(F \ominus B)(x) = inf\{f(x+m) - b(m)\} \quad \text{where } m \in S, x+m \in P \tag{3.7}$$

where $f(x \pm m)$ represents moving along a vector over the input image F. The morphological opening and closing operations of F and B are defined, respectively, by the combination of the erosion and dilation operations as follows [22]

$$(F \circ B)(x) = (F \ominus B) \oplus B \tag{3.8}$$

$$(F \bullet B)(x) = (F \oplus B) \ominus B \tag{3.9}$$

Based on (3.8) and (3.9), the opening Top- Hat and closing Top-Hat operations are defined, that is [22]

$$OTH_{F,B}(x) = (F - F \circ B)(x) \quad (3.10)$$

$$CTH_{F,B}(x) = (F \bullet B - F)(x) \quad (3.11)$$

The Top-Hat operation has some characteristics of high-pass filtering. Therefore the opening Top- Hat operator can detect the peaks of an image and the closing Top-Hat operator can detect the minimum points [22]. Opening Top-Hat is appropriate for dim target detection considering its characteristics of filtering. In *Figure 3.5*, the Top-Hat operation output of the SENSOR TYPE-1 image is shown.

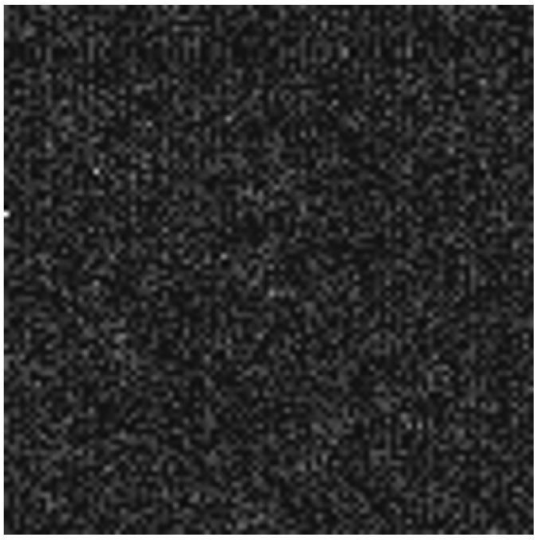


Figure 3.5 SENSOR TYPE-1 image after Opening Top Hat Morphological Operation

3.4.1.3. PROJECTING OPERATION

Even after improving the image SNR with morphological operations, the target still cannot be detected. The need for gathering statistical information of the image sequences arises in order to detect the target or target candidates. This operation called projecting [12] performed by summing the preprocessed images along the time axis can be formulated as [23]

$$I_c(x,y) = \sum_{k=0}^{m-1} \frac{D(x,y,k)}{\sigma_k} \quad (3.12)$$

where $D(x,y,k)$ is the preprocessed image, m is the number of images that have taken part in this projecting operation and σ_k is the standard deviation of an image $D(x,y,k)$.

The parameter *m* is chosen as 3, as small as possible, to minimize time delay. *Figure 3.6* shows the final image of the projected 1st frame of the SENSOR TYPE-1 image.

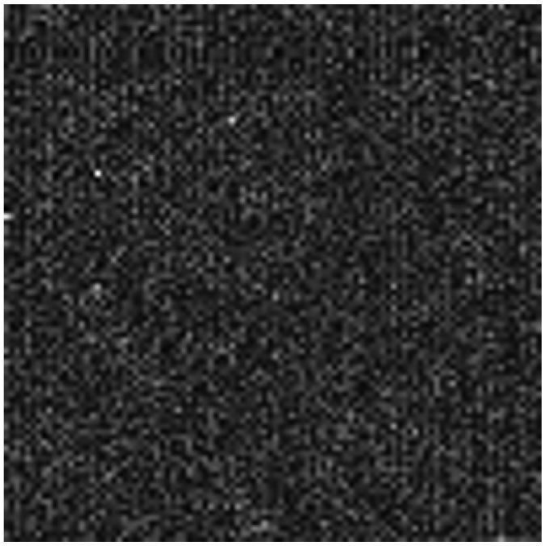


Figure 3.6 SENSOR TYPE-1 sequence first frame after projection operation

3.4.1.4. ADAPTIVE THRESHOLDING AND BINARIZATION

After the projection operation, a thresholding step is needed to distinguish the candidate points from background pixels. Considering the change of the image in each frame, an adaptive thresholding solution has been found more appropriate. We use a mean and standard deviation dependent threshold as in [24]:

$$T = \mu + \alpha\,\sigma \tag{3.13}$$

where μ is the mean and σ is the standard deviation of the selected frames. α is a variable for changing the threshold level. Once the threshold is set, the points whose intensities are above the threshold T are determined. They are the candidate points which may be real target points, background points or noise points.

The candidate target locations are obtained by binarizing the thresholded image. *Figure 3.7* shows the output of this stage for the 1st frames of the SENSOR TYPE-1 and SENSOR TYPE-2 datasets for the target intensity level of 30. The target is circumscribed in red and is in the coordinates [70,1]. One can see that reducing the threshold results in more candidate points to be tracked by the PF algorithm.

(a) 1st frame of SENSOR TYPE-1 image when $\alpha = 2.5$

(b) 1st frame of SENSOR TYPE-1 image when $\alpha = 4$

(c) 1st frame of SENSOR TYPE-2 image when $\alpha = 2.5$

(d) 1st frame of SENSOR TYPE-2 image when $\alpha = 4$

Figure 3.7 Output of thresholding and binarization for two datasets (when target intensity level is 30)

3.4.2. PARTICLE FILTER TARGET TRACKING

From this step on, candidate points which are the output of the preprocessing algorithm will be tracked separately and eliminated progressively based on a motion consistency error criterion computed on the estimated trajectories. The observation and target state transition models given in Chapter 3 are used in the PF algorithm. Some difficulties have been faced on datasets where the dim target has a very low intensity compared to the background. In order to overcome these difficulties, we have used some hybrid

weighting strategies in the PF algorithm is strengthened with using some information feedback algorithms which will also be detailed in PF target tracking steps.

In Section 2.6.1, we describe the conventional scenario where the weights of the particles are set based only on intensity measurements, while in Section 3.4.2.2.1 we present the hybrid weighting strategy that also employs a motion consistency criterion.

3.4.2.1. BACKGROUND ESTIMATION

We first discuss an online background estimation strategy for background subtraction, in order to reduce the effect of background intensity changes on the performance of the algorithm.

In order to estimate a background image, it is required to have a few frames that represent the properties of the true background. Although the background may change over time due to temporal variations of the background pattern, an estimate can be found by calculating the mean of the observed frames until a given time instant. The necessary number of accumulated frames depends on two important facts:

- ✓ The more the number of accumulated frames is, the better the background will be estimated.
- ✓ When the background is estimated from a small number of frames, the shadow of the target can appear in the background-subtracted frames because of its slow motion.

To be more specific, if the algorithm waits too much to acquire a sufficient number of frames for the estimation of the background, it can cause an error accumulation in the PF algorithm. But if the background is estimated from a small number of frames, the probability of subtracting the target from the input image is higher. We find a compromise by applying an adaptive approach, which is detailed in **Algorithm 8** and **Algorithm 9**.

Algorithm 8 Background Accumulation Algorithm in Projecting Operation Step

Total Image=0

for *k= 1 : 3*

- $Total\ Image = Total\ Image + \ Input\ Image(k)$

end for

Algorithm 9 Background Subtraction Algorithm in Particle Filter

for *k= 4 : # of frames*

- $Total\ Image(k) = Total\ Image(k-1) + \ Input\ Image(k)$
- $Estimated\ image = Total\ image/k$
- $Subtracted\ Image(k) = \ Input\ Image(k) \ - \ Estimated\ image$
- *Use subtracted image for PF algorithm*

end for

The background estimation process decreases the tracking error and helps the PF algorithm to estimate the target more accurately. A subtracted image example is given in *Figure 3.8.*

Figure 3.8 10th frame of subtracted image for SENSOR TYPE-1 data

3.4.2.2. PARTICLE FILTER IMPLEMENTATION

A candidate list of potential targets is generated from the above preprocessing steps and given as input to the PF algorithm. In this section, we explain how the weights

can be set based only on intensity measurements. In accordance with the theoretical explanation of the PF SIR algorithm in Chapter 2, in the state update equations, $p(x_k|x_{k-1}^i)$ is defined based on the target state transition model, the measurement z_k is defined as the intensity of the estimated target point and the weight update is defined based on the intensity change. The distribution of the weights of the particles are set as follows

$$P_w(m) = 1/\sqrt{2\pi{\sigma_k}^2}\exp(-(z - z_k(m))^2/(2{\sigma_k}^2)) \quad m=1,2,\ldots N \tag{3.14}$$

where z is the reference intensity of the target, measured at the initially detected target pixel in the beginning of the sequence, $z_k(m)$ is the *mth* estimate from the *mth* particle, and σ_k is the standard deviation of the image intensity on the *kth* frame. *N* represents the number of particles.

Algorithm steps will be presented in a specific order, described below:

Algorithm 4 Particle Filter Implementation

Input the list of candidate points

for *k= 4: # of frames*

for *n=1:# of candidate points*

for *i= 1: N_s*

- $x_k^i \sim q(x_k^i|x_{k-1}^i, z_k)$
- *Take the new measurement z_k for x_k^i*
- *Assign the particle a weight, ω_k^i according to (3.14)*

end for

- *Normalize: $\omega_k^i = t^{-1}\omega_k^i$*
- *Resample weights (**Algorithm 2**)*
- *Calculate final estimated position as the mean value of resampled particle points*

end for

- *Save estimated trajectory for each candidate*

end for

This algorithm runs for each frame and at the end of each frame it estimates the new target position for each one of the candidate targets.

During the PF algorithm implementation, a couple of issues are faced with and some approaches to solve these problems are proposed. These issues and solution oriented approaches are discussed in the next section.

3.4.2.2.1. WEIGHTING STRATEGIES IN PF

3.4.2.2.1.1. STANDARD PARTICLE WEIGHTING

As described in Section 2.6 standard PF weighting algorithm uses only the intensity measurements. This may results in weighting the noisy pixel more, as a consequence of the intensity similarities between background pixels and target pixel. Thus algorithm may track the background pixel.

3.4.2.2.1.2. INTENSITY AND MOTION INFORMATION COMBINED PARTICLE WEIGHTING

In the PF, the weights of the particles are determined according to the intensity measurements. In scenarios where the target intensity is very low and the background has a pattern with intensity close to that of the target, for instance as in the SENSOR TYPE-1 images in *Figure 3.3*, computing the weights purely based on the intensity measurements may fail as the algorithm favors particle locations with higher intensity even though the motion model directs it where to search the new estimate.

The proposed solution is to use not only intensity dependent weighting but also to use the information about the motion model that is learned from the past behavior of the target.

In order to apply this, we define likelihood functions of particles based on the velocity change in both x and y coordinates as in (3.16). This weight is then multiplied by the weight in (3.14), so that the overall weights of the particles depends on both intensity and the motion behavior of the candidate.

$$P_{w_p}(m) = 1/\sqrt{2\pi{\sigma_v}^2}\exp\left(-\left(\Delta v - v_k(m)\right)^2/(2{\sigma_v}^2)\right) \text{ m=1,2,…N} \qquad (3.16)$$

Here Δv represents the mean velocities in both horizontal and vertical directions, $v_k(m)$ represents the displacement vector of the *mth* particle from the *(k-1)th* estimated candidate position. Velocity dependent weighting is chosen for reducing the effect of intensity and imposing the consistency of the motion of the particles with the current trajectory estimates. Combining the intensity information with motion consistency clues is especially helpful for not losing the target on certain frames of the sequence where the intensity information becomes temporarily unreliable.

This solution is proposed considering the real target motion where the velocity of the target cannot change rapidly. Even though the velocity of the target may vary over time, this usually happens smoothly in practice.

Algorithm 7 summarizes the steps of the combined weighting process. *K* represents the number of frames up to which the mean velocities are calculated. We have experimentally observed that this combined weighting strategy has been very useful for successfully tracking the target in image sequences such as SENSOR TYPE-1 where the background has a challenging pattern.

We have so far discussed several strategies to address the aforementioned challenges in dim point target tracking. All of the proposed approaches aim to handle low SNR problems.

Algorithm 7 Combined Particle Weighting

for *k= K: # of frames*

  for *i= 1: N_s*

- $x_k^i \sim q\left(x_k^i \middle| x_{k-1}^i, z_k\right)$
- *Take the new measurement z_k for x_k^i*
- *Assign the particle a weight, ω_k^i according to (3.14) and (3.16)*
- *Calculate $P_{w_p} = P_{w_p} P_w$*

end for

- *Normalize:* $\omega_k^i = 1/N_s\, \omega_k^i$
- *Resample weights (**Algorithm 2**)*
- *Calculate final estimated position as the mean value of resampled particle points*

end for

The performance evaluation of the algorithms detailed in this chapter will be presented in Chapter 4.

3.4.2.2.2. ELIMINATE CANDIDATE POINTS

For datasets with high intensity target, choosing a relatively high α value, thresholding and binarization process results in few candidates to track with the PF. Each candidate is taken as an input for the PF and the algorithm tracks it through the frames.

When the target intensity is low in a dataset, α should be decreased to a lower level for catching the target in the first frame. On the other hand, decreasing the threshold level produces dozens of candidate points for the PF. The PF algorithm performs better at large population sizes, but in the case of dozens of particle filters each having thousands of particles, finding the true target and tracking it becomes a computationally complex process and is not preferable for systems where a fast response is crucial.

The solution that we adopt for this problem is to begin tracking each one of the initially detected candidates and eliminate them gradually throughout the frames based on some error criterion on the estimated trajectories. This approach leads to analyzing a few frames, then eliminating some of the candidates based on their mean error in these frames, continuing to track the rest of the candidates for a few frames more, then eliminating again, and continuing this procedure until the number of candidates decreases to 2 or 1.

The error that the algorithm uses for the elimination process, is called the "Trajectory Cost" and is explained in **Algorithm 5**.

Algorithm 5 Calculation of trajectory cost for each candidate points based on motion consistency

for *n=1:# of candidate points*

for *k= 4: # of frames*

- *Take the first M estimated points in consecutive frames (p_1,p_2,..,p_M)*
- *Calculate the displacement* Δx *in x direction and the displacement* Δy *in y direction*; $\Delta x_1 = p_{M_x} - p_{1_x}$ and $\Delta y_1 = p_{M_y} - p_{1_y}$
- *Define* $\Delta v_1 = [\Delta x_1;\ \Delta y_1]'$
- *Take the second Mth points beginning from the p_M (p_M, p_(M+1),…,p_(2M-1))*
- *Calculate* $\Delta x_2 = p_{(2M-1)_x} - p_{M_x}$ and $\Delta y_2 = p_{(2M-1)_y} - p_{M_y}$
- *Define* $\Delta v_2 = [\Delta x_2;\ \Delta y_2]'$
- *Calculate* $\frac{\|\Delta v_2 - \Delta v_1\|}{(\|\Delta v_2\| + \|\Delta v_1\|)/2}$ *as trajectory cost*

end for

- *Calculate mean value of trajectory cost for all candidate points*
- *Eliminate candidates that have* $Traj\_cost_n \geq mean(Traj\_cost_{1:n})$
- *Make a new list of residual candidates*

end for

As described above, the trajectory cost is computed over group of frames of size *M*. The aim of the algorithm is to check how much the velocity changes throughout the frames. The proposed trajectory cost is thus helpful for eliminating trajectories where the velocity of the target changes unexpectedly over adjacent groups of frames, which is not likely to happen in a real scenario. In order to control the speed of elimination of the candidates within the first few frames, the windows are chosen to consist of some variable size of frame intervals.

3.4.2.2.3. UPDATE OF THE STATE TRANSITION MODEL

Particles near the target have weights larger than other particles, since their intensity values are higher and thus closer to the predetermined target intensity. As a result, the particle set moves along with the target.

In settings with low target intensity, it becomes more difficult for the algorithm to distinguish between the target and the noisy background. To overcome this issue, the expected displacement vector B vector in (3.2) is updated in every S frames and fed back to the state transition model. The algorithm is given below:

Algorithm 6 Update B in state transition

Take estimated point in k^{th} frame, p_k

s=1

for *k=k+1: # of frames*

- *Take estimated point in $(k+1)^{th}$ frame , p_{k+1}*
- *Calculate Δx displacement in x direction and Δy displacement in y direction*; $\Delta x = p_{k+1_x} - p_{k_x}$ and $\Delta y = p_{k+1_y} - p_{k_y}$
- *Define* $\Delta v(k) = [\Delta x; 0; \Delta y; 0]'$
- $p_k = p_{k+1}$
- $s = s + 1$

  if *s=S*

  - *mean(Δv)* = $\frac{1}{S}\sum_k \Delta v(k)$
  - *B= mean(Δv)*

  end if

end for

The displacement vector is thus computed as the average displacement of the target between adjacent frames along the estimated trajectory. This update process helps particles to move towards a direction that is predicted from the target's past motion information.

CHAPTER 4

EXPERIMENTAL RESULTS

In this chapter we experimentally study the performance of the proposed algorithm. The algorithm is applied on 2 different data sequences (SENSOR TYPE-1 and SENSOR TYPE-2) and tested at different target intensity values. The synthetically generated target motion is embedded into each image sequences. The speed of the target in both the horizontal and diagonal motion models is selected as 1 pixel per 2 frames, in order to realistically model the motion of a real target at a distance of 10-15 kms away from the sensor.

Point target SNR calculation of an IR image sequence can not be done using in traditional SNR, PSNR, etc. definitions that concern logarithmic equations. According to J.Tang et al. [21] dim target SNR can be calculated as:

$$SNR = \frac{\sum_i^{N_s} abs(S_i - \bar{B})}{N_s} \Bigg/ \sqrt{\frac{\sum_i^{N_B} (S_B - \bar{B})^2}{N_B}} \tag{4.1}$$

where S_i is the signal value corresponding to each pixel i. S_B is the signal value corresponding to each background pixel. $\bar{B}$ is the mean of the background signal, defined as $\bar{B} = \sum_i^{N_B} S_B / N_B$. The parameter N_B is the total number of pixels in the processed area, in this case 100 x 100, 10000 pixels without the target pixel, only background. N_S is the number of pixels occupied by the target which is 1.

The SNR levels that are calculated with respect to the definition in (3.1) are given in *Table 4.1, Table 4.2* and *Table 4.3*. The SNR value of each frame in the sequence is calculated, and the minimum and mean SNR values over all frames are reported in the tables SNR values are small and too close to each other, in the following analyses and discussions image sequences will be referred to by their target intensity level for better comprehension.

Table 4.1 Mean and Minimum SNR for SENSOR TYPE-1 and SENSOR TYPE-2 image sequences (intensity levels 100, 60 and 50)

Intensity Level / Data	Mean+100		Mean+60		Mean+50	
	Min SNR	Mean SNR	Min SNR	Mean SNR	Min SNR	Mean SNR
SENSOR TYPE-1	0.1079	0.1093	0.0648	0.0656	0.0540	0.0546
SENSOR TYPE-2	0.2249	0.2423	0.1349	0.1454	0.1124	0.1212

Table 4.2 Mean and Minimum SNR for SENSOR TYPE-1 and SENSOR TYPE-2 image sequences (intensity levels 30, 20 and 10)

Intensity Level / Data	Mean+30		Mean+20		Mean+10	
	Min SNR	Mean SNR	Min SNR	Mean SNR	Min SNR	Mean SNR
SENSOR TYPE-1	0.0324	0.0328	0.0216	0.0219	0.0108	0.0109
SENSOR TYPE-2	0.0675	0.0727	0.0450	0.0485	0.0225	0.0242

Table 4.3 Mean and Minimum SNR for SENSOR TYPE-1 and SENSOR TYPE-2 image sequences (intensity levels 15, 14 and 13)

Intensity Level / Data	Mean+15		Mean+14		Mean+13	
	Min SNR	Mean SNR	Min SNR	Mean SNR	Min SNR	Mean SNR
SENSOR TYPE-1	0.0162	0.0164	0.0151	0.0153	0.0140	0.0142
SENSOR TYPE-2	0.0337	0.0363	0.0315	0.0339	-	-

The algorithm is first tested for high SNR targets. In these datasets, the target can be easily seen and tracked. But in lower SNR cases, distinguishing the target from the background becomes difficult and thresholding becomes a particularly critical stage, where the threshold must be chosen properly to avoid the risk of not detecting the candidate in the beginning of the sequence. After encountering such problems, we have decided to eliminate candidates with preprocessing and the adaptive thresholding steps.

As described in Chapter 3, the proposed algorithm starts with preprocessing the image sequence. The preprocessing algorithms are applied only on the first 3 frames of the input image to gather the statistical information. Primarily the aim is to eliminate as many candidates as possible and keep the input list of candidates for PF algorithms. The first step is to eliminate temperature effects using (3.4) and (3.5). After that, Top-Hat Morphological Filter is implemented in the algorithm for each image frame to reject the background while preserving the target in the image. Then the projecting operation is done for gathering the statistical information of the image sequences for the first 3 frames, and, candidate list is generated for the PF algorithm. The PF algorithm starts to tracks the candidates from the 4^{th} frame on. Hence, the studied tracking algorithm is implemented in a sequential and causal way, where each frame is processed using the information obtained from only the preceding frames.

Waiting for four frames to get the statistical information of target, the initial background image to be subtracted from the sequence is also obtained from the first four frames.

Following the projecting algorithm, the binarization process is applied. The threshold level is chosen as described in (3.13).The α values for the given datasets are selected as in *Table 4.4* .

Table 4.4 α values for given image sequences

Intensity Level / Data	Mean+ 100	Mean+ 60	Mean+ 50	Mean+ 30	Mean+ 20,15,14	Mean+ 10
SENSOR TYPE-1	9	8	6	4	3	2.5
SENSOR TYPE-2	9	8	6	4	3	2.5

After the thresholding step, the list of the locations of target candidates is generated and given as input to the PF algorithm.

Before running the PF algorithm, some parameters should be initialized firstly. These parameters are listed below.

- Population size of particles
- The *A* matrix and the *C* in (3.2)
- Number of frames (window length) during which target candidates are tracked before elimination

The population size parameter is generally set to the value, 5 000, which has been experimentally observed to give usually good results in the trials of the algorithm. The population size will be analyzed in the next section. The *A* matrix is defined regarding the constant velocity motion of the target as;

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

and *C* value is assigned as 0.5 for the general usage of the algorithm. The effect of the *C* parameter will also be examined in detail, later.

The values of window intervals are given below in *Table 4.5*. Window intervals are assigned regarding the difficulties of the image sequences, especially the SENSOR TYPE-1 sequence, and the beginning of the tracking with PFs.

Table 4.5 Chosen window intervals

Window number	Frame numbers
Window 1	4-10
Window 2	11-19
Window 3	20-28
Window 4	29-134

After initialization step is done, PF algorithm is ready to be run. Outputs of PF algorithm is the trajectory estimates and maximum estimation error which can be defined as the maximum squared error may done in each frame of estimation.

The first tests are conducted on, high SNR data sequences and the SNR levels are reduced gradually throughout the tests. The MSE and the maximum estimation error values are given in *Table 4.6*, *Table 4.7* and *Table 4.8* for the trials which are given in following figures. Dash cells mean, image sequence with the chosen motion was not available for this study. Image sequences were generated considering the performance comparison of the algorithm in different SNR and different image sequence patterns.

$$MSE = \frac{1}{N}\sum \left\| \begin{bmatrix} x_k \\ y_k \end{bmatrix} - \begin{bmatrix} \tilde{x}_k \\ \tilde{y}_k \end{bmatrix} \right\|^2 \tag{4.1}$$

Where x_k and y_k are the true location of the target and $\tilde{x}_k$ and $\tilde{y}_k$ are the estimated locations. *N* represents the total number of frames.

Table 4.6 MSE and Maximum Estimation Error for SENSOR TYPE-1 and SENSOR TYPE-2 image sequences where N=5000 (intensity level 100, 60 and 50)

Data \ Intensity Level		Mean+ 100		Mean+ 60		Mean+ 50	
	Motion number	Max Estimation Error	MSE	Max Estimation Error	MSE	Max Estimation Error	MSE
SENSOR TYPE-1	Motion 1	0.4274	0.0504	0.4738	0.0532	0.4294	0.0528
	Motion 2	0.2859	0.0254	0.2744	0.0233	0.2845	0.0261

SENSOR TYPE-2	**Motion 1**	0.4031	0.0523	0.4842	0.0537	0.4236	0.0536
	Motion 2	-	-	-	-	-	-

Table 4.7 MSE and Maximum Estimation Error for SENSOR TYPE-1 and SENSOR TYPE-2 image sequences where N=5000 (intensity level 30 and 20)

Intensity Level / Data		**Mean+ 30**		**Mean+ 20**	
	Motion number	**Max Estimation Error**	**MSE**	**Max Estimation Error**	**MSE**
SENSOR TYPE-1	**Motion 1**	3.4665	0.1742	3.8947	0.7032
	Motion 2	0.2901	0.0244	0.5785	0.1677
SENSOR TYPE-2	**Motion 1**	0.4261	0.0496	1.7588	0.3710
	Motion 2	0.2805	0.0246	0.5841	0.1701

Table 4.8 MSE and Maximum Estimation Error for SENSOR TYPE-1 and SENSOR TYPE-2 image sequences where N=5000 (intensity level 30, 20 and 10)

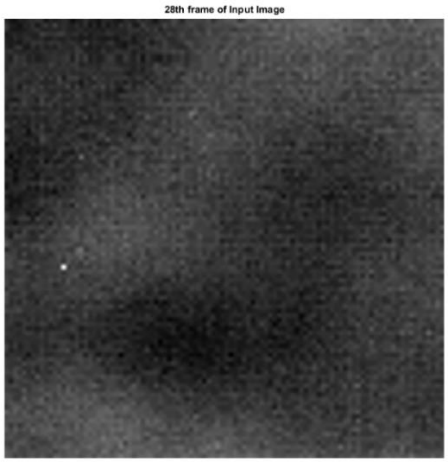
Intensity Level / Data		**Mean+ 15**		**Mean+ 14**		**Mean+ 13**		**Mean+ 10**	
	Motion number	**Max Estimation Error**	**MSE**	**Max Estimation Error**	**MSE**	**Max Estimation Error**	**MSE**	**Max Estimation Error**	**MSE**
SENSOR TYPE-1	**Motion 1**	9.0340	12.320	8.3779	14.901	12.840	55.306	37.242	379.87
	Motion 2	2.6710	0.2260	-	-	-	-	2.4577	0.5166
SENSOR TYPE-2	**Motion 1**	1.2659	0.4094	6.8281	3.2369	-	-	NO TRACK	
	Motion 2	-	-	-	-	-	-	-	-

Following figures show the outputs of the PF algorithm where the population size is N=5000 in all setups, except for the SENSOR TYPE-1 data motion 1 with target

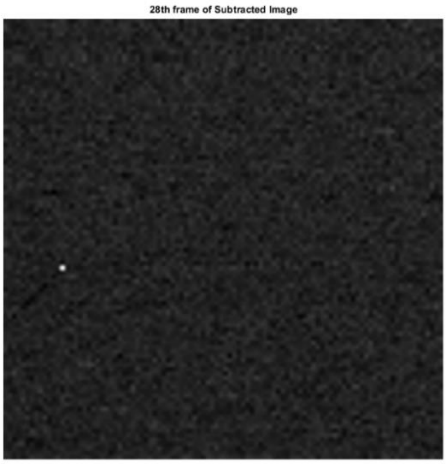
intensity levels 15, 14 and 13, where the population size is selected as N=7000. The input image and the image after background subtraction for the randomly chosen 28th frame are shown in each figure to visualize the SNR of the image sequences and the visibility of the target. Red circles are indicate the target when it is not visible. In each figure, the evolution of the estimation error for the sequence and a trajectory estimation are given. The aim of this representation is to compare and visualize both the estimated trajectory and the estimation error obtained on the given sequence.

In the cases where intensity levels are below 20, the PF algorithm needs a feedback loop where the estimated motion model captured by the *B* vector directs the particles towards the correct pixel and velocity information is included in the particle weights to reduce the dependency on intensity measurements. It is also necessary to increase the population size to reduce the MSE of tracking.

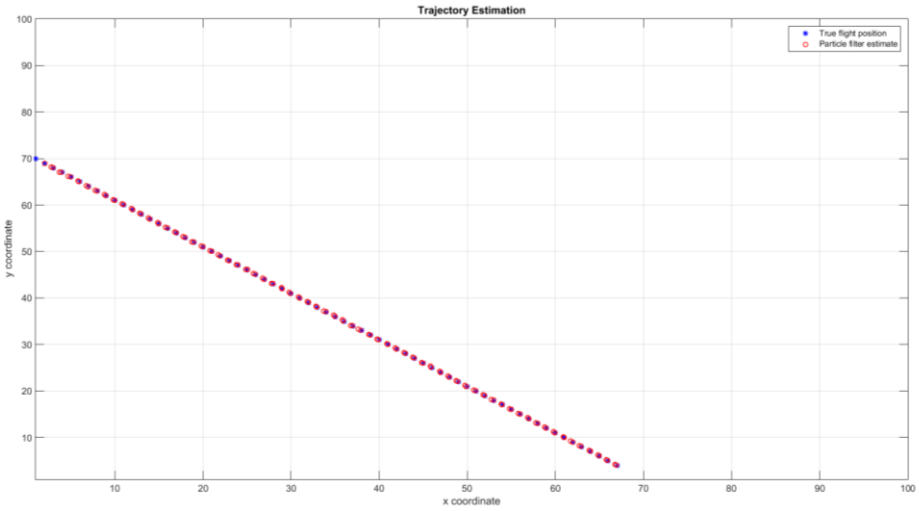
According to *Table 4.6*, Table 4.7 and *Table 4.8* as the intensity of the target decreases, the MSE increases gradually. But in the levels of 15, 14 and 13; the SNR gets lower and the estimation error of the PF algorithm increases. Although it is an expected observation, the proposed algorithm reaches its limit in Motion Model 1 approximately in the target intensity level of 10 based on the effect of the background pattern.



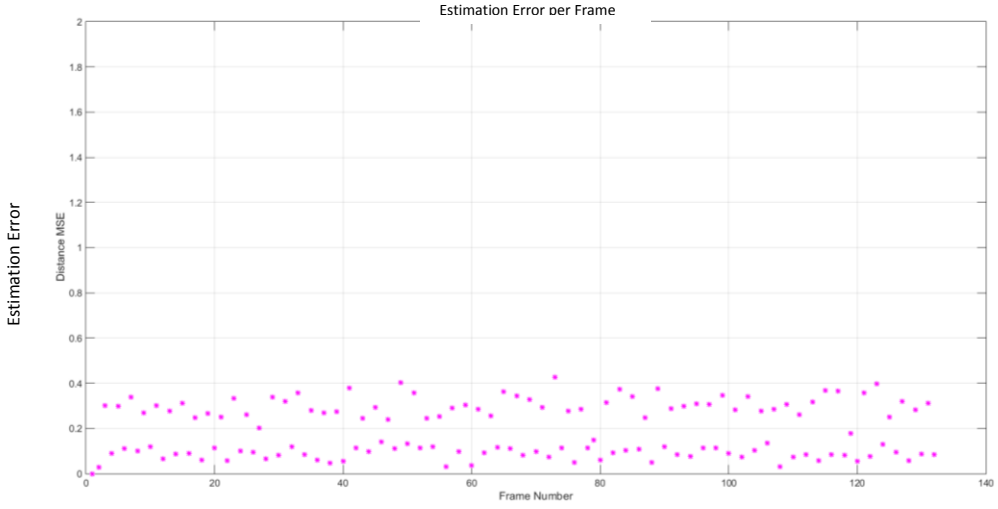
a. 28th frame of Input Image



b. 28th frame of background subtracted Input Image



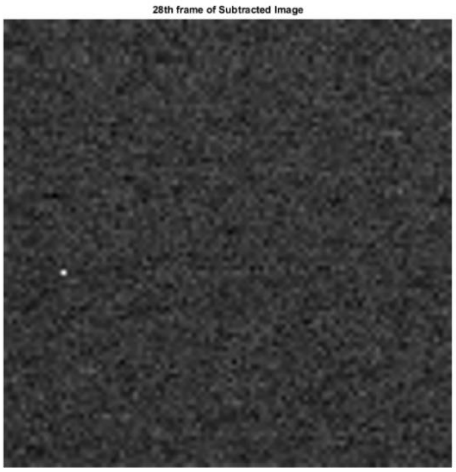
c. Trajectory Estimation output of PF algorithm



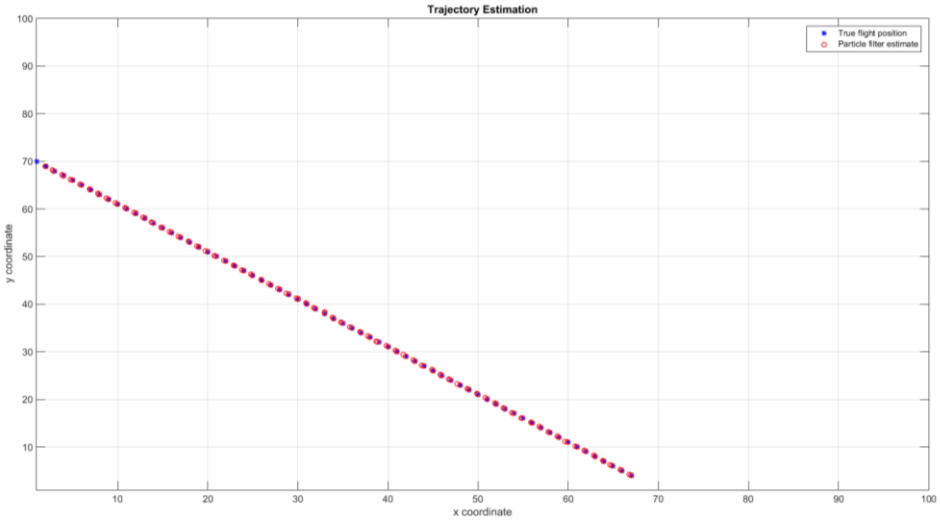
d. Estimation Error for each Frame

Figure 4.1 SENSOR TYPE-1 data for Motion 1 with target intensity level 100

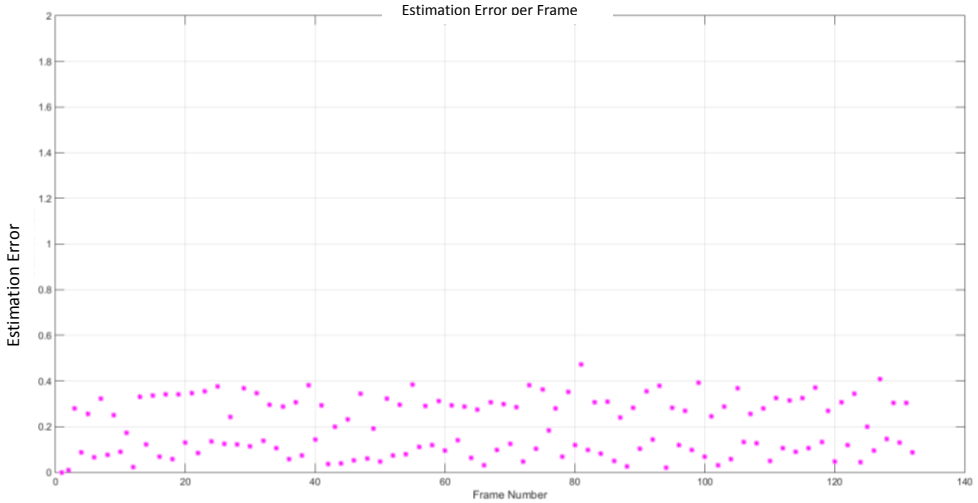
a. 28th frame of Input Image



b. 28th frame of background subtracted Input Image



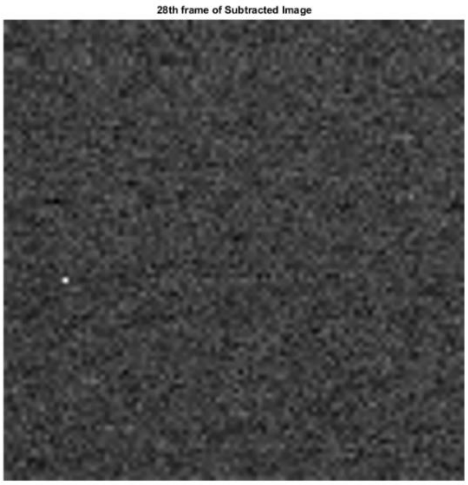
c. Trajectory Estimation output of PF algorithm



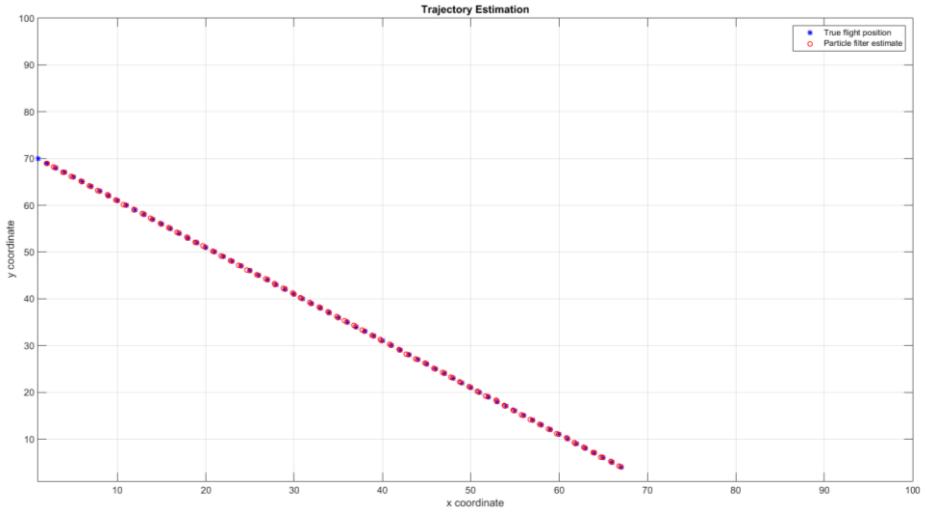
d. Estimation Error for each Frame

Figure 4.2 SENSOR TYPE-1 data for Motion 1 with target intensity level 60

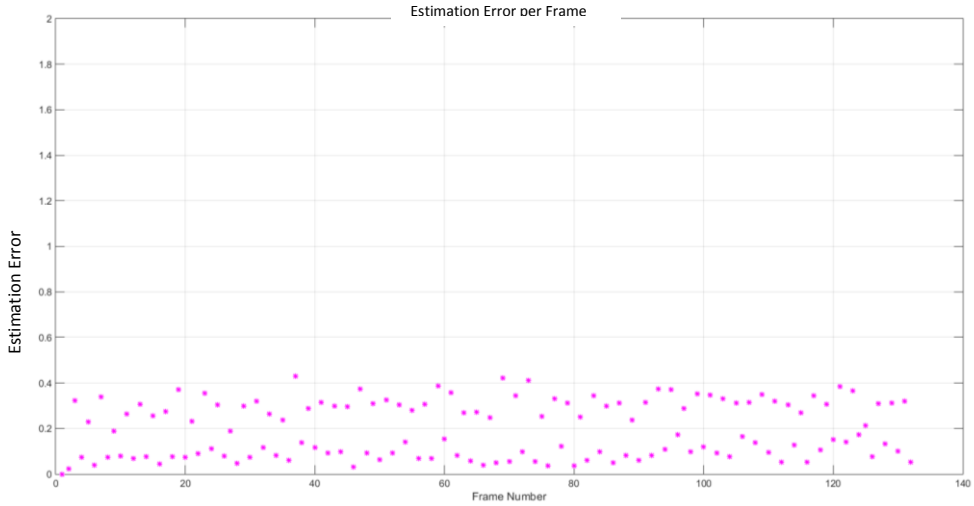
a. 28th frame of Input Image



b. 28th frame of background subtracted Input Image



c. Trajectory Estimation output of PF algorithm



d. Estimation Error for each Frame

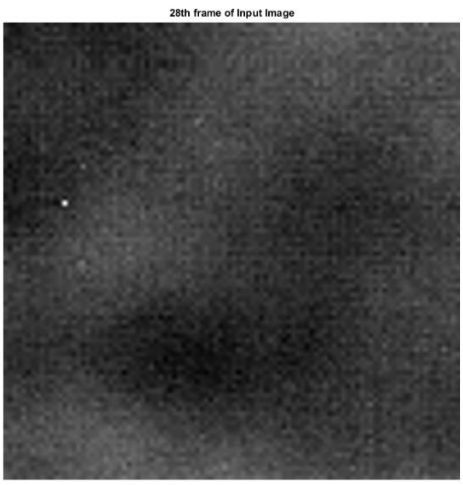
Figure 4.3 SENSOR TYPE-1 data for Motion 1 with target intensity level 50

In *Figure 4.1*, *Figure 4.2* and *Figure 4.3* simulation results are given for the SENSOR TYPE-1 image sequence with target intensity levels 100, 60 and 50 where the target motion is Motion 1. In this situation, there is no need to use B_k in the state model and the velocity based weighting algorithm. Tracking algorithm uses only intensity measurements and the estimated trajectory accurately traces that of the actual target. As one can see, the estimated target trajectories match with the true trajectories with the given MSEs in each frame. As given in *Table 4.6* the MSE values are around 0.05 pixel per frame.

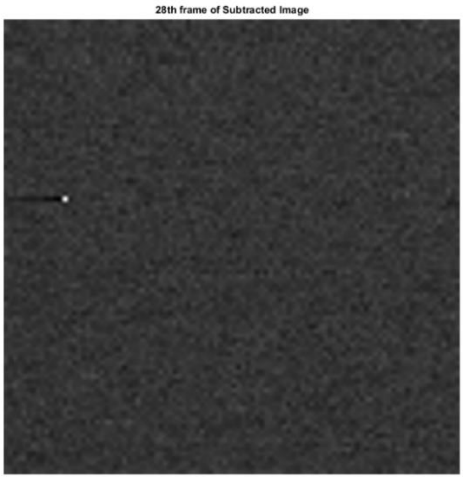
In *Figure 4.4*, *Figure 4.5* and *Figure 4.6* the results of SENSOR TYPE-1 data in Motion 2 with target intensity levels 100, 60 and 50 are given. The accuracy of the algorithm can be seen in both the estimated trajectories and the MSE plots. As given in *Table 4.6* the MSE values are about 0.02 pixel per frame.

Figure 4.7, *Figure 4.8* and *Figure 4.9* present the results obtained on the SENSOR TYPE-2 data sequence under the same conditions with Motion 1. As given in the *Table 4.6* the MSE values are about 0.05 pixel per frame.

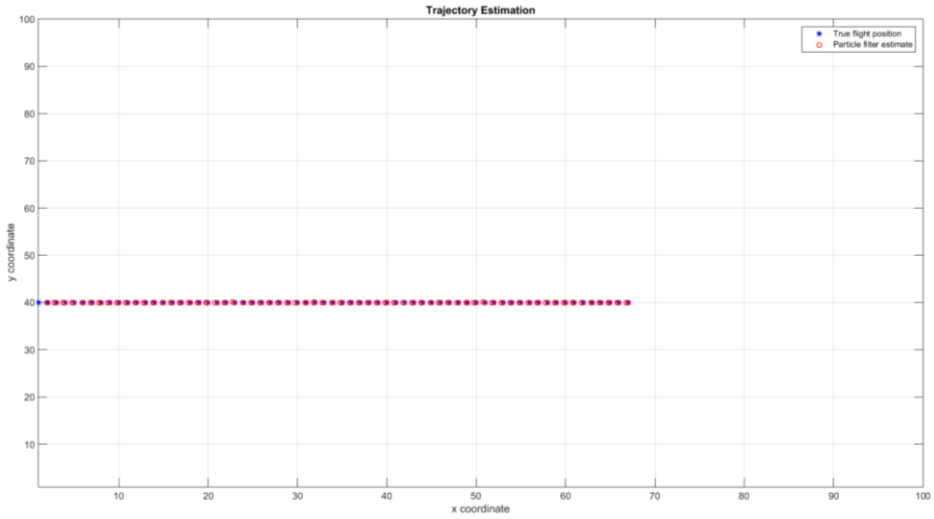
As a result of high intensity, the PF algorithm can accurately distribute, weight and resample the particles by using only the intensity measurements only. In such situations where the SNR is high, the PF algorithm performs perfectly as expected.



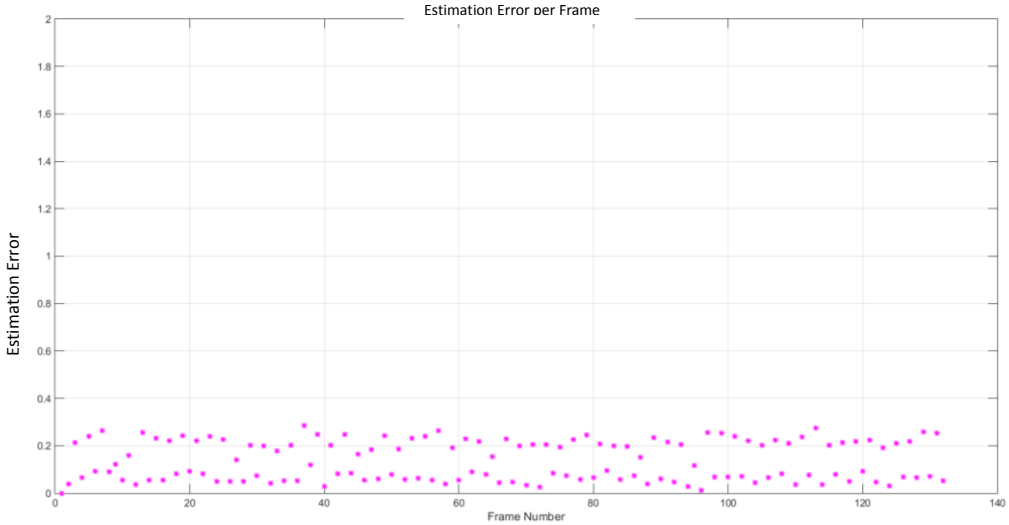
a. 28th frame of Input Image



b. 28th frame of background subtracted Input Image

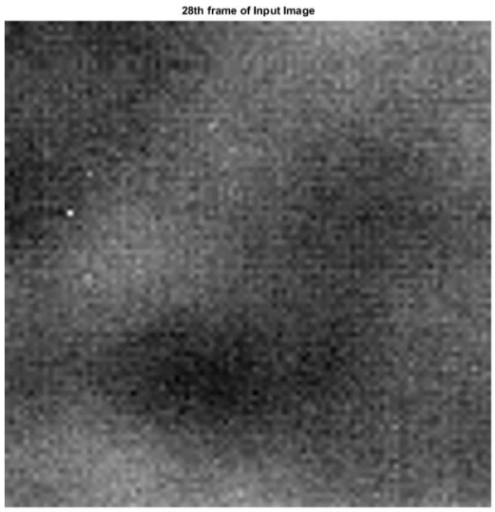


c. Trajectory Estimation output of PF algorithm

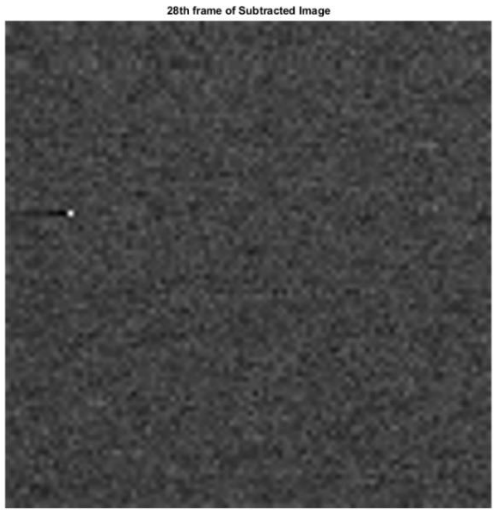


d. Estimation Error for each Frame

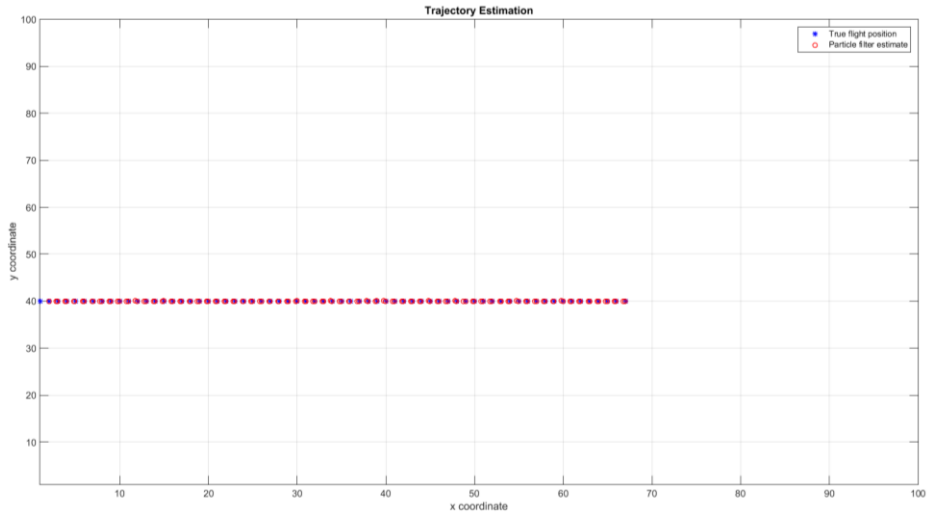
Figure 4.4 SENSOR TYPE-1 data for Motion 2 with target intensity level 100



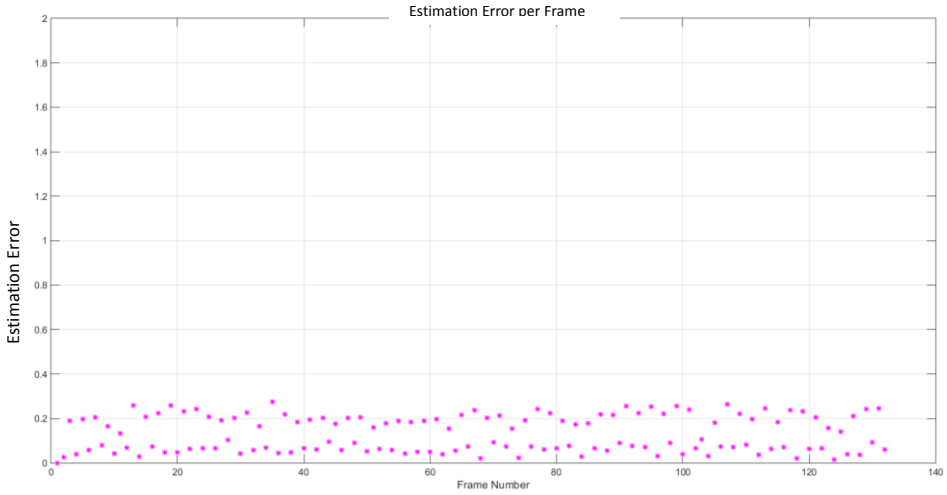
a. 28th frame of Input Image



b. 28th frame of background subtracted Input Image

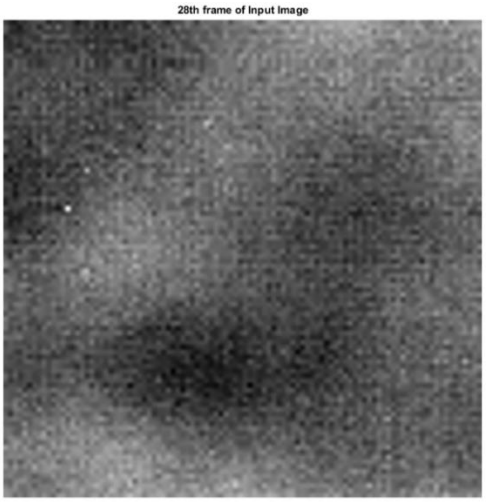


c. Trajectory Estimation output of PF algorithm

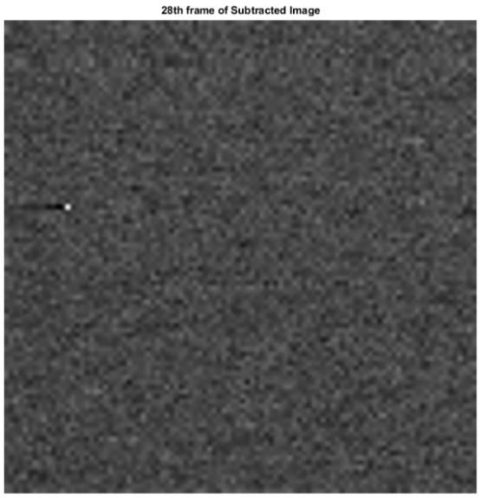


d. Estimation Error for each Frame

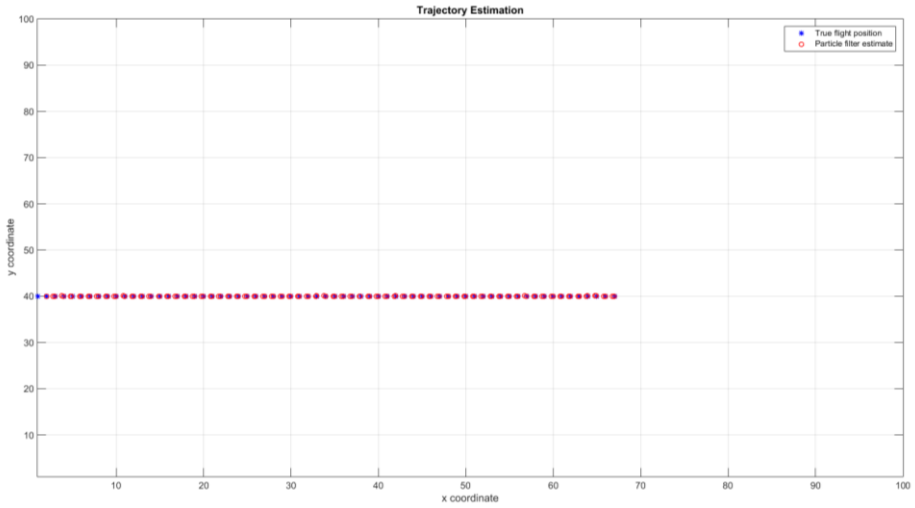
Figure 4.5 SENSOR TYPE-1 data for Motion 2 with target intensity level 60



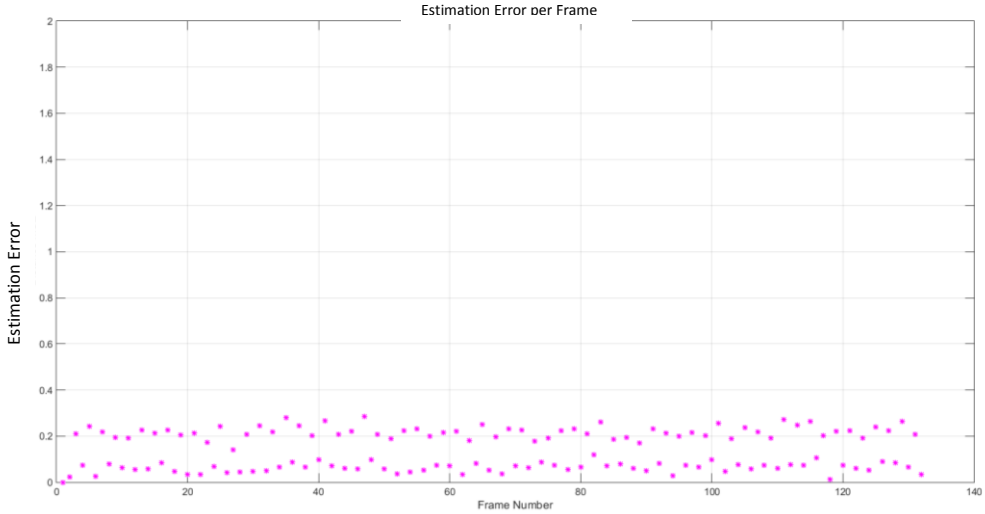
a. 28th frame of Input Image



b. 28th frame of background subtracted Input Image

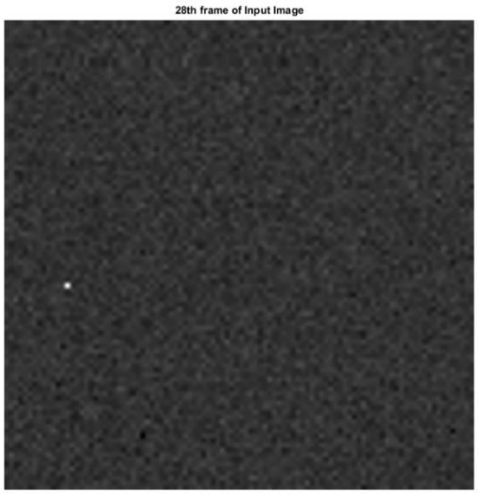


c. Trajectory Estimation output of PF algorithm

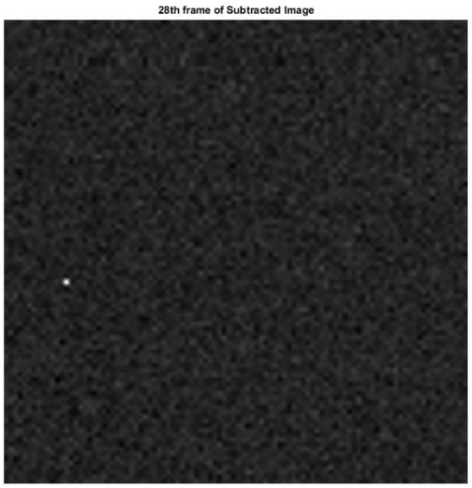


d. Estimation Error for each Frame

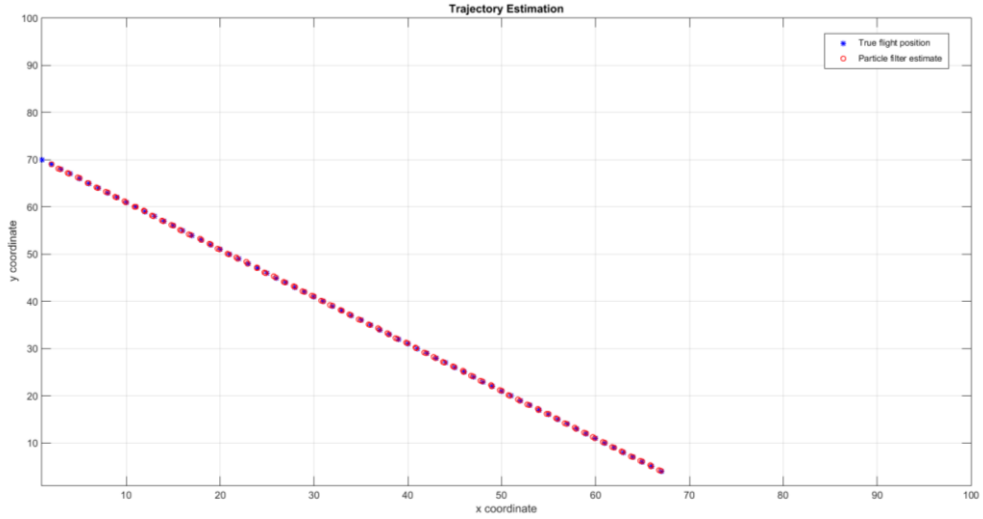
Figure 4.6 SENSOR TYPE-1 data for Motion 2 with target intensity level 50



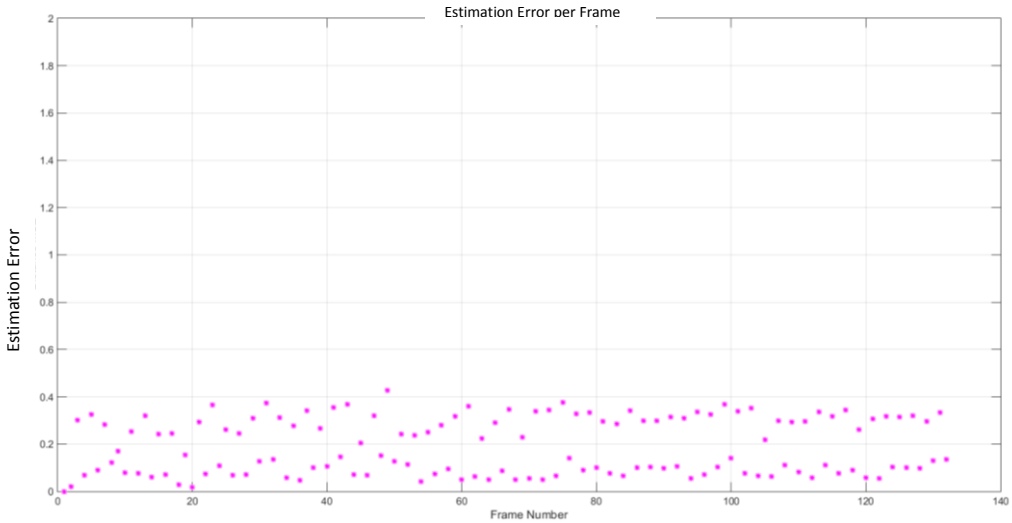
a. 28th frame of Input Image



b. 28th frame of background subtracted Input Image

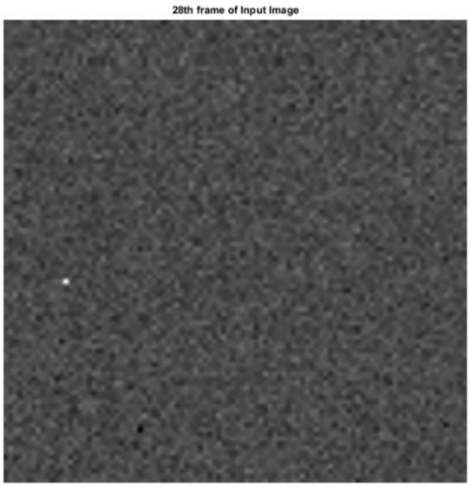


c. Trajectory Estimation output of PF algorithm

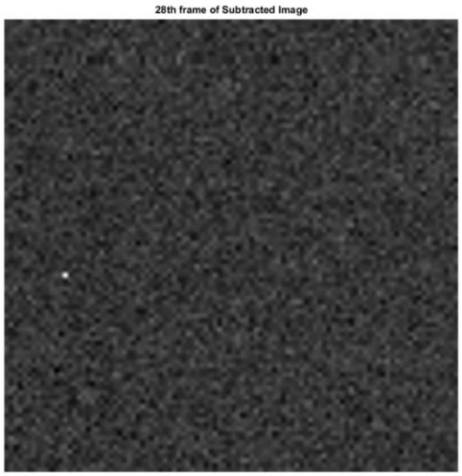


d. Estimation Error for each Frame

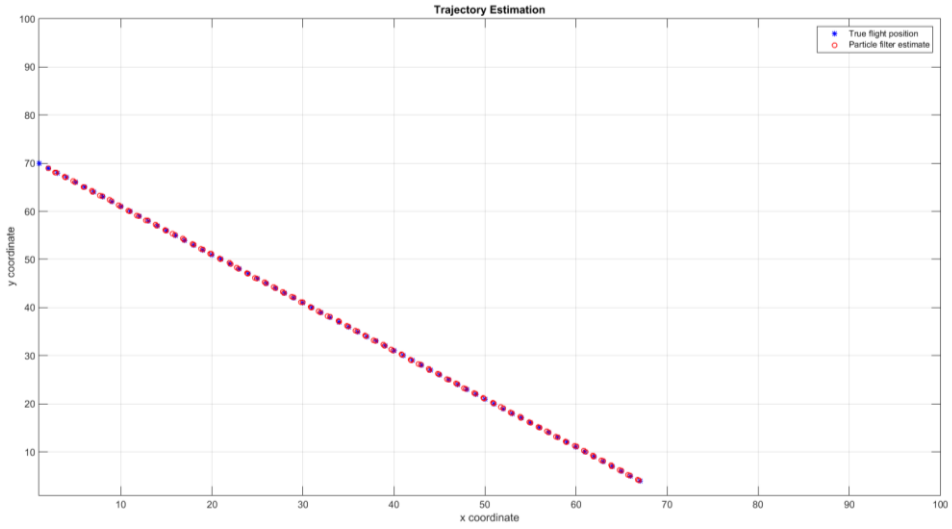
Figure 4.7 SENSOR TYPE-2 data for Motion 1 with target intensity level 100



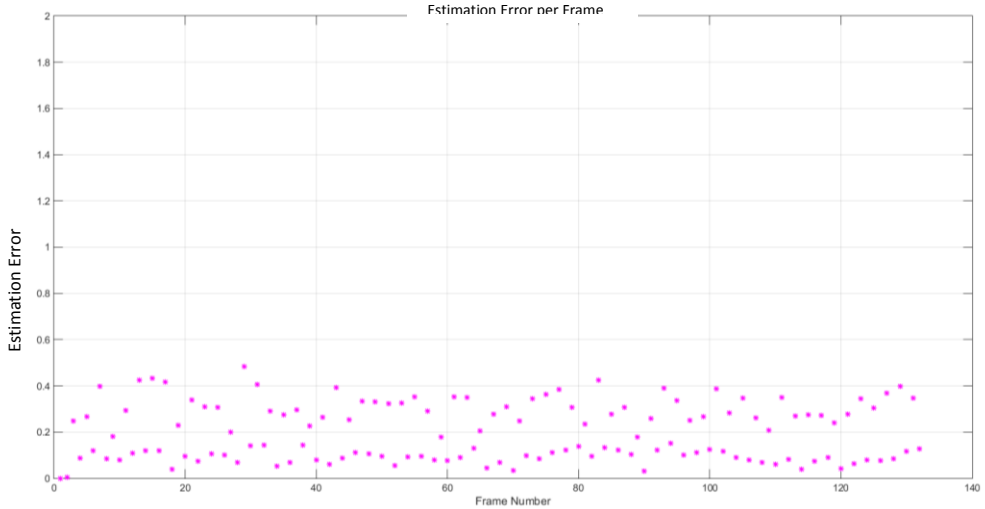
a. 28th frame of Input Image



b. 28th frame of background subtracted Input Image

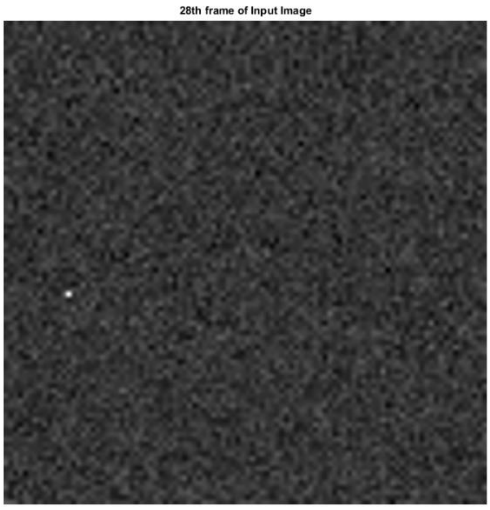


c. Trajectory Estimation output of PF algorithm

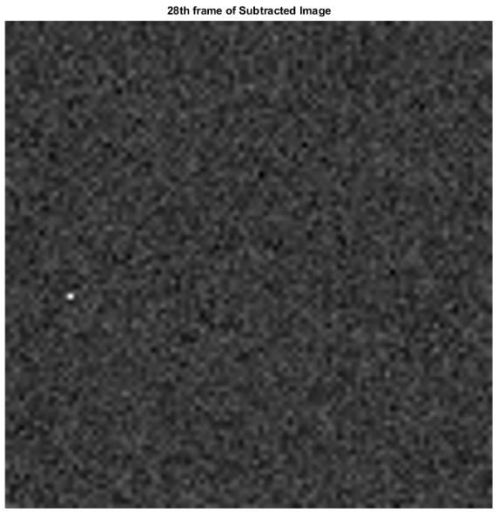


d. Estimation Error for each Frame

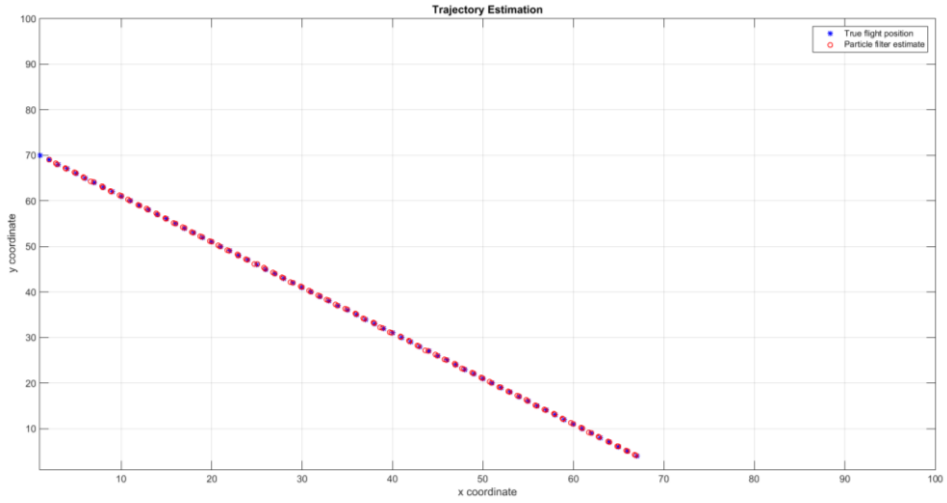
Figure 4.8 SENSOR TYPE-2 data for Motion 1 with target intensity level 60



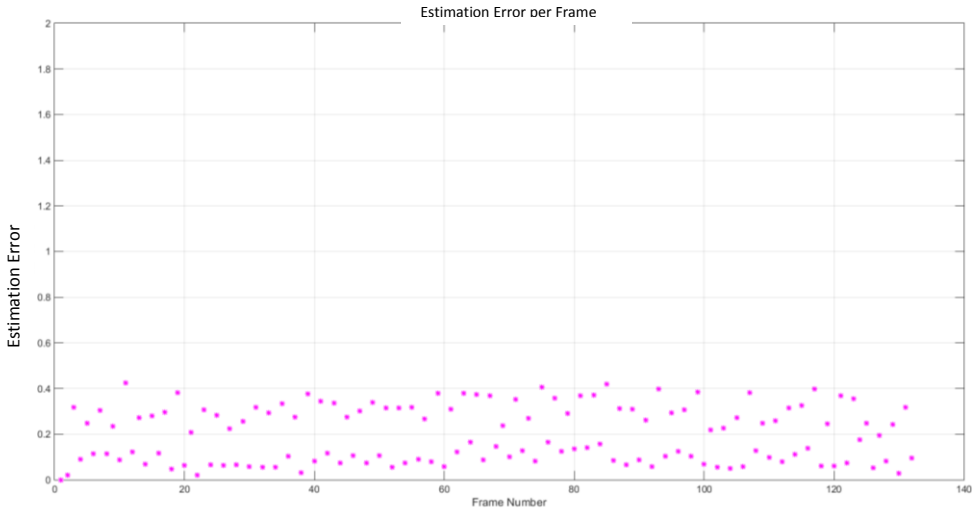
a. 28th frame of Input Image



b. 28th frame of background subtracted Input Image

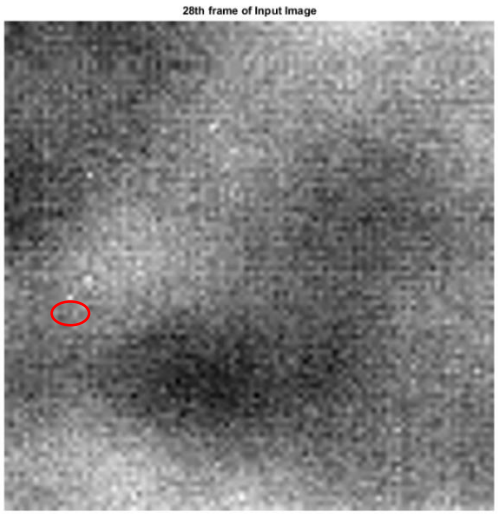


c. Trajectory Estimation output of PF algorithm

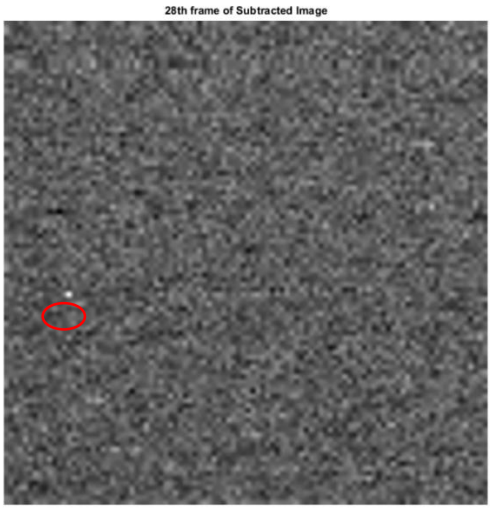


d. Estimation Error for each Frame

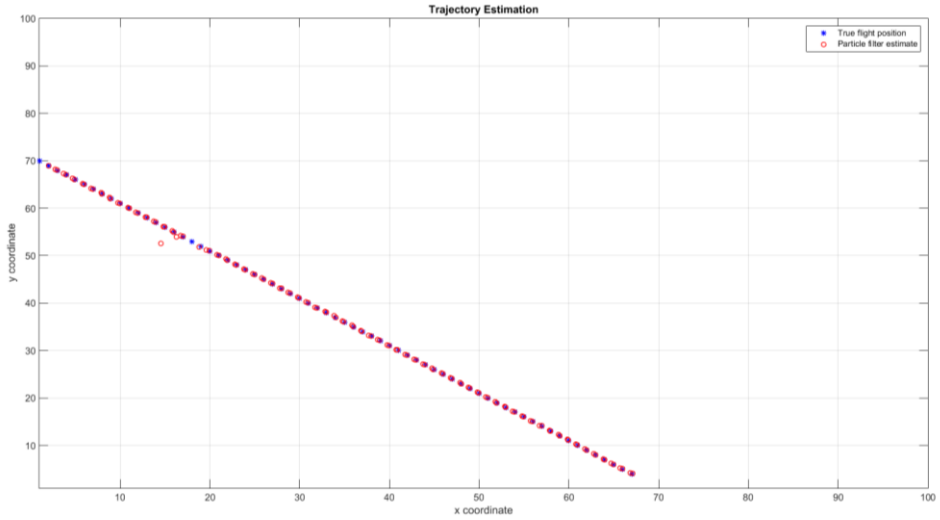
Figure 4.9 SENSOR TYPE-2 data for Motion 1 with target intensity level 50



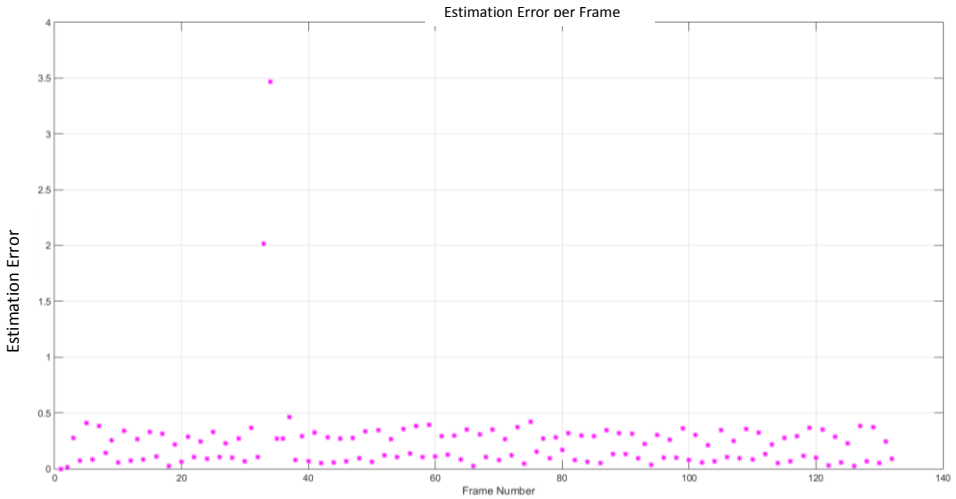
a. 28th frame of Input Image



b. 28th frame of background subtracted Input Image

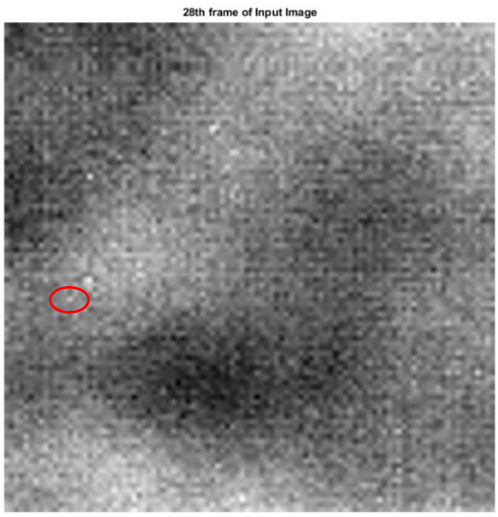


c. Trajectory Estimation output of PF algorithm

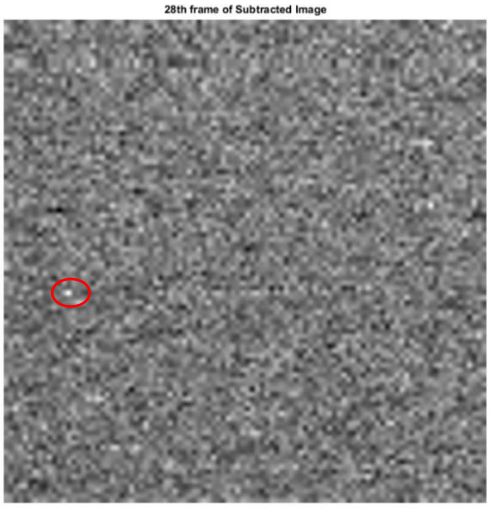


d. Estimation Error for each Frame

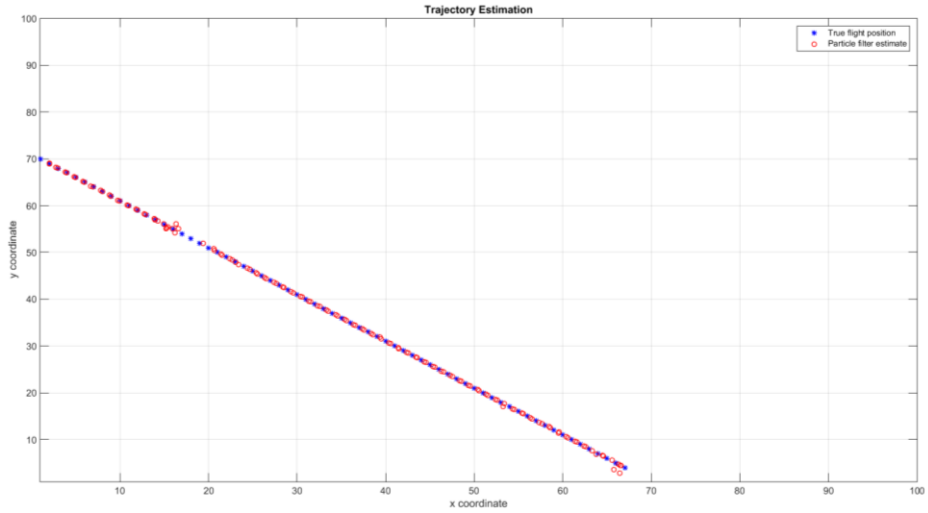
Figure 4.10 SENSOR TYPE-1 data for Motion 1 with target intensity level 30



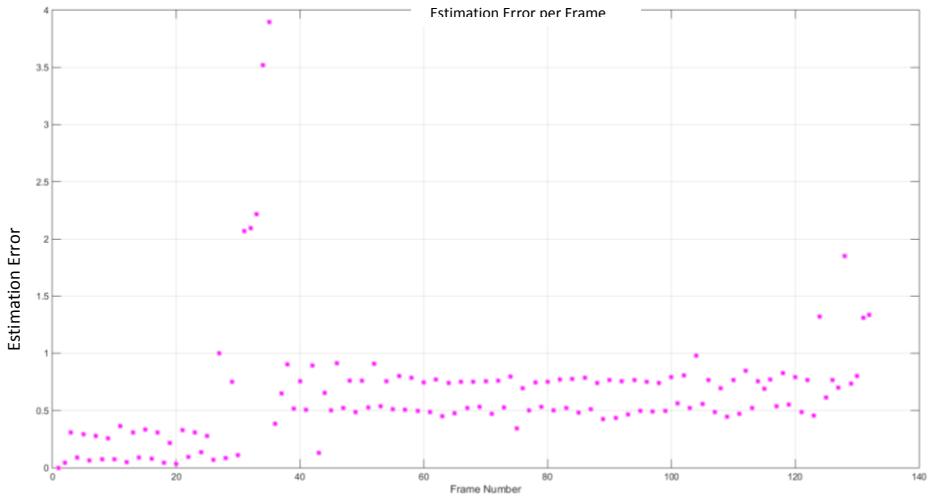
a. 28th frame of Input Image



b. 28th frame of background subtracted Input Image



c. Trajectory Estimation output of PF algorithm



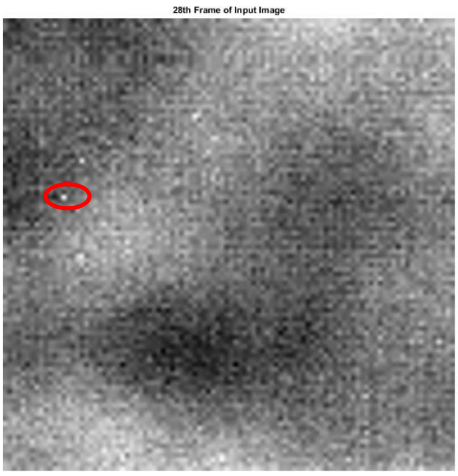
d. Estimation Error for each Frame

Figure 4.11 SENSOR TYPE-1 data for Motion 1 with target intensity level 20

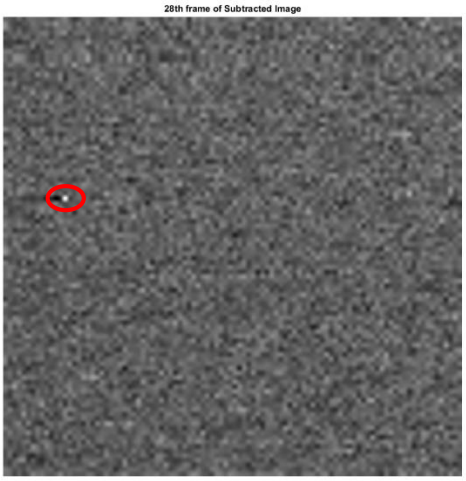
Figure 4.10 and *Figure 4.11* are the results obtained at intensity levels 30 and 20 for the SENSOR TYPE-1 image with Motion 1. Without background estimation, the algorithm mixed up the target with near background pixels. As given in 28^{th} frame of input image, there are many background pixels whose intensities are similar to that of the target. Thus background estimation is a crucial step at these SNR levels. In *Table 4.7*, the MSE of the intensity level 30 is still seen to be low, 0.17. But at the intensity level 20 it gets hard for the algorithm to track the target and the MSE rises to 0.7.

In *Figure 4.12* and *Figure 4.13* the results of SENSOR TYPE-1 with Motion 2 are shown. In the first few frames, the target can be estimated correctly. In the following frames the target is tracked successfully thanks to the accurate estimation of the B_k vector in the motion model, which is fed back to the PF algorithm. The velocity based weighting is not necessary and not used in these settings. In *Table 4.7* the MSEs are calculated as 0.02 and 0.16.

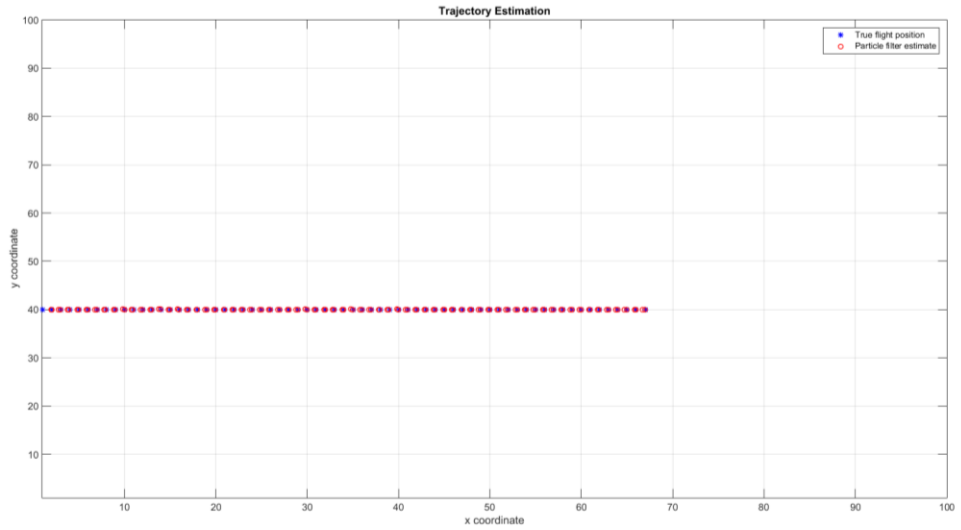
Figure 4.14, *Figure 4.15*, *Figure 4.16* and *Figure 4.17* present the results obtained under the same conditions for the SENSOR TYPE-2 image sequences. The MSE values corresponding to these figures are given in *Table 4.7* respectively as 0.05, 0.37, 0.02 and 0.17. As one can see, the MSE increases as the target intensity is reduced.



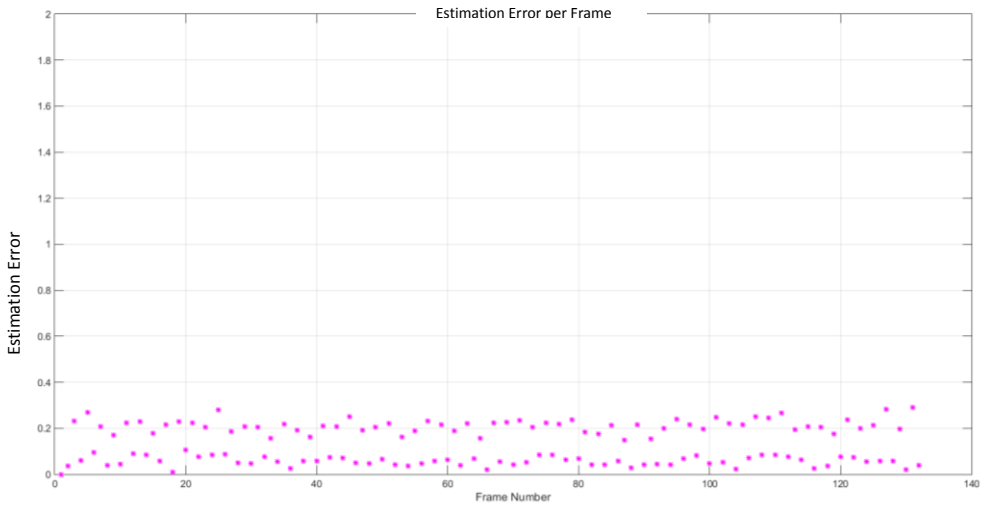
a. 28th frame of Input Image



b. 28th frame of background subtracted Input Image

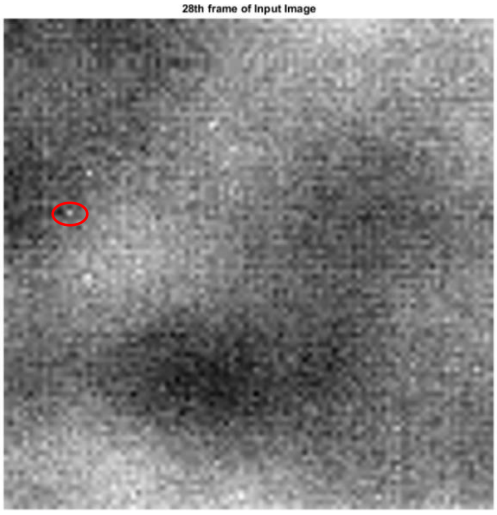


c. Trajectory Estimation output of PF algorithm

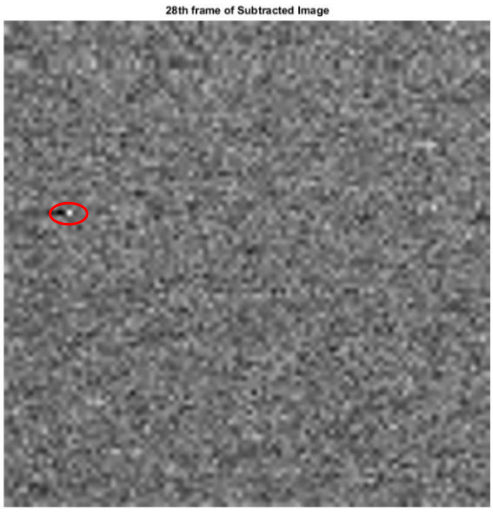


d. Estimation Error for each Frame

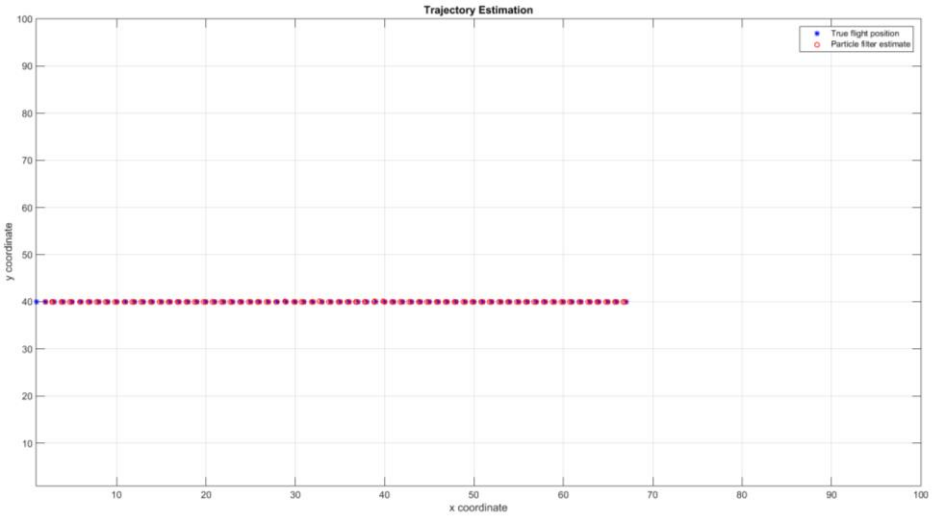
Figure 4.12 SENSOR TYPE-1 data for Motion 2 with target intensity level 30



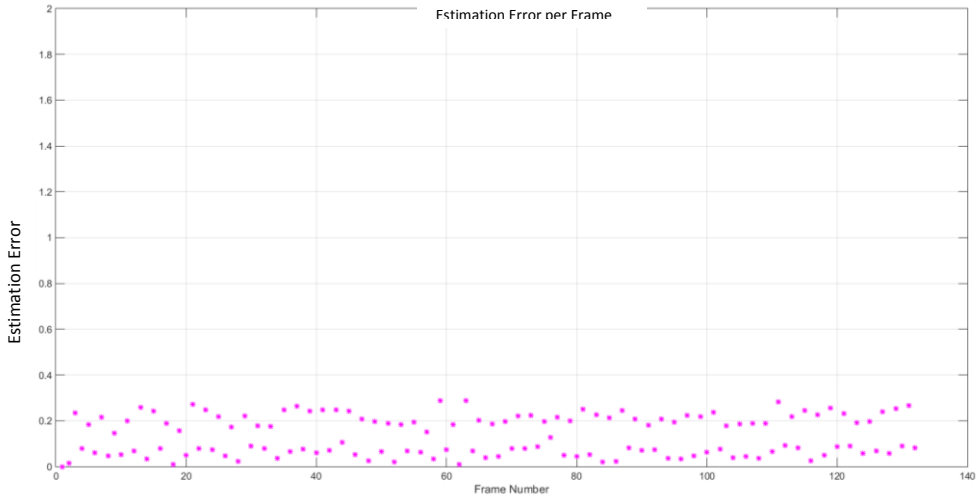
a. 28th frame of Input Image



b. 28th frame of background subtracted Input Image

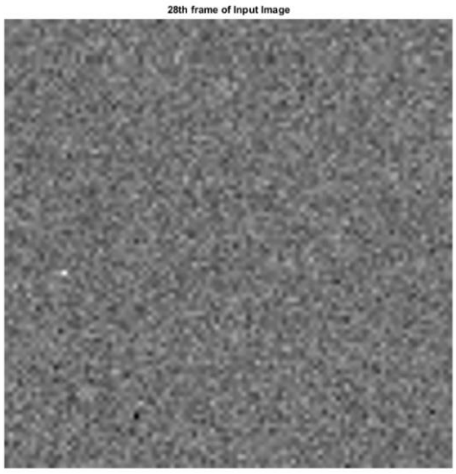


c. Trajectory Estimation output of PF algorithm

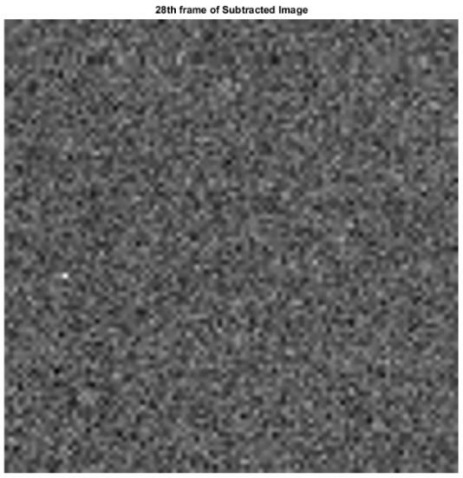


d. Estimation Error for each Frame

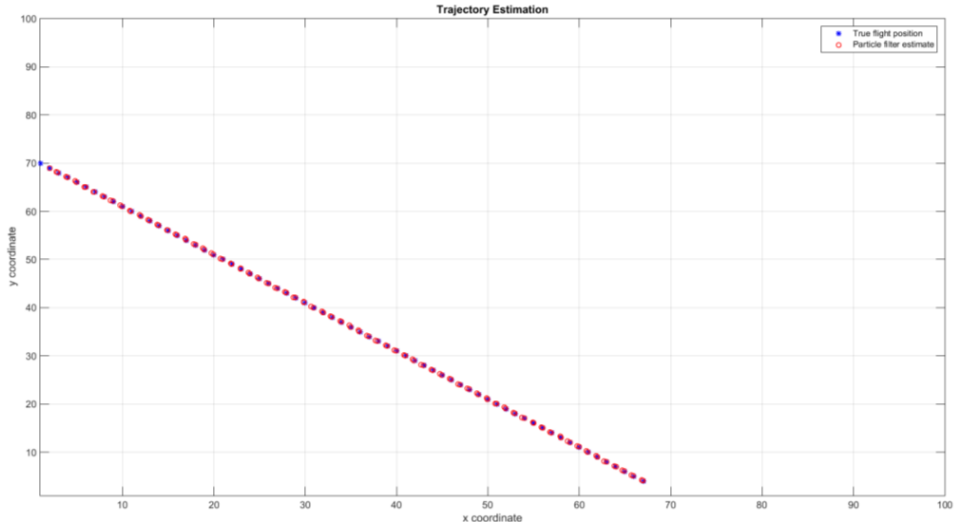
Figure 4.13 SENSOR TYPE-1 data for Motion 2 with target intensity level 20



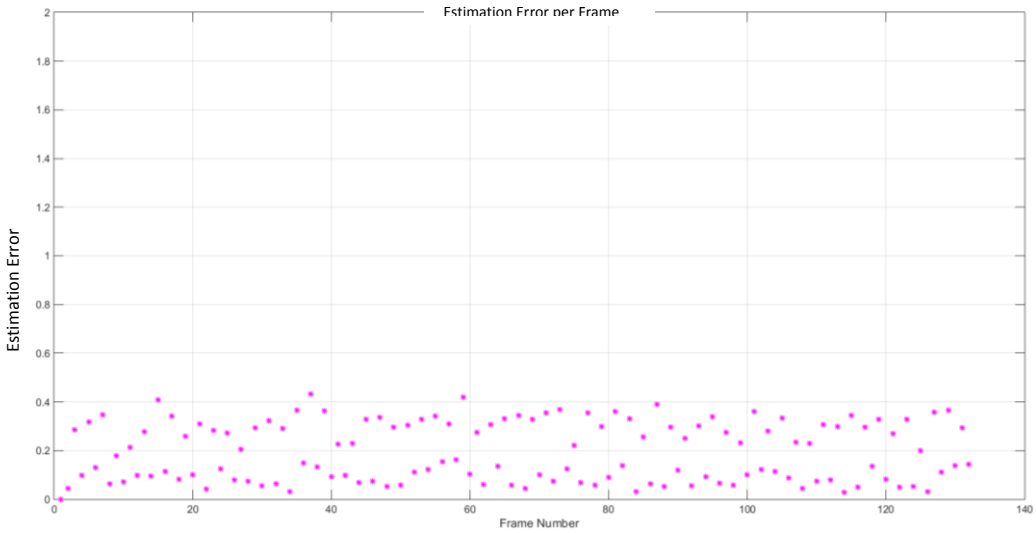
a. 28th frame of Input Image



b. 28th frame of background subtracted Input Image

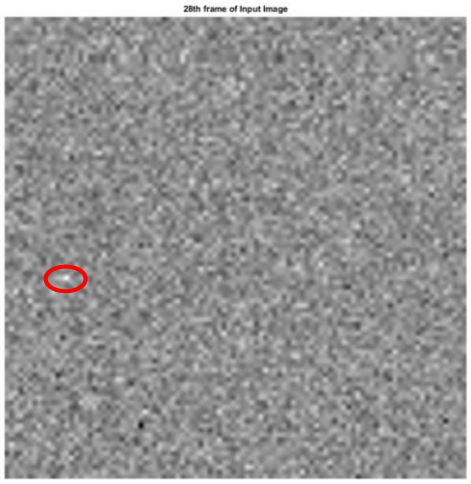


c. Trajectory Estimation output of PF algorithm

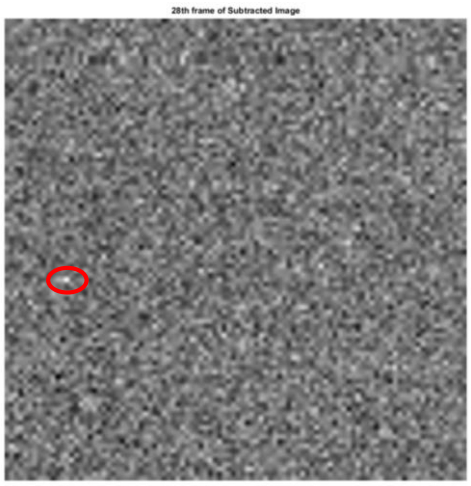


d. Estimation Error for each Frame

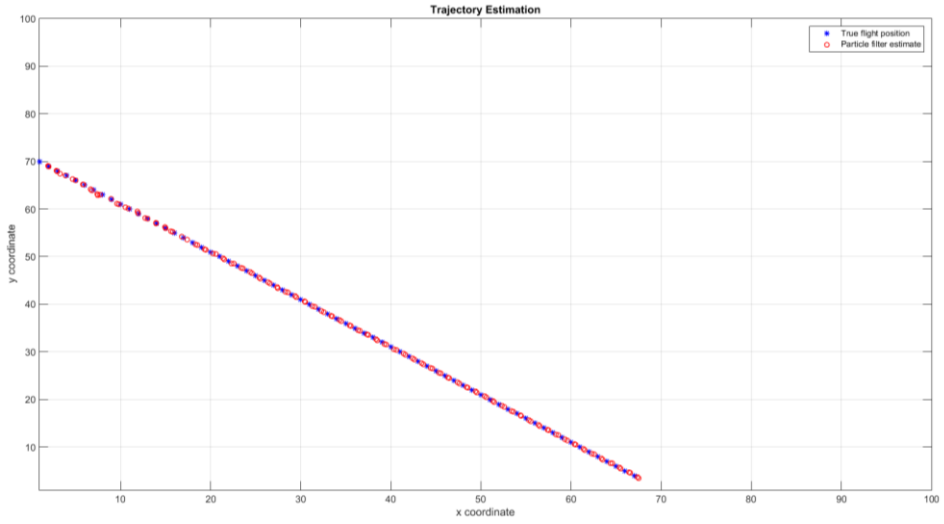
Figure 4.14 SENSOR TYPE-2 data for Motion 1 with target intensity level 30



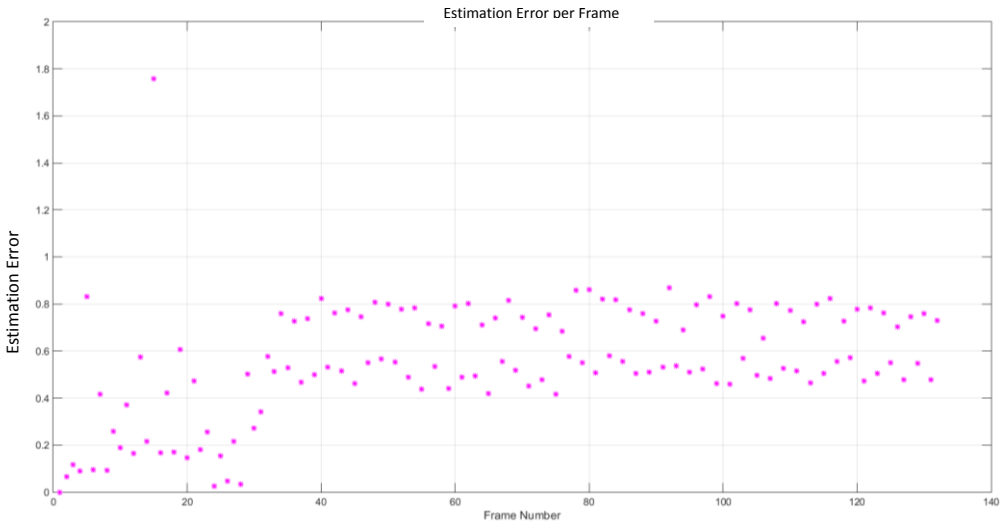
a. 28th frame of Input Image



b. 28th frame of background subtracted Input Image

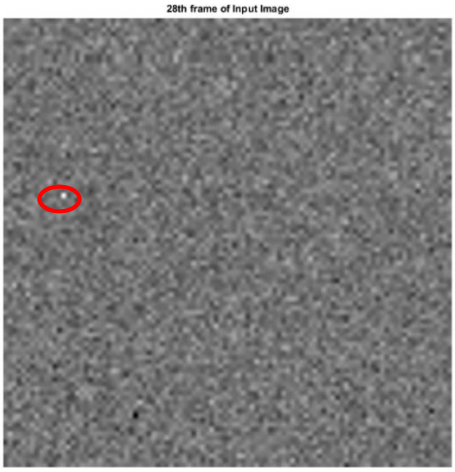


c. Trajectory Estimation output of PF algorithm

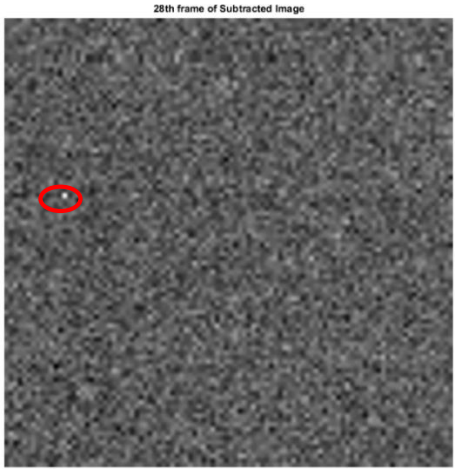


d. Estimation Error for each Frame

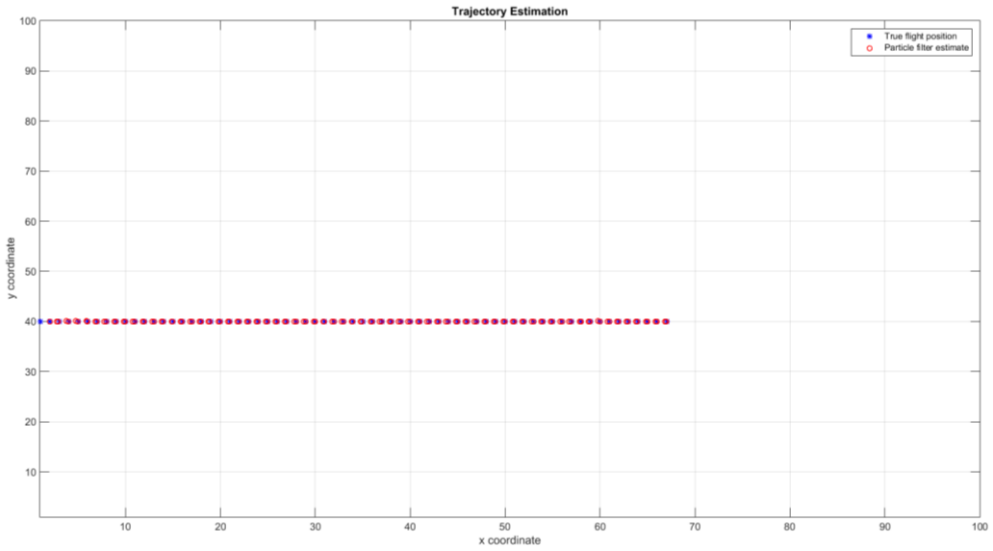
Figure 4.15 SENSOR TYPE-2 data for Motion 1 with target intensity level 20



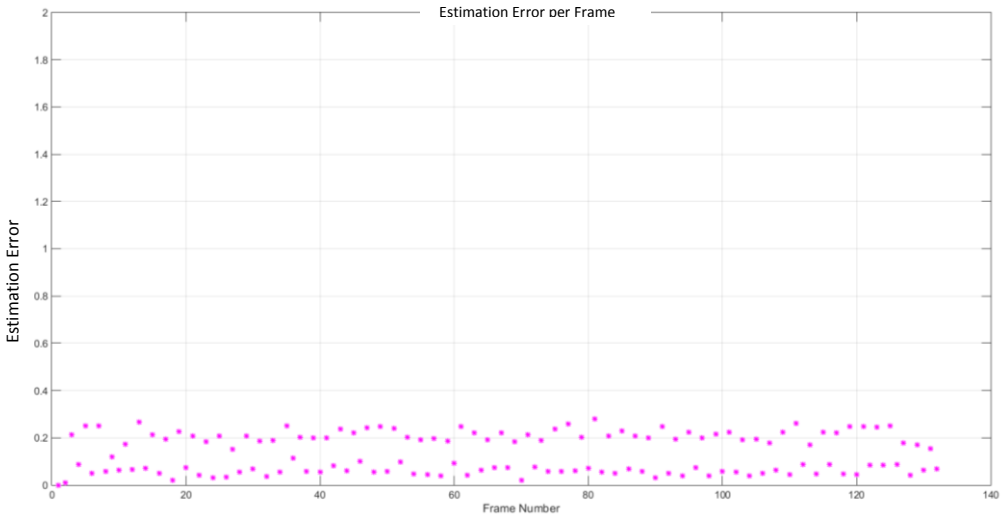
a. 28th frame of Input Image



b. 28th frame of background subtracted Input Image

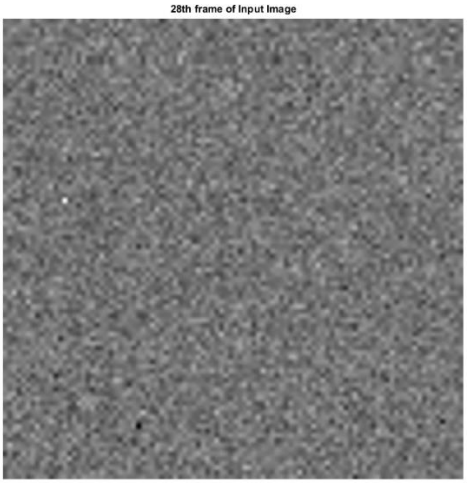


c. Trajectory Estimation output of PF algorithm

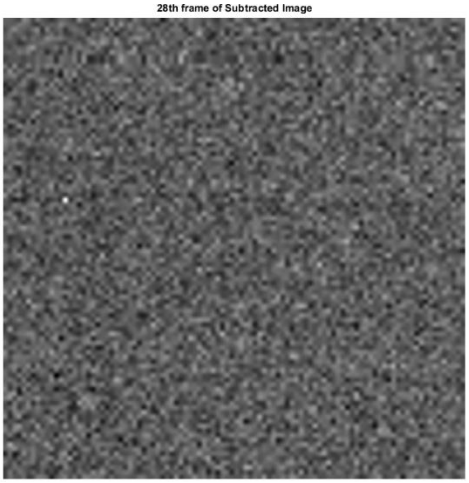


d. Estimation Error for each Frame

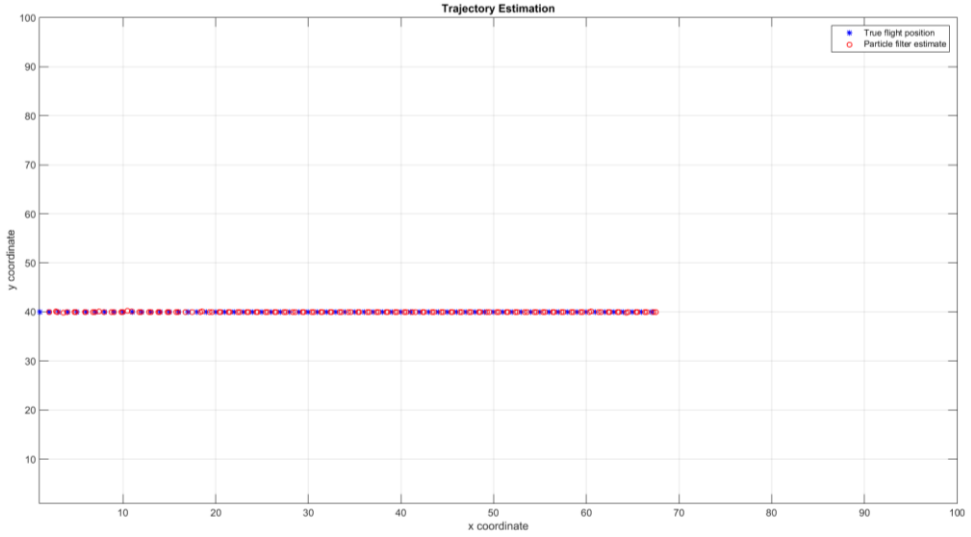
Figure 4.16 SENSOR TYPE-2 data for Motion 2 with target intensity level 30



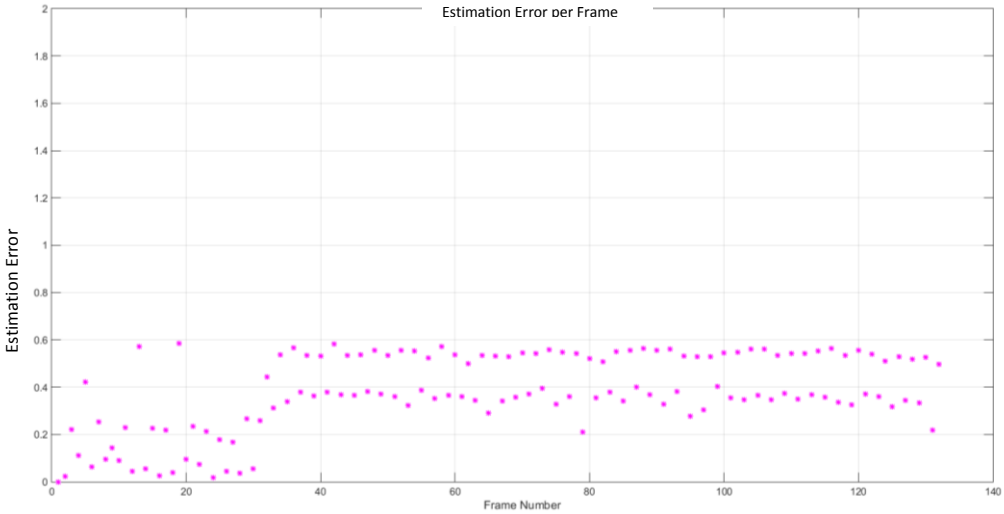
a. 28th frame of Input Image



b. 28th frame of background subtracted Input Image

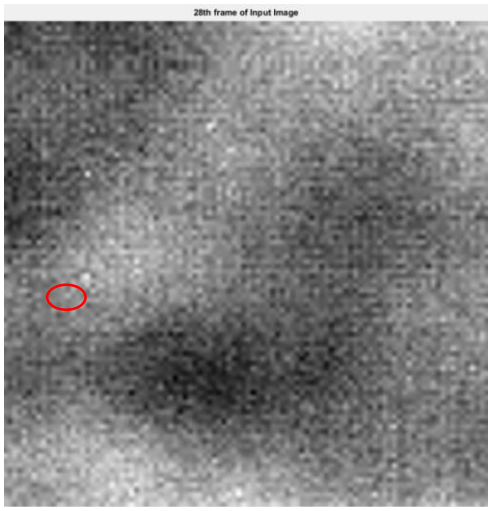


c. Trajectory Estimation output of PF algorithm

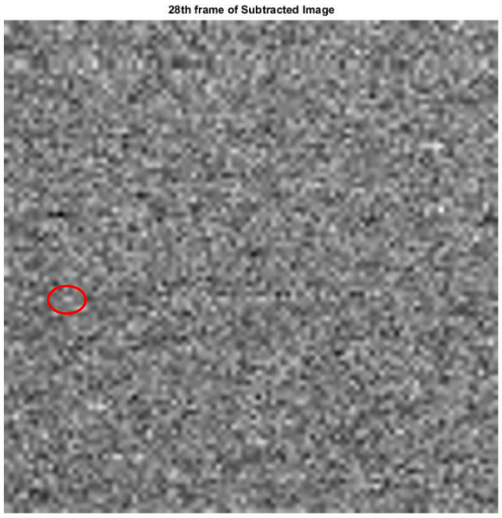


d. Estimation Error for each Frame

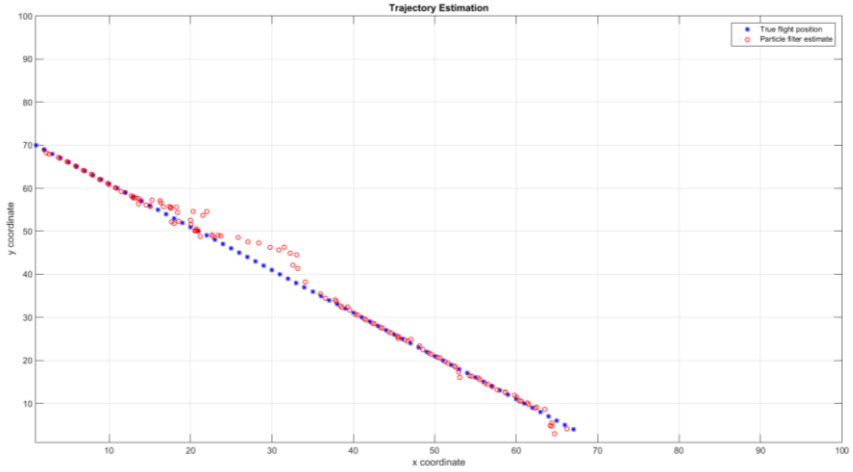
Figure 4.17 SENSOR TYPE-2 data for Motion 2 with target intensity level 20



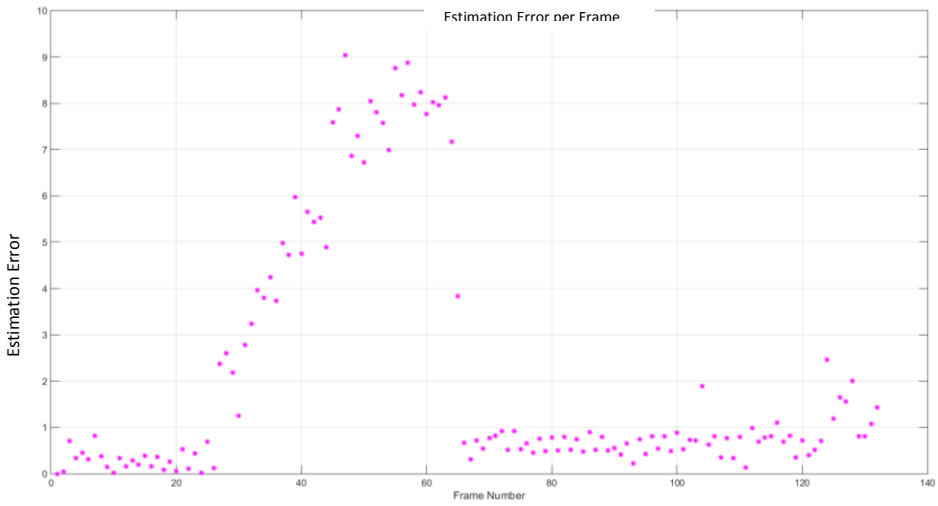
a. 28th frame of Input Image



b. 28th frame of background subtracted Input Image

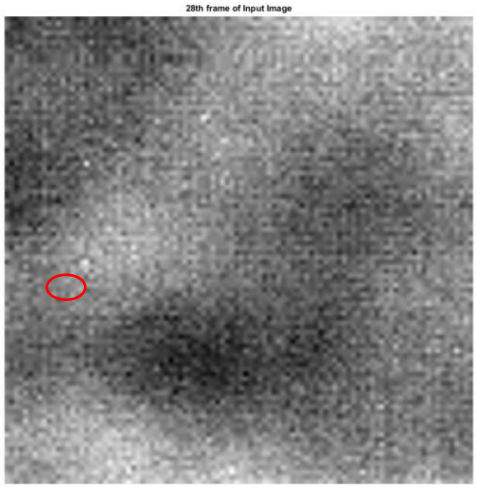


c. Trajectory Estimation output of PF algorithm

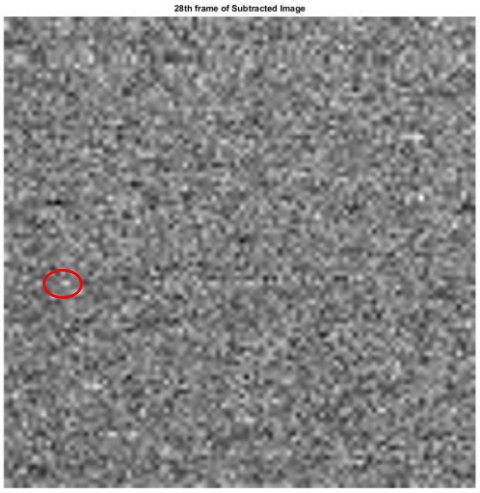


d. Estimation Error for each Frame

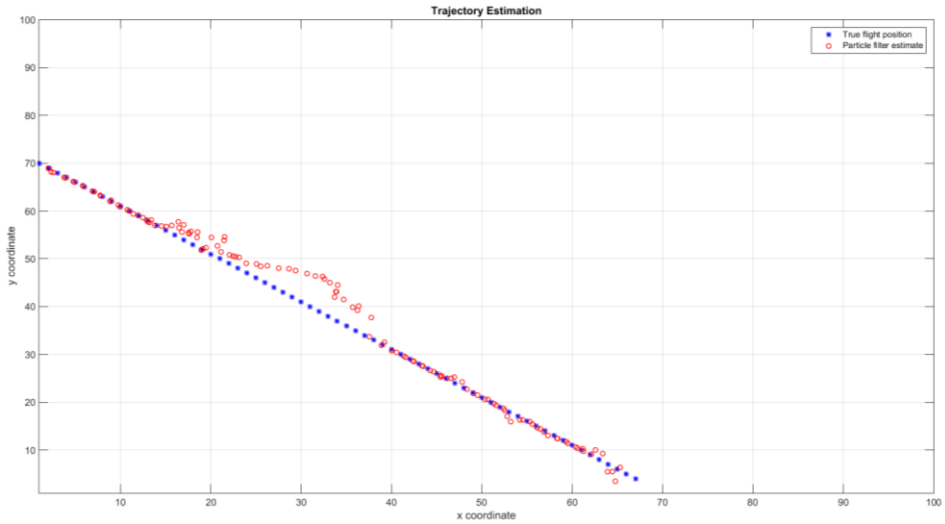
Figure 4.18 SENSOR TYPE-1 data for Motion 1 with target intensity level 15



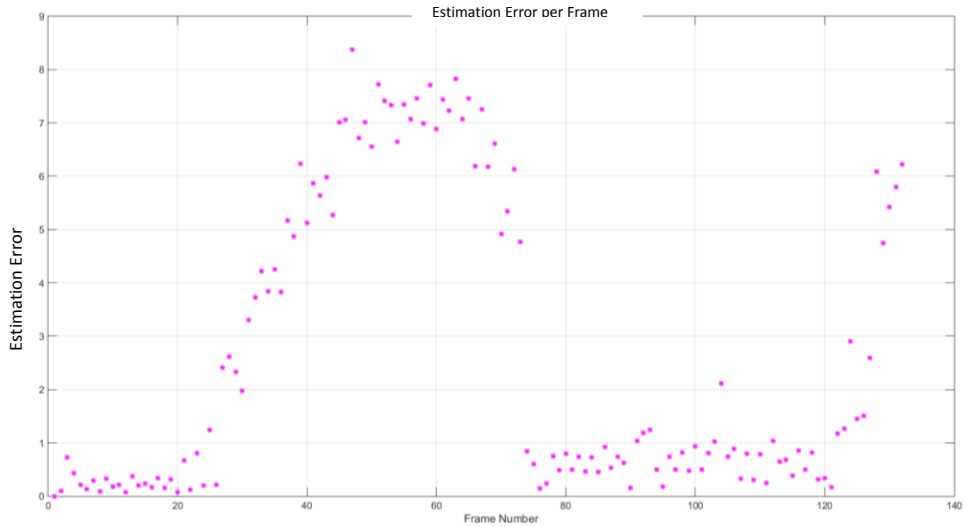
a. 28th frame of Input Image



b. 28th frame of background subtracted Input Image

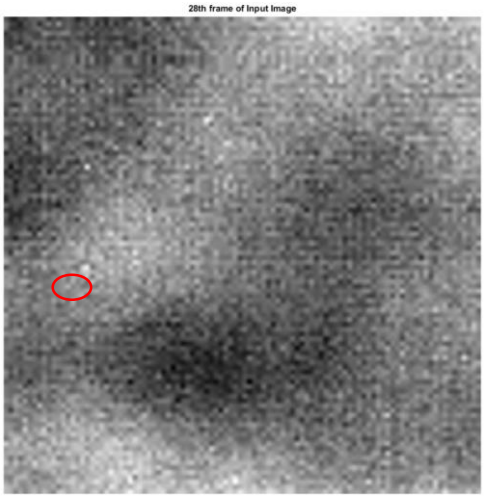


c. Trajectory Estimation output of PF algorithm

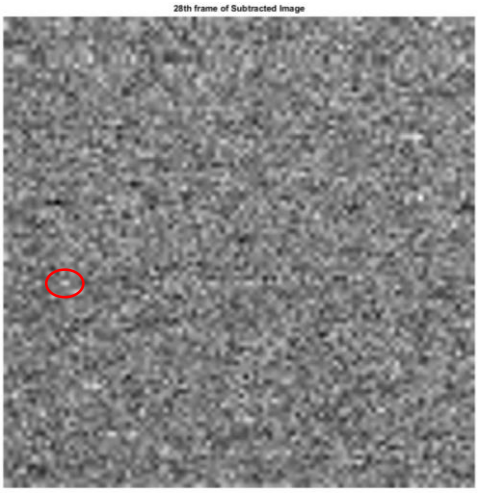


d. Estimation Error for each Frame

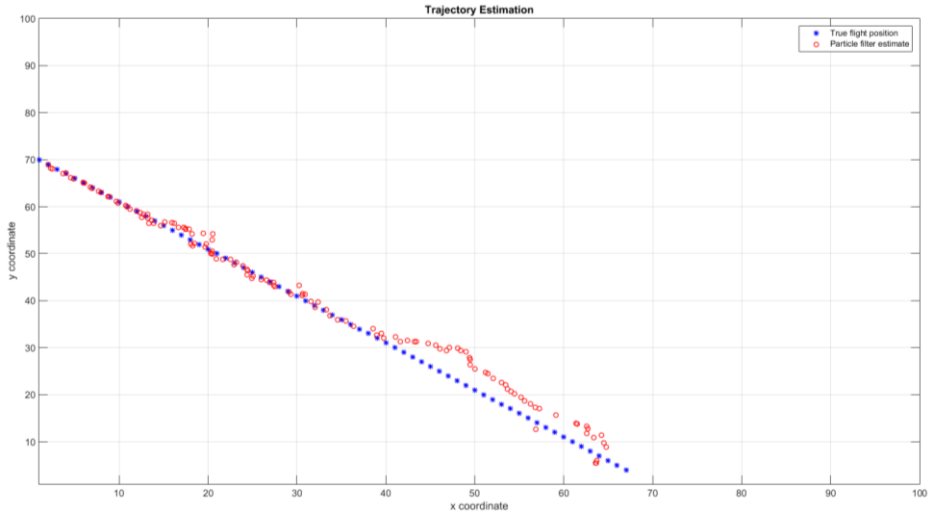
Figure 4.19 SENSOR TYPE-1 data for Motion 1 with target intensity level 14



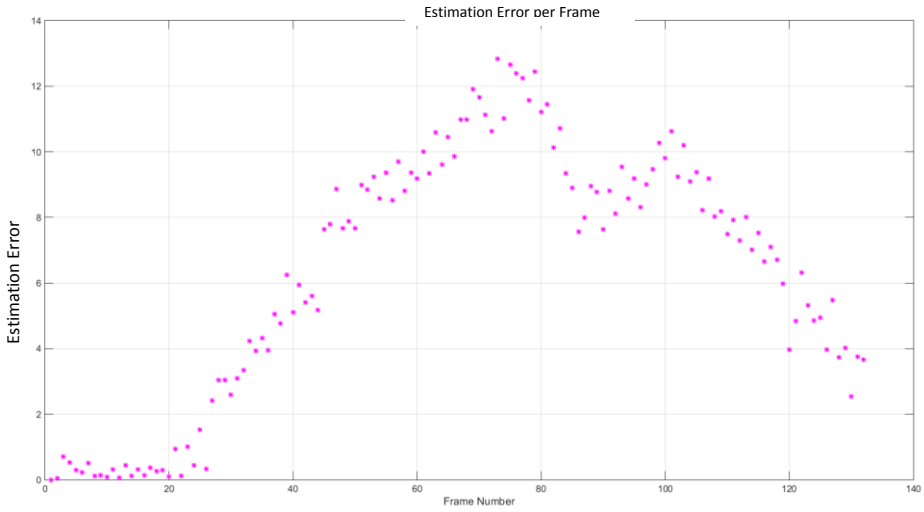
a. 28th frame of Input Image



b. 28th frame of background subtracted Input Image



c. Trajectory Estimation output of PF algorithm



d. Estimation Error for each Frame

Figure 4.20 SENSOR TYPE-1 data for Motion 1 with target intensity level 13

The intensity levels 15, 14 and 13 are the levels where the target cannot be seen by human eyes. The SNR is really low, thus the target intensity becomes quiet close to the background pixels. In these situations, updating B_k recursively, weighting the particles based also on the motion information and increasing the number of particles do really enhance the tracking performance of the algorithm.

As shown in *Figure 4.18* and *Figure 4.19*, even though the algorithm loses the target in the middle of the sequence, it is later capable of catching up and tracking the target until the end of the sequence. This is due to the inclusion of the velocity information in the combined weighting strategy.

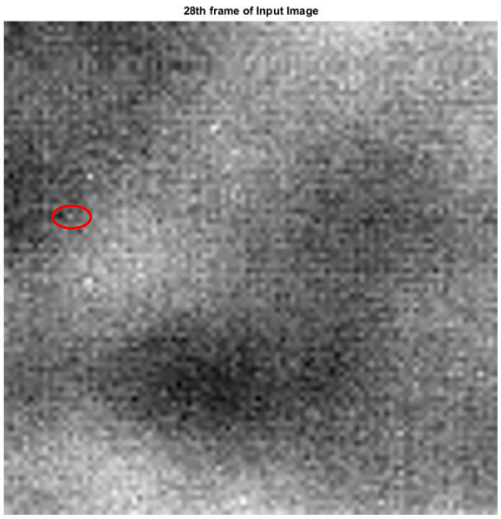
As one can see in *Figure 4.20*, it is impossible to see the target by human eyes even in the background subtracted image in this case. But the PF algorithm still can track the target in most of the frames.

Figure 4.21 presents the results obtained with the Motion 2 for the same case defined above. The target can easily be tracked in this low SNR case. The error level is given in *Table 4.8*.

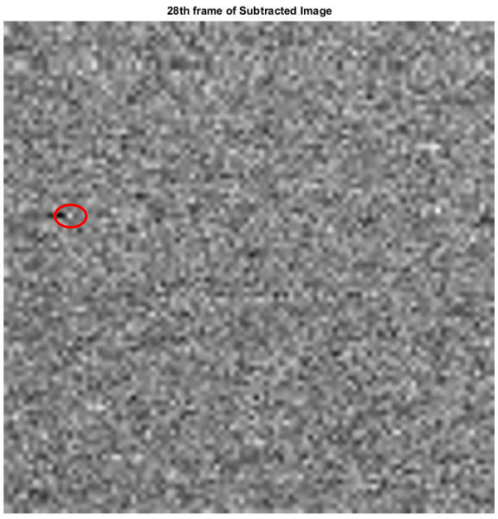
The performance difference in between Motion 1 and Motion 2 on the SENSOR TYPE-1 dataset is because of the movement of the target over different regions of the background pattern. In Motion 1, the target enters a quiet complex region of the background pattern and can be easily lost in the background pixels which have similar intensity values.

Figure 4.22, *Figure 4.23*, the low intensity level is observed to be tolerable by the algorithm and the target is tracked successfully.

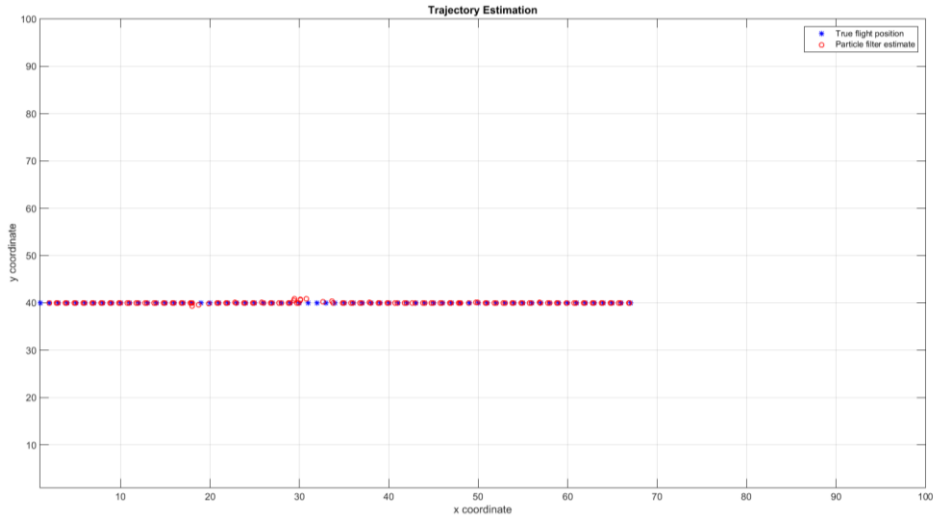
Lastly, the algorithm is tested on the FLIR sequence at intensity level 10. For the SENSOR TYPE-1 image with Motion 1 in *Figure 4.24*, the PF starts to track the target but then reaches its limits at the chosen parameter set (noise, particle size, etc.) and loses the track. On the other hand, Motion 2 still can still be tracked even at this level of SNR. The results obtained on the SENSOR TYPE-1 dataset for Motion 2 are given in *Figure 4.25*.



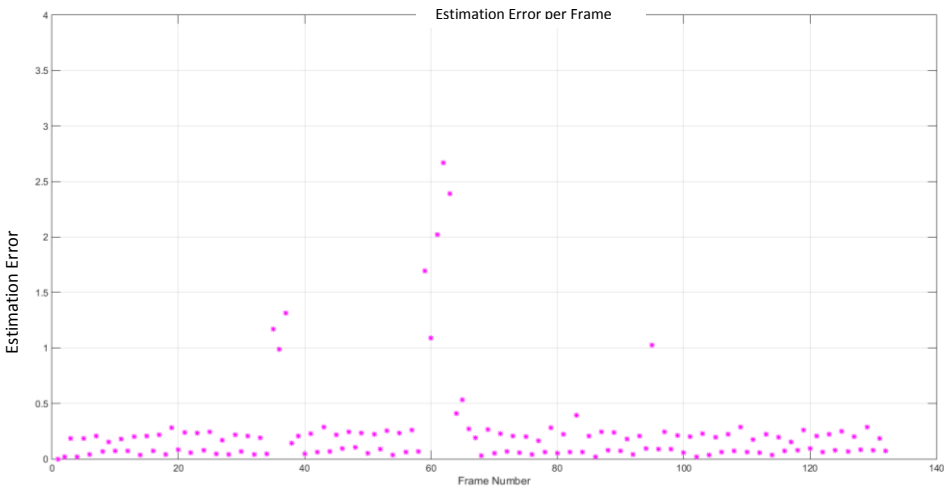
a. 28th frame of Input Image



b. 28th frame of background subtracted Input Image

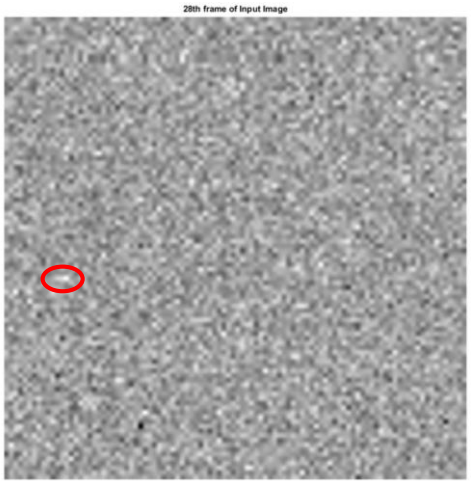


c. Trajectory Estimation output of PF algorithm

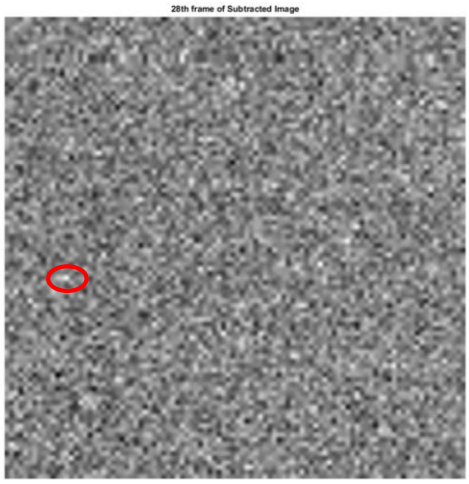


d. Estimation Error for each Frame

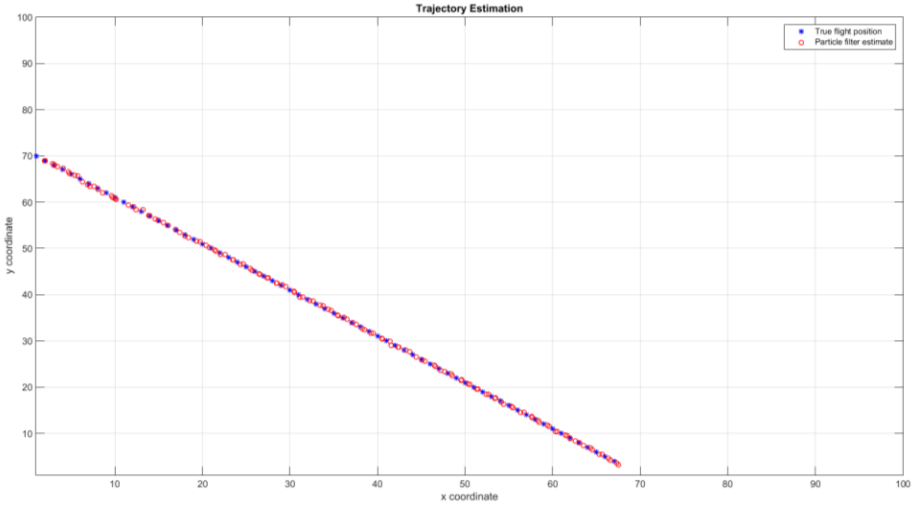
Figure 4.21 SENSOR TYPE-1 data for Motion 2 with target intensity level 15



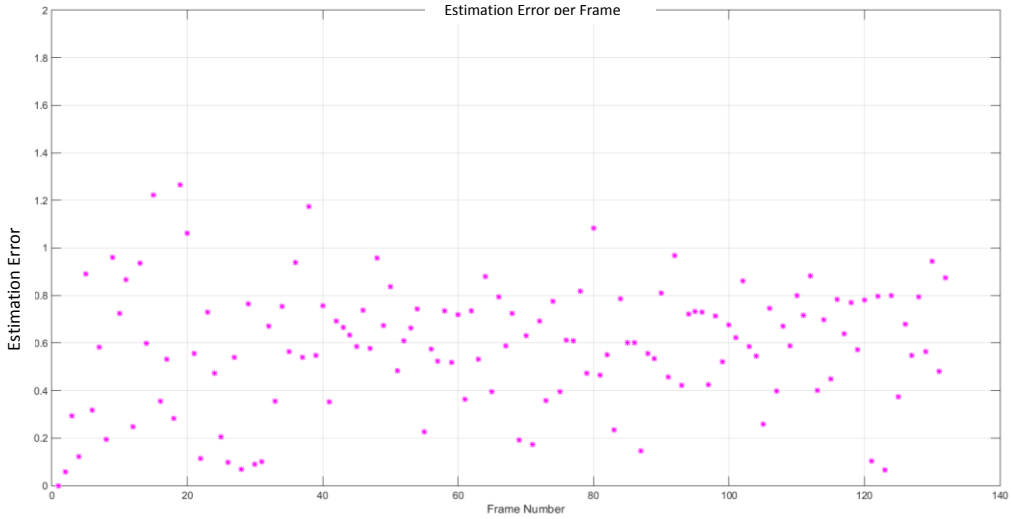
a. 28th frame of Input Image



b. 28th frame of background subtracted Input Image

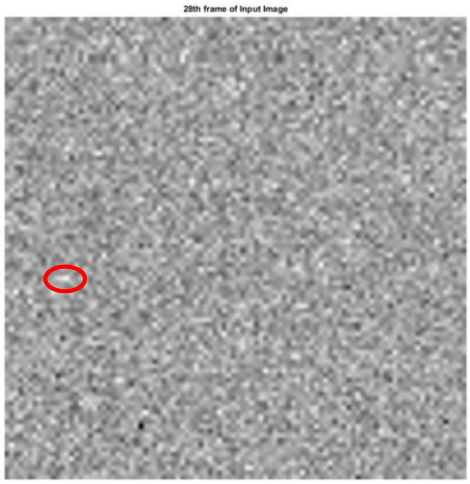


c. Trajectory Estimation output of PF algorithm

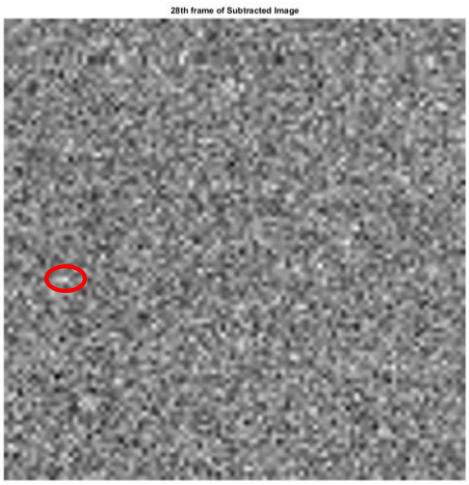


d. Estimation Error for each Frame

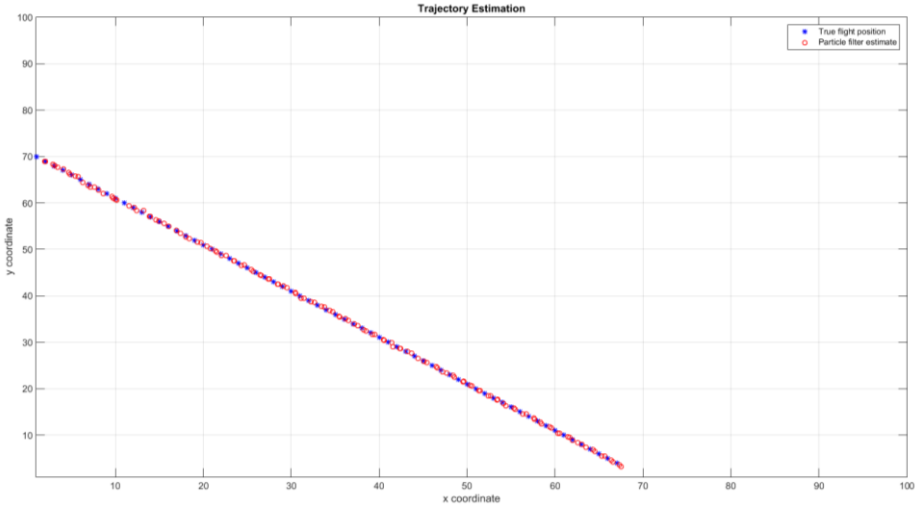
Figure 4.22 SENSOR TYPE-2 data for Motion 1 with target intensity level 15



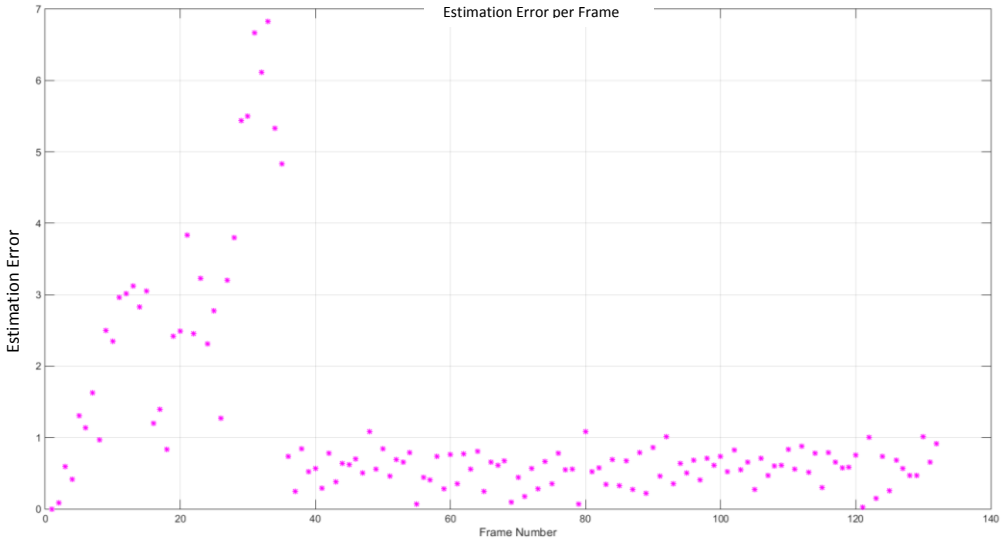
a. 28th frame of Input Image



b. 28th frame of background subtracted Input Image

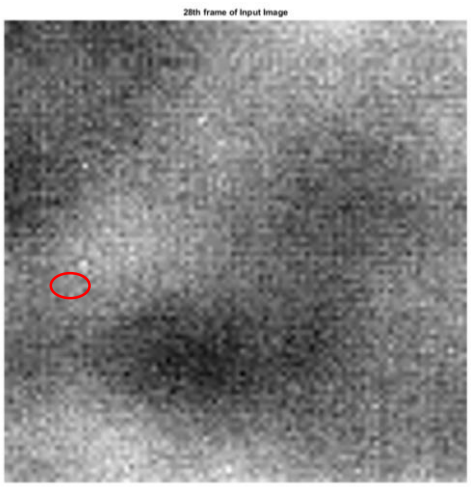


c. Trajectory Estimation output of PF algorithm

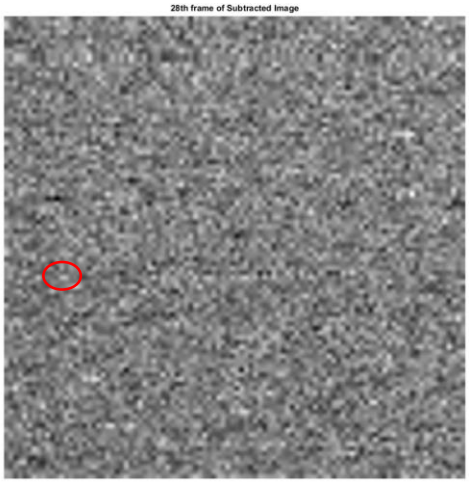


d. Estimation Error for each Frame

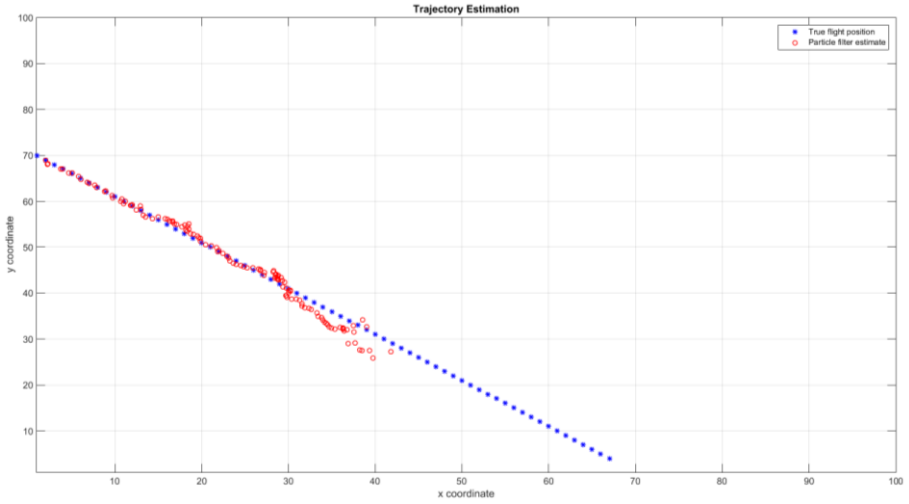
Figure 4.23 SENSOR TYPE-2 data for Motion 1 with target intensity level 14



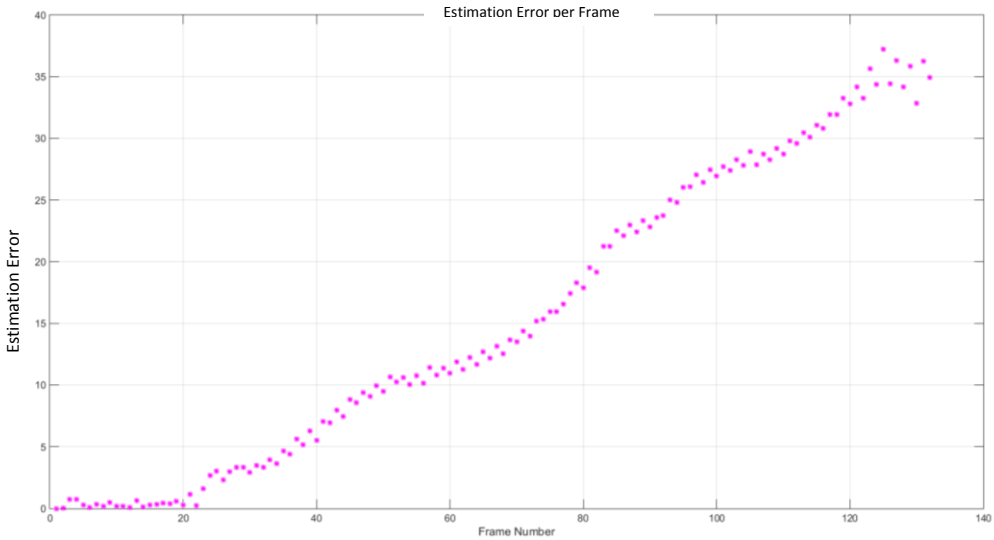
a. 28th frame of Input Image



b. 28th frame of background subtracted Input Image

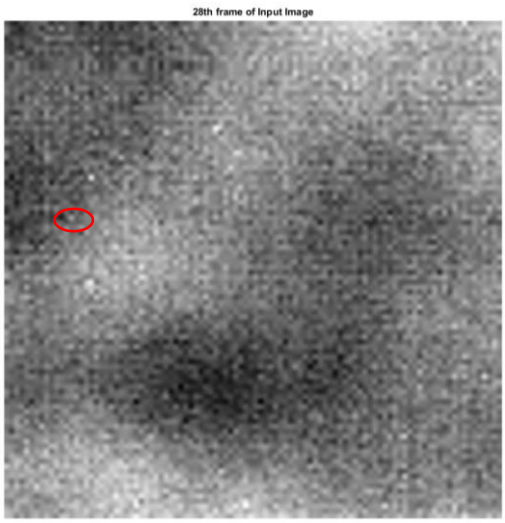


c. Trajectory Estimation output of PF algorithm

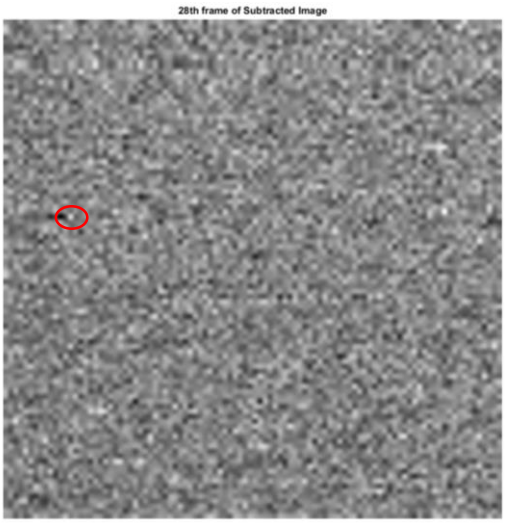


d. Estimation Error for each Frame

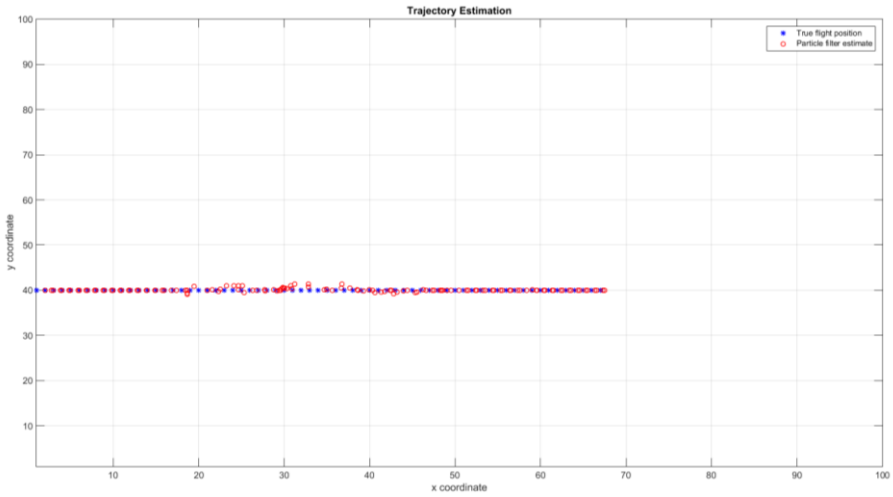
Figure 4.24 SENSOR TYPE-1 data for Motion 1 with target intensity level 10



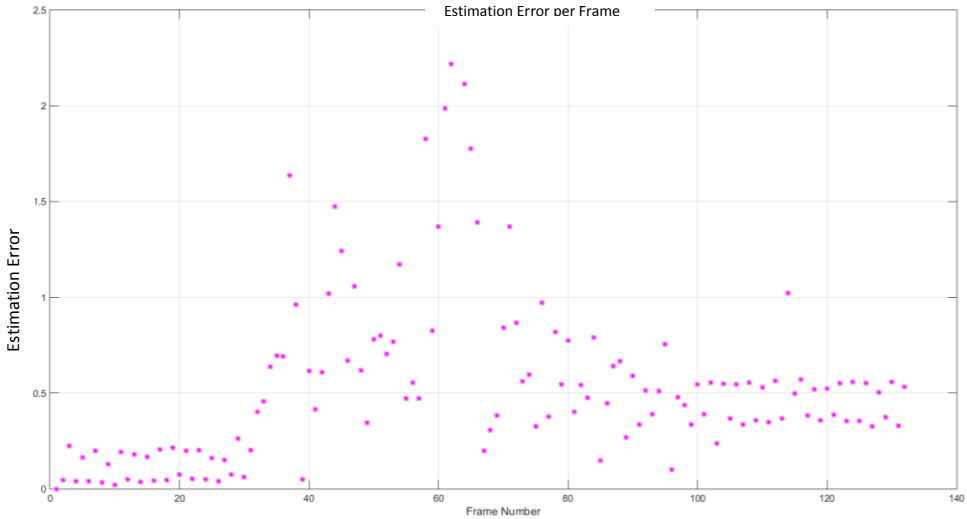
a. 28th frame of Input Image



b. 28th frame of background subtracted Input Image



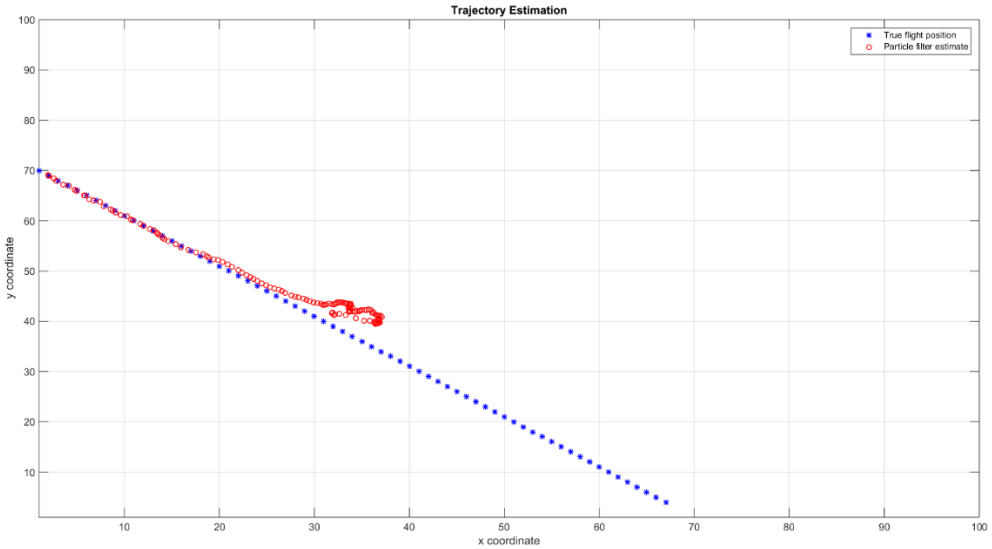
c. Trajectory Estimation output of PF algorithm



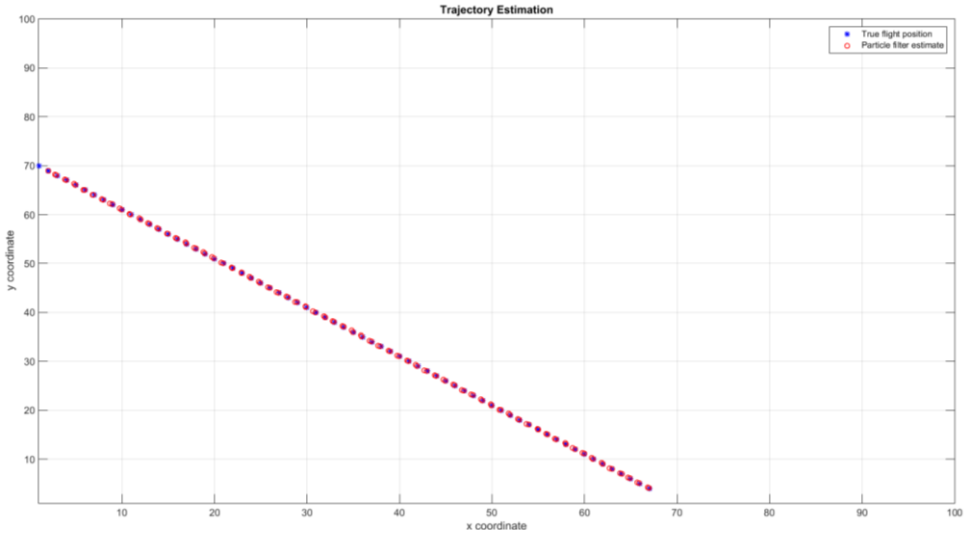
d. Estimation Error for each Frame

Figure 4.25 SENSOR TYPE-1 data for Motion 2 with target intensity level 10

The effect of motion based weighting process which was mentioned in 3.4.2.2.1 is given in below figures. SENSOR TYPE-1 data with intensity 30 is chosen for the comparison of hybrid and standard weighting of particles.



a. Trajectory Estimation output of PF algorithm with only intensity based weighting



b. Trajectory Estimation output of PF algorithm with combined weighting

Figure 4.26 Comparison of standard and combined weighting

As one can see from *Figure 4.26* intensity based weighting process in the PF algorithm fails when the SNR is low. On the other hand the combined weighting process enhances the accuracy of the tracking.

The effect of changing the particle population size is also analyzed. The algorithm is tested using SENSOR TYPE-1 image with target intensity 30 for varying population sizes. The MSE values are generated by simulating the same particle size for ten times and calculating the mean of the MSE. The values of MSE with respect to the particle size is given in *Table 4.9*. The results are also displayed in *Figure 4.27*, which is obtained by interpolating the data.

Table 4.9 MSE Distribution with respect to Particle Size

Particle Size	MSE
250	0,39605
500	0,36482
750	0,35549
1000	0,35629
1250	0,34512
1500	0,3459
1750	0,34662
2000	0,3514
2250	0,34502
2500	0,34423
2750	0,34776
3000	0,35508
3500	0,34867
4000	0,35218
5000	0,34457

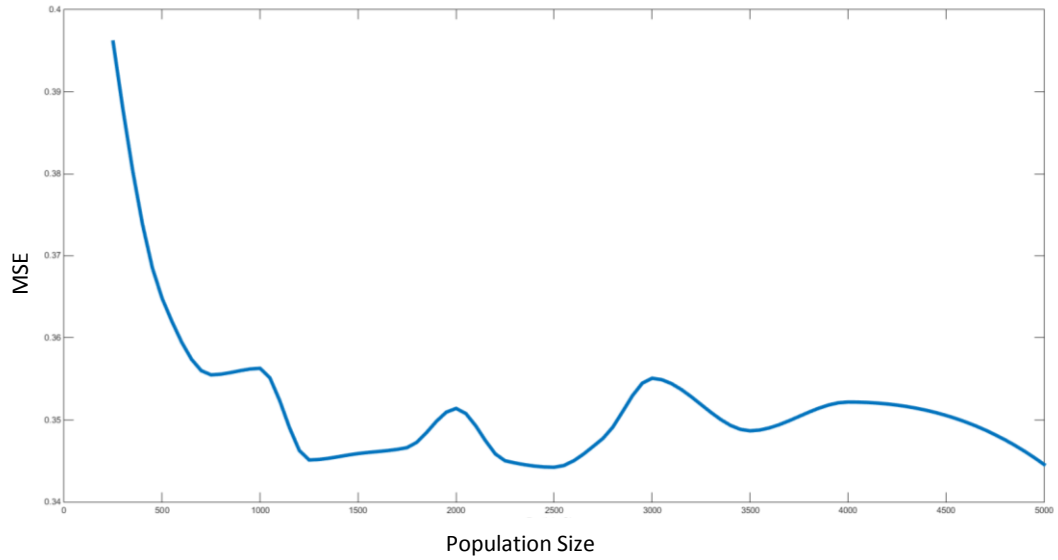


Figure 4.27 Effect of Particle Size on MSE

As one can see from the MSE values in *Table 4.9* increasing the number of particles improves the accuracy of the estimation. But above a level of population size, the error values become similar to each other.

Another parameter which should be analyzed is the vector B_k that is used in the motion model. B_k vector is updated as described in **Algorithm 6**, where *S*, represents the number of frames over which the average displacement is estimated to update B_k. The variation of the MSE with the number of frames is reported in *Table 4.10* and *Figure 4.27*.

Table 4.10 The effect of B vector generation on MSE

S, # of frames	**MSE for Motion Model 2**
3	0,1667
5	0,1641
10	0,1598
15	0,1499
20	0,1426
25	0,1356

30	0,1381
35	0,1229
40	0,1190
45	0,1089
50	0,0985

The B_k vector is used to estimate and update the motion model based on the target's past motion. The choice of the motion model 2 is due to the reason that the algorithm is less affected by the background pattern and the tracking is more accurate than Motion model 1 at the same intensity. Simulation results show that, in motion model 2, increasing the number of frames to estimate B_k , results in a decrease in the MSE. The reason of the improvement of the algorithm performance due to the increase in the number of frames to calculate the B_k vector is because of the linearity of the target motion model chosen in these experiments. Since the true B_k vector does not change throughout the frames its estimate gets better when a larger number of frames is used. Another motion model, like a curvy trajectory with a non-constant velocity vector would probably not favor a very large number of frames due to the change in the velocity.

We have also observed that for a lower MSE in motion model 2, as the number of frames for updating B_k is increased, a larger number of particles is needed. It can be explained as follows: A smaller population size causes an increase in the MSE of the trajectory, so the estimated states that are used for the calculation of B_k contain some error. Thus updating B_k over a larger number of frames results in an error accumulation. The accuracy of the estimation of B_k improves as the number of frames increases as in the previous case.

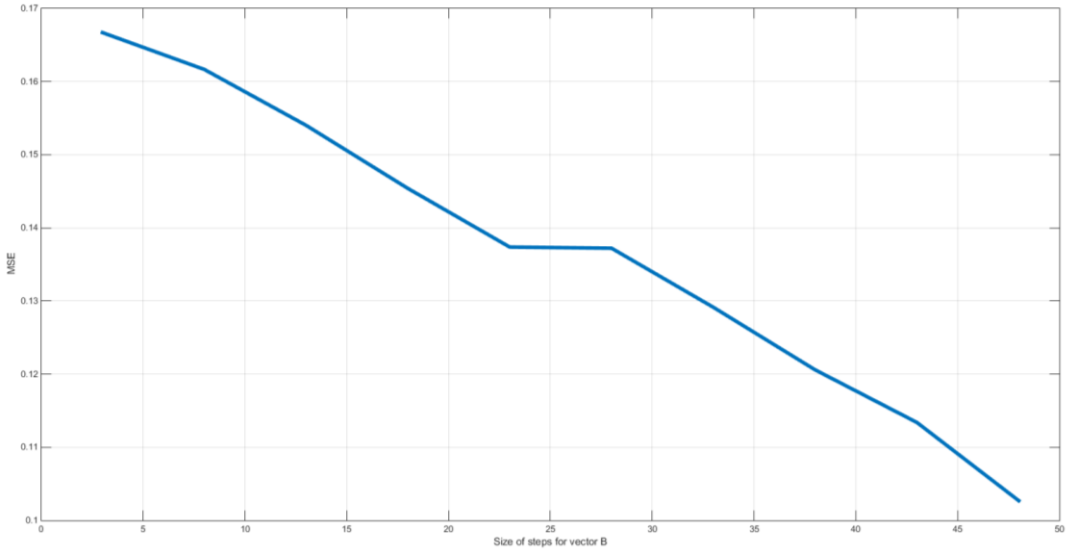


Figure 4.28 The effect of B_k vector generation on MSE

The next parameter to analyze in detail, is the measurement noise in the motion model, *C*. The algorithm is tested for the different *C* values given in *Table 4.11*.

Table 4.11 Noise coefficient effect on MSE

Noise Coefficient C	**MSE**
0,005	No track
0,05	No track
0,1	No track
0,5	0,34240
1	0,46484
1,5	0,86140
2	3,55750

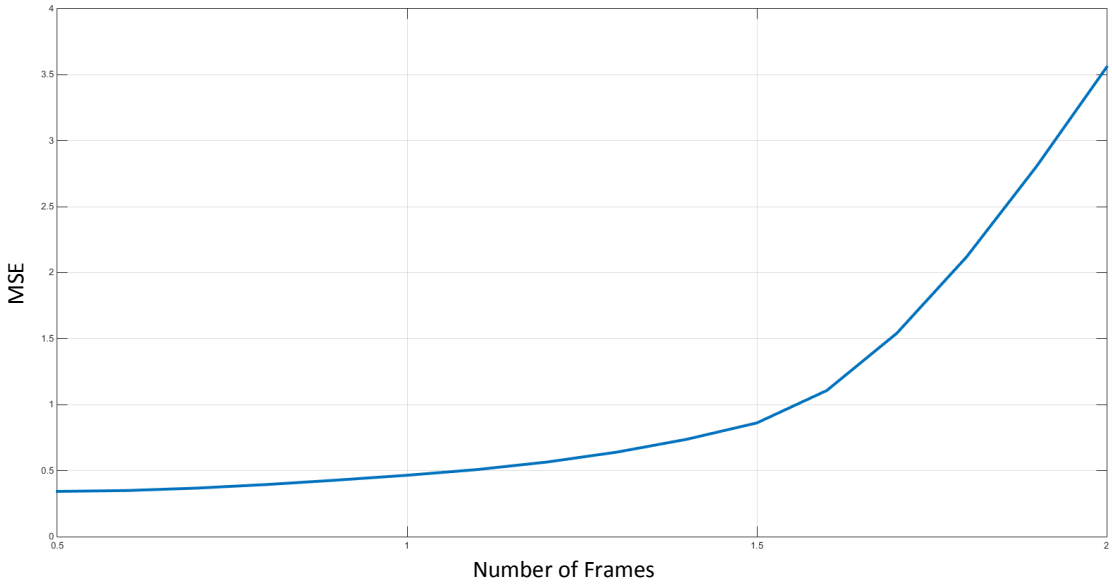


Figure 4.29 The effect of measurement noise in PF MSE

As seen in *Figure 4.29*, the increase in the noise coefficient, *C* in (3.2) results in an increase in the trajectory estimation, MSE. For lower values of *C*, the process noise in the state update equation is taken so small that the algorithm cannot distribute the particles within a sufficiently large region where the target probably exists. Hence, the target cannot be tracked as reported in *Table 4.11*. On the other hand, when *C* is larger, the process noise is assumed to be bigger and the area of the distributed particles increases but the target is mostly lost in this too large region.

To conclude, the experimental results presented in this chapter have led to the following findings. The proposed PF based tracking algorithm can track the target in really low SNR even if it is not visible to human eyes. However in situations where the SNR is very low, the algorithm may result in a high estimation error and it may even lose the target. It is observed that, the performance degradation in very low SNR sequences can be mitigated by including priors on the motional behavior of the target in the state transition model.

CHAPTER 5

CONCLUSION

In this study, a Particle Filter based tracking algorithm is offered to solve the problem of tracking dim point targets in IR image sequences with low SNR.

In the thesis, two IR image sequences generated from two different IR cameras are used and the target motion is embedded into the sequences synthetically. The performance of the algorithm is studied for different SNR values generated by changing the intensity of the target.

The proposed algorithm includes some approaches that enhance the performance and robustness of the algorithm in the low SNR cases. These approaches are listed below:

- The proposed algorithm is a TBD algorithm which uses preprocessing steps to reduce the number of candidates for tracking.
- The proposed algorithm is based on the Particle Filter where the particles are weighted using both intensity measurements and the estimated motion information.
- The proposed algorithm includes a state transition model which is updated with the past motion information that directs the particles towards the target's real coordinates.
- The proposed algorithm offers an approach for gradually eliminating the initial candidates since the number of candidates may be large, which increases the computational complexity. This approach is based on the motion consistency of each candidate point between its past and present motion.
- The tracking of the target in each frame is achieved by using the information of only previous frames. For this reason the algorithm is causal.

The results show that the Particle Filter tracking algorithm with the proposed approaches performs well in low SNR IR image sequences with a tolerable MSE. It is also shown that the background pattern affects the traceability of the target. In order to improve the performance of the algorithm especially in very low SNR data, we have adopted several approaches such as the inclusion of the velocity information in the PF weighting process and the adaptive estimation and update of the state transition model parameters. The experimental results suggest that these proposed strategies give promising results especially in data with challenging background patterns.

The performance of the algorithm may change under the conditions listed below;

- If the target has high velocity during its motion the algorithm may fail to track it. State model can be redesigned to solve this problem with including the velocity and acceleration changes.
- If the target can not be detected at the beginning of its motion, the algorithm couldn't be able to track it because it can only track candidate points at the beginning.
- If the target has an oncoming motion, the algorithm may fail to track it because it appears at the same pixel for a while before its size starts to increase.

5.1. FUTURE DIRECTIONS

Some future directions of our study are the following:

- Even though the algorithm is observed to track the target under very low SNR conditions the trajectory estimation error may be high. This error may be decreased by choosing a state transition model that includes velocity and acceleration information. Enriching the state transition model in this way may result in better tracking performance.
- Although the algorithm has not been tested on maneuvering targets, it can potentially handle this scenario due to the estimation and update of the target velocity information. The performance evaluation of the proposed tracking algorithm on maneuvering targets is an interesting future direction.

REFERENCES

[1] Yilmaz,A., Javed, O., Shah, M., "Object Tracking: A Survey",ACM Computing Surveys, Vol. 38, No. 4, Article 13, 2006.

[2] Rout, R. P., "A Survey on Object Detection and Tracking Algorithms",Department of Computer Science and Engineering,National Institute of Technology Rourkela,India,2013.

[3] Warren R. C. , "The Performance of Small Support Spatial and Temporal Filters for Dim Point Target Detection in IR Image Sequences", Weapons Systems Division Aeronautical and Maritime Research Laboratory, 2002.

[4] Abdo M. and Hongzuo L. , "New detection algorithm for dim point moving target in IR-image sequence based on an Image frames transformation", School of Electronics and Information Engineering, Changchun University of Science and Technology,Changchun,China.

[5] Xing, S., Ji, H., " A Track-Before-Detect Algorithm Based on Particle Filter with Model Estimation" ICSP '04., Signal Processing, 2004. Proceedings.

[6] Carlson,B.D., Evans,E.D., and Wilson,S.L., "Search Radar Detection And Track With The Hough Transform". IEEE Transaction on Aerospace and Electronic Systems, Jan. 1994.

[7] Barniv, Y. , "Dynamic programming solution for detecting dim moving targets,'' IEEE Transactions on Aerospace and Electronic Systems,AES-21 (Jan. 1985), 144-156.

[8] Yong H., Guo-feng,J., Kai-lan, Q., Chang-wen, Q., "Radar Track-Before-Detect Algorithm of Multitarget Based on the Dynamic Programming", Radar, CIE'06. International Conference on Digital Object Identifier:10.1109/ICR.2006.

[9] Rago, C. , P. Willett, et al. (1995). A comparison of the JPDAF and PMHT tracking algorithms. Acoustics, Speech, and Signal Processing, 1995. ICASSP-95. , 1995 International Conference on.

[10] Wei,L.,Zhang,X.,Fan,L., “A TBD Algorithm Based On Improved Randomized Hough Transform for Dim Target Detection”,2nd International Conference on Signal Processing Systems (ICSPS),2010.

[11] Wang,K., Liu, Y. , “Small Moving Infrared Target Detection Algorithm under Low SNR Background”, 2009 Fifth International Conference on Information Assurance and Security, IEEE.

[12] Zhang, F. , Li, C. , Shi, L. ,“Detecting and tracking dim moving point target in IR image sequence”, Department of Physics, Wuhan University, Wuhan, 430072, China, 2004.

[13] Li, Y.,Liang,S., Bai, B., Feng, D., “Detecting and tracking dim small targets in infrared image sequences under complex backgrounds”,Springer Science and Business Media,2012.

[14] He, L., Xie, L, Xie, T. Pan, H., Zheng, Y.,“An Effective TBD Algorithm for the Detection of Infrared Dim-Small Moving Target in the Sky Scene” Center for Engineering & Scientific Computation, and School of Aeronautics and Astronautics, 2012.

[15] Zhang,Y. , Su, X., Ma, P. “Multi-Hough Transform Track Initiation for Detecting Target with Constant Acceleration”, International Symposium on Information Science and Engieering,2008.

[16] Rotman, S.R., Nichtern, O., “Tracking of a Point Target in an IR Sequence using Dynamic Programming Approach”, IEEE 24th Convention of Electrical and Electronics Engineers in Israel,2006.

[17] Huang, D., Xue, A., Guo, Y., “A Particle Filter Track-before-detect Algorithm for Multi-Radar System”, Elektronika Ir Elektrotechiıka, ISSN 1392-1215, VOL. 19, NO. 5, 2013.

[18] Jia,L., Li, M., Xing, L., Wu, Y., Song, W.,“An Improved Particle Filter For Dim Radar Target Detection And Tracking”, National Key Lab of Radar Signal Processing, Xidian University and Electronic Engineer Institute of Xi’an, Shaanxi, China .

[19] Arulampalam, M.S., Maskell,S., Gordon, N., Clapp, T., “A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking”, IEEE TRANSACTIONS ON SIGNAL PROCESSING, VOL. 50, NO. 2, 2002.

[20] Orhan, E. “Particle Filtering”,2012.

[21] Tang,J., Gao, X and Jin,G., “Dim and Weak Target Detection Technology Based on Multi-Characteristic Fusion”, Proceedings of the 26th Conference of Spacecraft TT&C Technology in China, Chapter 27.3.1, Springer, 2013.

[22] Zeng, M., Li, J., Peng, Z. ,“The design of Top-Hat morphological filter and application to infrared target detection”, Institute of Information and Control, School of Information and Control, Shanghai Jiao Tong University, Elsevier,2005.

[23] H. Askar, Z. Fu, Z. Li, “Detecting dim moving point targets using transformation of pixel statistics”, IEEE International conference on Communications, Circuits and Systems and West Sino Expositions, 2002.

[24] Qu, X., Zuo, Z., Li, X., “A novel algorithm for small dim target detection in distorted infrared image sequence” Institute for Pattern Recognition and Artificial Intelligence,Huazhong University of Science and Technology, China, SPIE, 2009.

[25] Mukherjee, A. , Kanrar, S. , “Enhancement of Image Resolution by Binarization”, International Journal of Computer Applications (0975 – 8887), Volume 10– No.10, November 2010.

[26] Strens, M. J. A., Gregory, I. N., “Tracking in cluttered images”, Image and Vision Computing 21 891–911, Elsevier, 2003.

[27] Liu, D., Li, Z., “Moving target detection by nonlinear adaptive filtering on temporal profiles in infrared image sequences”, School of Physics and Optoelectronic Engineering, Xidian University, Infrared Physics & Technology, Elsevier, 2015.

[28] Chen, Z., Song, L., Xie, T., Liu, J., Wang, G., Lei, G., “A novel infrared small target detection method based on BEMD and local inverse entropy”, Infrared Physics & Technology, Elsevier, 2014.

[29] Long, Y., Xu, H., An, W., Liu, L., “Track-before-detect for Infrared Maneuvering Dim Multi-target via MM-PHD”, College of Electronic Science and Engineering, National University of Defense Technology, China, Chinese Journal of Aeronautics, Elsevier, 2012.

[30] Lei, L., Zhijian, H., “Improved motion information-based infrared dim target tracking algorithms”, College of Electronic Engineering and Photoelectric Technology, Nanjing University of Science and Technology, China, Infrared Physics & Technology, Elsevier, 2014.

[31] Hlinomaz, P., Hong, L., “A multi-rate multiple model track-before-detect particle filter”, Department of Electrical Engineering, Wright State University, Dayton, United States, Mathematical and Computer Modelling, Elsevier, 2008.

[32] Buzzi S, Lops M, Venturino L. “Track-Before-Detect Procedures for Early Detection of Moving Target from Airborne Radars”, IEEE Transactions on Aerospace and Electronic Systems 2005; 41(3): 937-954, IEEE, 2005.

[33] Qingbo, J., Yang, Y., “The Arithmetic of Tracking before Detecting of Dim Infrared Targets Based on Particle Filter”, Information and Communication Engineering College Harbin Engineering University and School of Automation Science and Electrical Engineering, Beihang University, China, Microelectronics and Electronics (PrimeAsia), IEEE, 2010.

[34] Hamdulla, A., Xiaofeng, L., Zaiming, “Performance Analysis of Dim Moving Point Target Detection Algorithms”, College of communication and information Engineering, University of Electronic Science and Technology of China, China, IEEE, 2002.

[35] Davey, S. J., Rutten, M. G., “A Comparison of Three Algorithms for Tracking Dim Targets”, Intelligence, Surveillance and Reconnaissance Division Defence Science and Technology Organisation PO Box 1500, Edinburgh, SA 5111, AUSTRALIA, Information, Decision and Control, IEEE, 2007.

[36] Moshtagh, N., Romberg, P. M., Chan, M. W.,“Multisensor Fusion for 3D Target Tracking Using Track-Before-Detect Particle Filter”, Advance Technology Center

Lockheed Martin Space Systems Company, Sunnyvale CA USA, Signal Processing, Sensor/Information Fusion, and Target Recognition XXIV, SPIE, 2015.

[37] ZHANG, Y., ZHANG, Q., XU, Z., XU, J., "Dim point target detection against bright background", Institute of Optics and Electronics, Chinese Academy of Sciences and Graduated School of Chinese Academy of Sciences, China, Real-Time Image and Video Processing, SPIE, 2010.

[38] Moshtagh, N., Romberg, P. M., Chan, M. W.,"Tracking Low SNR Targets Using Particle Filter with Flow Control", Advance Technology Center Lockheed Martin Space Systems Company, Sunnyvale CA USA, Signal and Data Processing of Small Targets, Proc. of SPIE, 2014.

[39] Lehmann, F., "Recursive Bayesian Filtering for Multitarget Track-Before-Detect in Passive Radars", IEEE TRANSACTIONS ON AEROSPACE AND ELECTRONIC SYSTEMS VOL. 48, NO. 3 July 2012, IEEE, 2012.

[40] Tursun, D., Guiying, X., Hamdulla, A., "A Particle Filter Based Algorithm for State estimation of Dim Moving Point Target in IR Image Sequence", Institute of Information Science and Engineering, Xinjiang University, Urumqi 830046, China, Second International Symposium on Intelligent Information Technology Application, IEEE, 2008.

[41] Shaonan, G., Xiaoling, J., "A Track-Before-Detect Algorithm based on KA-HT", UESTC, Chengdu, Radar Conference, 2009 IET International.

[42] Tian, Y., Gao, K., Liu, Y., Han, L., "A novel track-before-detect algorithm based on optimal nonlinear filtering for detecting and tracking infrared dim target", Key Lab of Photoelectronic Imaging Technology and System, Ministry of Education of China, Beijing Institute of Technology, and North China Institute of Optoelectronic Technology, Beijing , China, Proc. of SPIE Vol. 9622 96220U-1, 2015.

[43] Bai, X., Zhang, S., Du, B., Liu, Z., Jin, T., Xue, B., Zhou, F., "Survey on dim small target detection in clutter background: wavelet, inter-frame and filter based algorithms", Advanced in Control Engineering and Information Science, Elsevier, 2011.

[44] Li, H., Shaohua, X., Luoqing, L., “Dim target detection and tracking based on empirical mode decomposition”, Department of Mathematics, Huazhong University of Science and Technology, Faculty of Mathematics and Computer Science, Hubei University, China, Elsevier, 2008.

[45] Huang, K., Mao, X., “Detectability of infrared small targets”, Electronic and Information Engineering School, Beihang University, Beijing, China, Infrared Physics & Technology, Elsevier, 2009.

[46] Moyer, L. R., Spak, J., “A Multi-Dimensional Hough Transform-Based Track-Before-Detect Technique for Detecting Weak Targets in Strong Clutter Backgrounds”, IEEE TRANSACTIONS ON AEROSPACE AND ELECTRONIC SYSTEMS VOL. 47, NO. 4 October 2011, IEEE, 2011.