UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

A STATE VECTOR AUGMENTATION METHOD FOR INCLUDING VELOCITY INFORMATION IN THE LIKELIHOOD FUNCTION OF THE SIR VIDEO TARGET TRACKING FILTER

A THESIS

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

degree of

MASTER OF SCIENCE

By

JESYCA FUENMAYOR BELLO
Norman, Oklahoma
2016

A STATE VECTOR AUGMENTATION METHOD FOR INCLUDING VELOCITY INFORMATION IN THE LIKELIHOOD FUNCTION OF THE SIR VIDEO TARGET TRACKING FILTER

A THESIS APPROVED FOR THE
SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

BY

Dr. Joseph P. Havlicek, Chair

Dr. Ronald D. Barnes

Dr. Tian-You Yu

To Mimi and Pipa, who will always be with me.

Acknowledgements

I would like to express my gratitude to my committee members for serving and providing me the technical knowledge during my studies at the University of Oklahoma. Especially, I would like to honor my academic advisor, Dr. Joebob Havlicek. He has had a great impact in me, both as a professor and a guide. Thank you for the innumerable life lessons you gave me and for always believing in me.

To my parents, Jesus and Carmen Cecilia who have always believed in me, and have fought so fiercely so I could have not a good, but an excellent life. Thank you for teaching me the importance in discipline, perseverance and solidarity. You are my motors, this achievement is as mine as yours.

Thank you to my family in Venezuela and around the world for loving me and supporting me always. To Katherine, Alejandro and Alexandra, thank you for being the way you are and never letting me down. I could not ask for better siblings and a sister-in-law. I love you.

I have many friends I would like to thank for their support and caring love. To my amazing ITS Center group friends, thank you for receiving me with open arms and a smile. I have grown to be so much better as a person and professional in the past two years working with you and I will be forever grateful with each one of you. Thank you to my friends around the world who have managed to always make me feel their love regardless of the time and distance. To my friends here in Norman for making these two years an

amazing experience.

Last but not least, to Luis, who has supported me completely since the moment we met. Thank you for being my sunshine even in rainy days and for always pushing me to become a better person. I couldnt have done it without you. Te amo.

Table of Contents

List of Tables

List of Figures

Abstract

This thesis is focused on visual target tracking. Visual target tracking has been widely studied. The main idea is to be able to determine the target's location from a video sequence. Techniques such as the Kalman Filter and its variations have been proved to be the optimal solution when the system is linear or can be linearized, and Gaussianity can be assumed. But these conditions often do not hold in real world applications. Therefore, an alternative approach based on Sequential Monte-Carlo methods, also known as the Particle Filter, arose among others and has become a popular technique for target tracking recently. The particle filter is able to estimate the target state under nonlinear, non-Gaussian conditions. Different types of particle filters have been developed over the years, but one of the most popular is the sampling importance resampling (SIR) algorithm. However, in conditions of highly structured clutter and occlusion the filter's performance is decreased and the tracker can lock into the background and loose the target. Since motion information has been shown to be very important for the unmanned target tracking problem, in this thesis I introduce a new method to make the SIR filter more robust against these conditions by indirectly including velocity information in the likelihood function of the SIR filter. I propose augmenting the SIR filter state vector in order to use particle velocity information to prevent particles with poor motion estimates from obtaining large weights. The main original contributions of this thesis include the following:

- I developed the theoretical formulation for the State Vector Augmented SIR filter algorithm.

- I reformulated the normalized cross correlation used in the Likelihood function of the SIR filter to include the velocity information in it.
- I developed an algorithm to generate synthetic data sequences with targets that can change both in magnification and rotation for testing the efficacy of tracking algorithms in a controlled environment.
- I developed a simple template update strategy to deal with target appearance changes.
- I prove the effectiveness of the proposed algorithm with tracking results obtained from two longwave infrared sequences and two synthetic data sequences.

The results show that this new method can improve tracking performance for moving targets immersed in strong structured clutter.

Chapter 1

Introduction

Recently, particle filters have become popular for video object tracking [33, 58].Traditionally, the state dynamics are assumed to obey a first-order Markovian model. Different particle filtering approaches are reviewed by [2]. In particular, the sampling importance resampling(SIR) filter [2] is widely applied in the target tracking community.

One of the main issues with the SIR particle filter is that it can not avoid assigning large weights to particles that have a poor motion hypothesis. Thus, developing a way to utilize velocity information is important in order to deal with this problem. Different approaches have been developed in recent years. From high-order particle filters that use multiple previous states [41], to including velocity information into the state transition model [63] for dealing with the target appearance model variations. Nevertheless, these approaches fail to include any velocity information directly in the likelihood function of the filter.

Considering this problem, the focus of this thesis is to develop a method that incorporates velocity information in the likelihood function of the SIR filter for video tracking. The proposed method is based on a state vector augmentation technique that uses the current and past measurements to estimate the particle velocity. A similar approach is presented in [40] but they utilize a

block/patch matching. In the method proposed in this thesis, I use template matching because it often works better than other approaches for infrared video signals.

1.1 Organization

This thesis contains six chapters where I discuss different topics related to video target tracking, making strong focus on the theory of particle filters.

Chapter 2 covers basic concepts of visual target tracking. I explain the main components of a tracking algorithm and, additionally, I review recent popular techniques in the civilian visual target tracking community.

Chapter 3 offers an extensive discussion about particle filters, including the most widely implemented algorithms for target tracking. In addition I review recent techniques that include velocity information for target tracking using a particle filter framework.

In Chapter 4 I introduce the concepts and mathematical formulation of the state vector augmentation technique for the likelihood function of the SIR filter, which is the main contribution of this thesis. Additionally, I discuss a target template update strategy that was implemented in order to achieve better results in tracking for one of the studied sequences.

Chapter 5 is focused on the experimental work developed in order to demonstrate the effectiveness of the new method proposed in Chapter 4. I show the results obtained from evaluating the proposed method in four different sequences, two synthetic and two obtained using a longwave infrared sensor, and compare the method's performance with that of the standard SIR filter.

Finally, in Chapter 6 I provide a summary of the main contributions of this thesis, mentioning its advantages and limitations. In addition, I suggest directions for the future research related to the presented work.

Chapter 2

Background

Visual tracking is the process of inferring the motion of objects of interest, *e.g., targets*, over time given an image sequence [33].It is used to identify, locate and determine the position of the targets in each frame of a video [13].

Target tracking is often associated with military applications but it is also used for civilian ones [17]. In the military field, the objective is to be able to determine the locations of enemy objects and to be able to predict accurately what their motion will be. Typical military objects that are of interest for tracking include aircrafts, missiles and ground vehicles. In the case of civilian practices, target tracking has been widely used in the last few decades for weather radars, security surveillance, public transportation systems, rescue missions, among others [33, 60]. In these applications a common interest is to determine the movement of the targets, specifically, body and facial recognition and expression identification, human behavior, Doppler velocity, wind speed and direction of movement [33]. Another field where target tracking is widely used is in economics where it helps to monitor and predict the stock market behavior.

The work presented in this thesis is related to visual video target tracking for civilian applications. Therefore, in this chapter I will cover the different definitions and methods related to visual target tracking, providing a clearer

explanation of the process itself, as well as a simple overview of the different tracking methods used in the last decades. I will discuss the process in three main stages: detection, data association and tracking. These stages are implemented using different probabilistic and statistical approaches.

2.1 Detection

The first stage of a track processor is *Detection*. It consist in being able to detect the presence of the target and recognize it from the images or information provided by the sensors. In video target tracking, the data recorded corresponds to a discrete image of a specific scenario. The goal is to be able to extract the fundamental, or at least, important features of the target(s). The idea is to identify the target(s) properties in the image and reject all the information that does not belong to any target of interest. For example, if the targets are cyclist in a competition video, it is important to discard any pieces of the image belonging to the scenario itself like the sky, trees, the racetrack, etc. since they can become false alarms and cause failures in tracking. Such objects and image structure are generally referred to as "background" and/or "clutter". In visual target tracking, the four main approaches that have been used for background and clutter rejection are: spatial filtering, motion estimation, background subtraction and segmentation [60].

2.1.1 Spatial Filtering

Since the beginning of the 20th century spatial filters have been used for target tracking [7]. They can be classified in two groups: linear and nonlinear. In the linear approach, background and clutter are usually modeled as additive noise.

Thus, a signal $r(\mathbf{x})$ acquired from the sensor and consisting of a target $s(\mathbf{x})$ immersed in background and clutter is assumed to be of the form

$$r(\mathbf{x}) = s(\mathbf{x}) + n(\mathbf{x}). \tag{2.1}$$

where $\mathbf{x} \in \mathbb{R}^2$ for processing the video in 2D on a frame by frame (*i.e.*, image by image) basis and $\mathbf{x} \in \mathbb{R}^3$ for true 3D spatiotemporal processing.

The spatial filter $h(\mathbf{x})$ attempts to obtain $s(\mathbf{x})$ from the composite signal $r(\mathbf{x})$. Assuming a white and stationary noise signal, it has been shown that in order to maximize the signal to noise ratio, the cross correlation filter is the optimal choice [28]. This approach is very limited and tends to fail in a wide number of situations, like infrared target tracking where the nature of the noise makes it non-stationary [7]. On the other hand, many nonlinear filters can perform better under this condition. Some commonly used nonlinear filters are the median filter and morphological filters. The median filter performs efficiently to reject impulse noise. For targets that are impulse-like in appearance, a widely used approach has been to apply the median filter and then subtract the median filtered image from the original. The rationale for this is that, for such targets, the median filter provides an estimate of the background at each pixel. In view of (2.1), subtracting the median at each pixel will therefore provide an estimate of the target signature by rejecting the background and clutter [55]. Through an *abus de langage*, this operation has historically been called the "high-pass median filter."

In situations where the background presents highly structured objects, *i.e.*, clutter, morphological filters are used to reject them. Typical morphological operations, such as erosion and dilation can be combined in order to create

more complex ones, such as opening and closing. The main idea is to suppress the background clutter while enhancing target-like structure within an image.

Another nonlinear filter is the thresholding filter. This filter is very useful in video sequences where the target color, illumination, and intensity, among others, are very different from the background and clutter. Therefore by setting a threshold one can easily detect the occurrences of the target through the sequence. The big issue with this filter is determining the optimal threshold level. Furthermore, this filter could be combined with other spatial filters in the latest stage in order to eliminate results that could cause false alarms in the detection process [7].

2.1.2 Motion Estimation

A main feature of motion estimation is to find the movement of the target between image frames. The most popular and simple method was introduced by Lucas and Kanade in 1981 [35]. It assumes small changes in the target appearance and position model between contiguous frames to estimate motion vectors. The technique is based in least squares minimization and the error cost function is formulated by the sum of square differences (SSD), or l_2 norm squared [35]. The main issue of this algorithm lies in its linearity assumption. Hence, it fails to track for targets under nonlinear kinematics or when there is substantial apparent motion of the target.

Motion estimation can be implemented by block matching [20,23,36,62]. Each image or video frame acquired from the sensor is divided into blocks of a fixed size, 8×8 or 16×16 in the case of [20], and then each block is compared with its correspondent best matching block in the following frame to estimate

the movement of the object. Motion estimation has been widely implemented for video compression [36] and in the target tracking community is popular for traffic movement tracking [36]. The algorithm is simply a matched filter that minimizes the l_1 or l_2 norm between blocks from adjacent frames. The principal advantage of motion estimation is that it can effectively reject false alarms with similar appearance, structure, and/or texture to the targets but different motion. The biggest disadvantage is its computational complexity [20, 62].

2.1.3 Background Subtraction

For a moving target in a stationary scene, background subtraction has been widely used to perform detection by rejecting the background and clutter based solely on the target motion; this approach is effective even in the absence of any *a priori* information about the target appearance [42]. Most of the algorithms based on background subtraction consider a static background model which simplifies the problem since static pixels are associated with background and clutter. Real-time surveillance systems use background subtraction because it's simple and fast to implement; besides, the position of the camera is usually fixed, which validates the static assumption of the background. A big problem with this assumption is that it might generate false alarms in situations where either the sensor or background are not static. Different approaches to alleviate this issue have been proposed over the years. Some of the most popular techiniques include statistical background models as a mixture of Gaussian distributions [52], or *a priori* known target free background models fed to the tracking processor to train the model and create a set of possible background images [21]. Another popular technique consists in a low compu-

tational approach for online applications based on median filters of gray-scale image sub-blocks in a fixed number of frames with background statistics and model update [50]. Nevertheless, all of these have disadvantages either because of their computational complexity or false alarms causing drifting issues in the target motion model.

2.1.4 Segmentation

Segmentation is the process of dividing the image into perceptually similar, homogeneous regions or objects [60]. Segmentation can be performed in the pixel spatial domain, frequency domain, modulation domain or in feature space. The effectiveness of this method lies in the fact that as it separates an image into target or background regions, the tracker can focus on the likely target objects and discard the rest if the image. Segmentation is the basis of Discriminative Tracking methods where the image is broken into regions determined by a binary classifier that distinguishes between target and background. It is important to know that segmentation can be either used to initialize the tracking process or as a primary detection scheme for each frame in the sequence. Some of the most popular algorithms include Mean-Shift Clustering [11], k-means clustering and fuzzy clustering [14], among others. Segmentation has been used in infrared target tracking as a pre-processing step to extract regions of interest [4]. In visual target tracking it has also been implemented in the initial stages of algorithms such as STRUCK [19], MIL [3], and TLD [26], making them more robust. The main problem with implementing segmentation as part of the detection process is its high computational complexity, but it has proven that discriminative models perform better in many situations than generative

models [58].

2.2 Data Association

Video target tracking algorithms typically process the incoming video frames sequentially as they are acquired from the sensor. In the traditional approach, detection is implemented as an independent process that identifies potential target measurements in the current frame. Data association is the process of determining which of these measurements will be used to update each target instance that is currently being tracked. There exist two typical ways to approach the data association problem. The simplest one is the Nearest Neighbor (NN) Association in which the track for each target instance is updated using the exceedance or measurement that is closest to the predicted target position [6]. This method shows good results in single target applications and tends to fail in situations with multiple targets if they are close to each other. The other traditional method is choosing the maximum *a posteriori* probability(MAP) obtained from a cost function, *e.g.*, minimum distance to the target, minimum energy difference, maximum cross-correlation. Like the NN method, it tends to fail under multiple target tracking with high clutter conditions. Considering this, better solutions using statistical methods were developed. Bar-Shalom [5] suggested the use of Probabilistic Data Association filter, which weights the contributions of the different measurements by adding them statistically. This is performed under the assumption that there exist only one true measurement that is the result of the distribution of all the target's instances [5, 44]. Although this method outperforms the previous two in single target situations, it is not recommended for multiple target tracking because it would need to

be implemented for each of the targets in the scene and is very sensitive to false measurements due to background and/or clutter with properties that are similar to the target appearance in some respect [5]. A solution for multiple target tracking is the joint probabilistic data association filter (JPDA) [5] which prevents distinct target instances from being merged.

The following stage of the target tracking process uses the measurements obtained from the data association process to update estimates of the variables such as position, appearance, and state that are maintained for each target instance currently being tracked. In Sections 2.3 and 2.4 I will describe state space tracking methods and feature tracking methods and discuss some of their respective advantages and disadvantages.

2.3 State Space Tracking

In different types of systems the state space model is used to represent explicitly the relationship between the objects of interest and their measurements. In the real world, the measurements obtained from different sensors or devices are corrupted by some uncertainties which cause the obtained measurements to not be the *true* quantities of interest. The nature of these uncertainties can vary from inherent measurement device inaccuracies to environmental influences to interference of unwanted objects in the transmission paths, among others. In the state space methods, it is assumed that the target is governed by a state space system model with stochastic inputs. The target measurements acquired from the sensor are modeled as noisy observations of the true target. In the vast majority of practical applications, the measurement noise is assumed to be additive. After establishing the relationship between the available measure-

ments and true target state, the problem is reduced to one of optimization where the goal is to minimize the error between the true target state and the estimated target state in some respect(*e.g.*, average error, mean squared error, mean absolute error). The popular state space trackers are the Kalman filter, its extensions (EKF and UKF) and particle filters. In this section I will cover the basics of the state space model as well as a simple explanation for each of the aforementioned filters. An extended discussion of particle filters, which are the main subject of this thesis, appears in Chapter 3.

2.3.1 State Space Model

In the state space model the problem is addressed in two components: the observation model, correspondent to the measurements obtained by the sensing device, and the state model that is used to express the evolution of the target state. It is important to realize that the state vector usually includes both observable and unobservable states.

Let $\mathbf{x}_k \in \mathbb{R}^m$ be the true target state vector at discrete time instant $k \in \mathbb{N}$. Throughout this thesis, it is assumed that 0 is included in the natural numbers so that $\mathbb{N} = \{0, 1, 2, \ldots\}$. The target behavior is governed by the state update equation (or dynamic equation)

$$\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}). \tag{2.2}$$

where $\mathbf{u}_k$ is a zero-mean white stochastic process that is independent and identically distributed (i.i.d.). In general, the function $\mathbf{f}(\cdot)$ in (2.2) could be linear or nonlinear.

The measurements (or observations) $\mathbf{z}_k \in \mathbb{R}^n$ are given by

$$\mathbf{z}_k = \mathbf{h}(\mathbf{x}_k, \mathbf{v}_k) \tag{2.3}$$

where the measurement noise $\mathbf{v}_k$ is zero-mean, white, i.i.d., and mutually uncorrelated with $\mathbf{u}_k$. Additionally, the discrete state space model described by (2.2) and (2.3) usually admits the first order Markov process assumption. This assumption means that, probabilistically, the conditional probability of the current state given all previous states is equal to the conditional probability of the current state given only the previous state:

$$p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \ldots, \mathbf{x}_1, \mathbf{x}_0) = p(\mathbf{x}_k \mid \mathbf{x}_{k-1}). \tag{2.4}$$

Given the noisy measurements $\mathbf{z}_k$ in (2.3), the goal of target tracking is to estimate the true sequence of target states $\mathbf{x}_k$.

Under these conditions, target tracking can be interpreted as a problem of obtaining state vector estimates given their relationship with the measurements and model parameters, *e.g.*, the parameters of the system and the measurement noises $\mathbf{u}_k$ and $\mathbf{v}_k$ and the functions $\mathbf{f}(\cdot)$ and $\mathbf{h}(\cdot)$ in (2.2) and (2.3). This is equivalent to estimating the probability density function (pdf) $p(\mathbf{x}_k|\mathbf{z}_{0:k})$ which can be solved by sequential Bayesian filters [2]. That is, attempt to predict future states given the available measurements, and, update the prediction when new measurements are available. Prediction involves the construction of the pdf $p(\mathbf{x}_k|\mathbf{z}_{0:k-1})$ by using the Chapman-Kolmogorov equation:

$$\begin{aligned} p(\mathbf{x}_k|\mathbf{z}_{0:k-1}) &= \int_{-\infty}^{+\infty} p(\mathbf{x}_k, \mathbf{x}_{k-1}|\mathbf{z}_{0:k-1}) d\mathbf{x}_{k-1} \\ &= \int_{-\infty}^{+\infty} p(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{z}_{0:k-1}) p(\mathbf{x}_{k-1}|\mathbf{z}_{0:k-1}) d\mathbf{x}_{k-1} \\ &= \int_{-\infty}^{+\infty} p(\mathbf{x}_k|\mathbf{x}_{k-1}) p(\mathbf{x}_{k-1}|\mathbf{z}_{0:k-1}) d\mathbf{x}_{k-1}. \end{aligned} \tag{2.5}$$

The main problem with this approach is that it allows the increase in the error of the estimation by different types of uncertainties. One of the causes is that it uses the first order Markov approximation, which limits the use of past states to the very last one before the current to predict a future state. Another cause is the incorporation of the measurement and the system noises to the estimation as additive sources of signal distortion. The update step is performed when the observable quantity $\mathbf{z}_k$ arrives. The filtered state $\mathbf{x}_k$ may then be estimated by using Bayes theorem to update the posterior distribution $p(\mathbf{x}_k|\mathbf{z}_{0:k})$ according to [2]

$$p(\mathbf{x}_k|\mathbf{z}_{0:k}) = \frac{p(\mathbf{x}_k|\mathbf{z}_k p(\mathbf{x}_k|\mathbf{z}_{0:k-1})}{p(\mathbf{z}_k|\mathbf{z}_{0:k-1})}. \tag{2.6}$$

The State Space Model does not give any restriction on the types of function for $\mathbf{f}(\cdot)$ and $\mathbf{h}(\cdot)$ or for the measurement and the system noises $\mathbf{u}_k$ and $\mathbf{v}_k$ besides being i.i.d, zero mean, mutually uncorrelated random processes. The following subsections explain some situations where linearity and noise distribution definition serve to define the optimal filters for each one of the cases.

2.3.2 The Kalman Filter

This estimation filter was introduced in 1960 by Rudolph Kalman [27]. It is very popular in real world applications for being an optimal, easy-to-implement filter. It has been widely used in the computer vision community for visual tracking and facial recognition [11, 24]. It is also widely used for navigation systems, military target tracking and GPS.

The Kalman filter is based on the assumption that the functions $\mathbf{f}(\cdot)$

and $\mathbf{h}(\cdot)$, introduced in the Section 2.3.1, are linear, and the measurement and the system noises $\mathbf{u}_k$ and $\mathbf{v}_k$ have a Gaussian distribution. In this sense, if both assumptions hold, the optimal filter to estimate the system state vector that minimizes the Mean Square Error (MMSE) is the Kalman filter. This guarantees that the MMSE will be achieved when estimating any affine function of the state vector. Under these assumptions, the state update and measurement equations reduce to

$$\mathbf{x}_k = \mathbf{F}_k \mathbf{x}_{k-1} + \mathbf{u}_k \tag{2.7}$$

$$\mathbf{z}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k, \tag{2.8}$$

where $\mathbf{F}_k$ is the known state transition matrix of size $m \times m$, $\mathbf{H}_k$ is the known measurement matrix of size $n \times m$, and $\mathbf{u}_k$ and $\mathbf{v}_k$ are zero-mean white Gaussian noises with known covariance matrices $\mathbf{Q}$ and $\mathbf{R}$. Using this approach, the pdf construction described in the previous section becomes a simple problem to solve. It can be proved [27] that if a Gaussian distribution is at the input of a linear system, the output of the system will also have a Gaussian distribution but with different statistics, *i.e.*, different mean and variance. Hence, the probability density functions of the state vector $\mathbf{x}_k$ can be reconstructed recursively from the first and second order statistics of the initial state $\mathbf{x_0}$, which is assumed to have a known Gaussian distribution. The reason behind this is that any Gaussian distribution can be entirely described by its mean and variance.

Under these considerations, a single cycle of the Kalman filter is composed by a two-step computation. The *prediction* step derives the predicted mean vector $\mathbf{m}_{k|k-1}$, predicted error covariance matrix $\mathbf{P}_{k|k-1}$ and Kalman gain

K_k as follows

$$\mathbf{m}_{k|k-1} = \mathbf{F}_k \mathbf{m}_{k-1|k-1}, \tag{2.9}$$

$$\mathbf{P}_{k|k-1} = \mathbf{Q}_{k-1} + \mathbf{F}_k \mathbf{P}_{k-1|k-1} \mathbf{F}_k^T, \tag{2.10}$$

$$\mathbf{S}_k = \mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{R}_k, \tag{2.11}$$

$$\mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{H}_k^T \mathbf{S}_k^- 1. \tag{2.12}$$

And given these, the predicted state $\mathbf{x}_{k|k-1}$ is distributed as [6]

$$p(\mathbf{x}_k \mid \mathbf{z}_{0:k-1}) = \mathcal{N}(\mathbf{x}_k, \mathbf{m}_{k|k-1}, \mathbf{P}_{k|k-1}), \tag{2.13}$$

where $\mathcal{N}(\mathbf{x}_k, \mathbf{m}_{k|k-1}, \mathbf{P}_{k|k-1})$ represents a Gaussian distribution with mean $\mathbf{m}_{k|k-1}$ and covariance matrix $\mathbf{P}_{k|k-1}$. Then, the *update* step refreshes the pdf of the state, posterior mean vector $\mathbf{m}_{k|k}$ and posterior error covariance $\mathbf{P}_{k|k}$ when the measurement vector $\mathbf{z}_k$ becomes available, as shown by [6]:

$$\mathbf{m}_{k|k} = \mathbf{m}_{k|k-1} + \mathbf{K}_k(\mathbf{z}_k - \mathbf{H}_k \cdot \mathbf{m}_{k-1|k-1}), \tag{2.14}$$

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k \mathbf{H}_k \mathbf{P}_{k|k-1}, \tag{2.15}$$

$$p(\mathbf{x}_k \mid \mathbf{z}_{0:k}) = \mathcal{N}(\mathbf{x}_k, \mathbf{m}_{k|k}, \mathbf{P}_{k|k}). \tag{2.16}$$

After building the posterior pdf $p(\mathbf{x}_k \mid \mathbf{z}_{0:k})$, both steps are repeated for a new cycle. When the system is linear and the noise sources are Gaussian, the *most optimal* filter in terms of MMSE is the Kalman filter. Even for non-Gaussian noises, the Kalman filter remains as the best linear filter available [6, 25].

However, if the dynamics are nonlinear or if the noises are non-Gaussian, then the Kalman filter is not optimal. The following two subsections are devoted to filters that try to tackle nonlinear dynamics, non-Gaussian noises, or both.

2.3.3 The EKF and UKF

The Extended Kalman Filter (EKF) emerged as a solution to the non-linearity problem discussed at the end of Section 2.3.2. At each time step, the EKF linearizes the system dynamics along a tangent to the state trajectory in order to simplify the pdf propagation of the KF. The process consists in implementing a Taylor series expansion on the state transition function $\mathbf{f}_k(\cdot)$ around the mean value $\mathbf{m}_{k-1|k-1}$ and on the measurement function $\mathbf{h}_k(\cdot)$ around the predicted mean value $\mathbf{m}_{k|k-1}$. Typically, the higher order terms of the series are discarded [6]. The linearization allows to estimate the predicted mean vector $\mathbf{m}_{k|k-1}$ in the same way it was done for the Kalman filter prediction (2.13)

$$\mathbf{m}_{k|k-1} = \mathbf{f}_k(\mathbf{m}_{k-1|k-1}). \tag{2.17}$$

However, the error covariance matrix $\mathbf{P}_{k|k-1}$ has to be calculated differently from that in the Kalman Filter. Before computing the covariance (2.10), the state transition function $\mathbf{f}_k(\cdot)$ is linearized around $\mathbf{m}_{k|k-1}$, according to [6]

$$\mathbf{P}_{k|k-1} = \mathbf{Q}_{k-1} + \tilde{\mathbf{F}}_k \mathbf{P}_{k-1|k-1} \tilde{\mathbf{F}}_k^T, \tag{2.18}$$

where $\tilde{\mathbf{F}}_k = \frac{\partial f_k}{\partial \mathbf{x}_k} \mid_{\mathbf{x}=\mathbf{x}_{k-1|k-1}}$ is the Jacobian of $\mathbf{f}_k$ evaluated at $\mathbf{x}_{k-1|k-1}$. The Kalman gain $\mathbf{K}_k$ also requires a different calculation from that of the Kalman Filter. The function $\mathbf{h}_k(\cdot)$ is linearized around the predicted mean $\mathbf{m}_{k|k-1}$ before computing the Kalman gain to obtain

$$\mathbf{S}_k = \tilde{\mathbf{H}}_k \mathbf{P}_{k|k-1} \tilde{\mathbf{H}}_k^T + \mathbf{R}_k, \tag{2.19}$$

$$\mathbf{K}_k = \mathbf{P}_{k|k-1} \tilde{\mathbf{H}}_k^T \mathbf{S}_k^{-1}, \tag{2.20}$$

where $\tilde{\mathbf{H}}_k = \frac{\partial h_k}{\partial \mathbf{x}_k} \mid_{\mathbf{x}=\mathbf{x}_{k|k-1}}$ is the Jacobian $\mathbf{h}_k(\cdot)$ calculated at the predicted mean $\mathbf{m}_{k|k-1}$ [6]. The construction of the predicted distribution $p(\mathbf{x}_k \mid \mathbf{z}_{0:k-1})$

is defined as

$$p(\mathbf{x}_k \mid \mathbf{z}_{0:k-1}) = \mathcal{N}(\mathbf{x}_k, m_{k|k-1}, \mathbf{P}_{k|k-1}), \tag{2.21}$$

where $\mathcal{N}(\mathbf{x}_k, \mathbf{m}_{k|k-1}, \mathbf{P}_{k|k-1})$ represents a Gaussian distribution with mean $\mathbf{m}_{k|k-1}$ and covariance matrix $\mathbf{P}_{k|k-1}$.

The EKF *update* step refreshes the redefined mean vector $\mathbf{m}_{k|k}$, error covariance matrix $\mathbf{P}_{k|k}$, and pdf of the state $\mathbf{x}_{k|k}$ when the measurement vector $\mathbf{z}_k$ becomes available according to

$$\mathbf{m}_{k|k} = \mathbf{m}_{k|k-1} + \tilde{\mathbf{K}}_k \cdot (\mathbf{z}_k - \tilde{\mathbf{H}}_k \cdot \mathbf{m}_{k-1|k-1}), \tag{2.22}$$

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k \cdot \tilde{\mathbf{H}}_k \cdot \mathbf{P}_{k|k-1}, \tag{2.23}$$

$$p(\mathbf{x}_k \mid \mathbf{z}_{0:k}) = \mathcal{N}(\mathbf{x}_k, \mathbf{m}_{k|k}, \mathbf{P}_{k|k}). \tag{2.24}$$

Although the EKF has been used in a variety of Navigation applications, it is not easy to implements, difficult to tune and diverges in many know situations. The main reason for these issues is that the local linearity approximation of the EKF fails to be effective when the system dynamics exhibit any significant nonlinearities. In the beginning of this century, Julier and Uhlmann proposed the Unscented Kalman filter (UKF) which, based on the unscented transformation allows to efficiently propagate the pdf through the state update of a non-linear system. It outperforms both the KF and EKF because it is more accurate in predicting the first and second order statistics of a Gaussian distribution through non-linear filters and does not need to compute the Jacobian Matrix [25].

Despite the fact both EKF and UKF deal with the non-linearity issue of real systems and perform better than the Kalman filter for Gaussian noise,

they fail to work for systems with non-Gaussian noise. Considering this, the particle filter, which will be simply explained in the following subsection and extensively covered in Chapter 3 address the non-Gaussianity issue.

2.3.4 The Particle Filter

The particle filter is used to solve Hidden Markov Chain (HMM) and nonlinear filtering problems. It estimates the hidden states in the system when only partial observations are made and the noises are random signals with any kind of distribution. It computes the posterior distributions of the states by using the sequential Monte-Carlo filtering. Although its original implementation suffers from degeneracy problems and fails to track under many circumstances, one of its variations, the sampling importance resampling particle filter has become the *de facto* tracking base algorithm in many tracking systems, *e.g.*, visible and infrared video target tracking, and radar. This subject will be covered in Chapter 3 since it is of interest in this work to explain this method and its variations extensively.

2.4 Feature Tracking

At each time step, the state vector $\mathbf{x}_k$ and measurement equation (2.3) specify how the target signature will appear in the observation $\mathbf{z}_k$. For video tracking, in the simplest case (2.3) characterizes the target appearance in terms of image pixel intensity values. However, in some cases improved tracking performance can be obtained by instead characterizing the target appearance in terms of some alternative derived features [60]. For example, it may be advantageous to formulate (2.3) in terms of the histogram of pixel intensities that occur in

a bounding box or window about the predicted target centroid. This is called *feature tracking* and the derived observation space is referred to as the *feature space*.

Thanks to the high frame rate of current video recording devices target tracking can be simplified based on the spatiotemporal redundancy in the shapes, sizes, orientations and textures of moving objects. This allows to use feature similarity between frames to do the tracking. The following subsections will cover popular feature-based tracking methods: Point tracking, kernel tracking, contour tracking, multi-feature tracking and track by learning.

2.4.1 Point Tracking

In 2-D imaging, the images are comprised of pixels. Although the spatial relationships between pixels are fixed and allows to visually represent the appearance of the objects of interest, some pixels are more important than others. These pixels are considered *interest points*. In general, the pixels located on the edges of a shape carry more information about the shape of the target than the ones inside it [47] as they mark the boundary between the target and the rest of the image. After determining interest points that are spatially close to each other, they can be used as unique features of the object [51].

Given that interest points are used in groups their detection is made by looking at local structures in the neighborhoods of pixels. In [47] a window of a small number of pixels is used to compute the interest points by obtaining the eigenvalues in the vertical and horizontal direction of a correlation matrix around each pixel. If both eigenvalues were above a certain threshold, the central pixel of the window was selected as an interest point of the image.

One of the advantages of using point representations is that they are a robust strategy to deal with occlusion, because the target is represented by several points. In addition, as the interest points are selected based on the magnitudes of the eigenvalues of the correlation matrix around each pixel, tracking is not affected by rotations. Furthermore, it is an easy algorithm to implement.

This approach has been used in several tracking algorithm by incorporating it into the Lucas-Kanade registration algorithm [35], building it into the particle filter framework [49], or for compensating the camera-ego motions [59]. Nevertheless, the interest point representation has some important limitations. First of all, they do not have any spatial relationships between each other, which makes them sensitive to false points, because they are computed independently for each pixel [47]. They also can not provide any information on the target's contour or texture which makes them unsuitable for large-size objects [60].

2.4.2 Kernel Tracking

Comaniciu *et al.* in [12] used kernel tracking to track human faces and moving people in subway stations and football games. This method consists in computing local features of the target by convolving a small isotropic kernel about regions of interest. These kernels have a convex and monotonic profile that assigns smaller weights to the pixels farther from the kernel's center [12]. They represented the targets using color histograms, and for each time frame, potential targets were extracted using an ellipse-shaped kernel with variable size. They constructed a color histogram from the extracted pixels and matched them with the reference targets histograms by computing the Bhattacharyya

distance between both distributions. Then, the mean-shift algorithm was used to place the kernels to the location where the computed distance was minimum and the process was repeated until the distance fell below a predefined threshold. Because the mean-shift calculation is approximated by a first-order Taylor series, the targets must have small motion between frames in order for the algorithm to work [12, 61]. The kernel approach has also been used recently in algorithm such as [19] by combining it with a labeling strategy in a machine learning framework. The method consists in using a kernelized structured output support vector machine (SVM) learning machine to allow adaptive tracking. This way, the authors saved from computing the intermediate classification step of many discriminative methods because in this case the SVM itself performed the labeling online. The method has been proven to work better than many current state-of-the-art trackers [33]. In their paper [19] the authors compare the results with many boosting or random forest algorithms by establishing two scales on a 4×4 grid with six different types of Haar-like features for a total of 192 features, each one normalized to give a value between $[-1, 1]$. These features are then integrated into a feature vector $\mathbf{x}$ and, finally, they implemented a Gaussian kernel with fixed variance for all analyzed sequences. Finally, they computed the corresponding discriminant and set the highest score pixel as the new target location.

2.4.3 Template Tracking

The template tracking technique is based on representing the target with a window of the image with a specific size but arbitrary shape known as *template*. This method utilizes the pixel intensities of the template and assumes

they do not vary drastically during the course of tracking. In videos with color information, three channels (RGB), the color provides another dimension of target-background separability [60] since the color distribution of the background is going to be constant through the sequence if the background is static. It also allows to distinct different targets because their color absorption will differ. This is a main advantage to monochromatic, gray-scaled, video sequences. Nonetheless, this advantage also increases the computational cost. Templates can also be modeled by probabilistic distributions such as histograms [12]. For instance, templates can be constructed from modulation domain features (AM-FM) [54] and present a different solution to the segmentation problem.

The main advantage of templates is that they capture both the spatial and morphological information of objects in an easy-to-implement and effective way. However, they are sensitive to size, rotation and luminance changes of the target signatures, as well as partial occlusion. In addition, selecting the size and shape of the target becomes a problem itself since if it is too big it could potentially include unwanted background into the signature model or on the other hand omit parts of it, if it is too small. Therefore, the template tracking technique is most effective for tracking rigid targets.

2.4.4 Contour Tracking

Considering the changes in the target's appearance for non-rigid objects, contour representation becomes one of the solution to the problem. There are three popular techniques which will be described in this work: active contours, articulated models and silhouettes [60].

Active Contours

Active Contour models try to identify the boundary of objects by minimizing an energy function that includes internal forces, external constraints and image energy at edges [30]. As this method deals with deformable objects it is very popular in tracking systems of different nature. Besides, it is robust to partial occlusion [60]. The main disadvantages of this approach are that it does not consider any spatial or texture information of the target and in general requires the object motion and deformation rate to be small [30].

Articulated Model

This technique consist in representing the target as a set of multiple-connected parts that are modeled separately with templates of different shapes, *i.e.*, normally rectangular or elliptical [9]. It is very convenient for human gesture or movement tracking [60].

Silhouettes

This method is similar to the active contour representation in that it focuses on the shape of the object, but, in this case, instead of modeling the boundaries of the target, it focuses on the interior of the object's shape to perform the tracking [60]. In general, target silhouettes are computed after segmentation where small pieces of targets are combined using morphological filters and since they are usually binary, the technique is robust to changes in color and texture [21].

2.4.5 Multiple Features Tracking

Although, in many applications, a single feature works to describe the object of interest completely, a way of making a tracking algorithm robust to different situations is combining some of the different methods aforementioned. A common approach is to combine two or more appearance based tracking algorithms to describe the target appearance model and incorporate them into a particle filter framework [58].

2.4.6 Learning

In many tracking situations, prior knowledge of the object of interest is available. This allow us to construct a more accurate model for the target. This training step, performed before initializing the tracking, allows the tracker to discriminate between clutter and targets. This method is known as *learning*.

The process itself allows to build a library of target's signatures by using the prior information on its appearances and behaviors. Then, these signatures are matched with real data during the tracking process to find targets. The main issue with this approach is its high computational cost and storage requirements for large library sets. Different techniques have been used to address this problem. Principal Component Analysis has been used to reduce library size by preserving only high energy components [45, 57]. Another popular technique is based on the use of support vector machines (SVM) [3, 19, 26]. The main issue with many learning algorithm is that once the training process is done the optimality of the classifier can not be preserved because new data is being incorporated to the process and new targets that did not belong to the training library will not be tracked by this classifier. Hence, tracking by

learning is application specific.

Chapter 3

Particle Filters

As mentioned in Chapter 2 it is of interest in this thesis to explain extensively the subject of Particle Filters. Since their introduction in 1993 [18], these recursive, online estimation methods have become very popular for tracking and even classifying targets based a state space system model.

Considering the probabilistic nature of the state-space model and its need for updating information as new data is available, a solution could be a Bayesian approach that tries to obtain the posterior probability density function (pdf) of the state from all the available information. In this case, the recursive approach is desired; that is, allowing data to be processed sequentially as it is received without the need of storing or reprocessing any data when new measurements are available [2]. In view of this, the concepts described in Subsection 2.3.1 open the discussion for having an optimal solution that, without having any constraint of linearity or Gaussianity, can estimate the state. This solution is the particle filter [18].

This chapter is structured as follows. Firstly, I will cover the basics of the particle filter, its advantages and limitations. Then, I will cover some popular variations in the video target tracking community, *e.g.*, SIR ASIR, and RPF. Finally, I will cover some recently developed variations to end up summarizing the information and the reason for the use of the proposed method

of this thesis that will be explained in Chapter 4.

3.1 Sequential Importance Sampling Algorithm

The Sequential Importance Sampling Algorithm, also known as bootstrap filter, condensation algorithm or particle filter [2] is the base of all the Sequential Monte Carlo filters [2], and is based on the Sequential Monte Carlo Sampling Method [2]. Its objective is to estimate the state by representing the posterior density function using random samples with associated weights [2]. When the number of samples used for the estimation is very large, the particle filter solution tends to the optimal Bayesian estimate. I will broaden the state-space defined in Chapter 2 as follows. Let $\{\mathbf{x}^i_{0:k}, w^i_{0:k}\}^{N_s}_{i=1}$ be a *random measure*, where $\{\mathbf{x}^i_{0:k}, i = 1, ..N_s\}$ are a set of N_s particles and $\{w^i_{0:k}, i = 1, ..N_s\}$ their associated weights that are normalized so they sum to 1. These points support the description of the posterior pdf $p(\mathbf{x}_{0:k}|\mathbf{z}_{0:k})$ allowing it to be approximated as follows:

$$\hat{p}(\mathbf{x}_{0:k}|\mathbf{z}_{0:k-1}) \approx \sum_{i=1}^{N} \omega^i_k \delta(\mathbf{x}_k - \mathbf{x}^{(i)}_k). \tag{3.1}$$

This allows us to have a discrete approximation of the true pdf. It is important to note that the weights are chosen by using the *importance sampling* principle which consists in estimating the properties of a density $p(x)$, difficult to describe, by using a different density, the *importance desity* $q(\cdot)$, with similar properties that can be easily calculated. In this sense, the weighted approximation to the true posterior given by a particle $\mathbf{x}^i_k$ is described as follows:

$$p(\mathbf{x}_k) \approx \sum_{i=1}^{N_s} \omega^i \delta(\mathbf{x}_k - \mathbf{x}^i_k), \tag{3.2}$$

where ω^i are the normalized weights of the particles. This allows us to define ω_k^i in (3.1):

$$\omega_k^i \propto \frac{p(\mathbf{x}_{0:k}^i|\mathbf{z}_{0:k})}{q(\mathbf{x}_{0:k}^i|\mathbf{z}_{0:k})}. \tag{3.3}$$

If one wanted to approximate $p(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})$ from an already obtained approximation to $p(\mathbf{x}_{0:k-1}|\mathbf{z}_{0:k-1})$, an intelligent approach will be to choose the importance density such that at time k it admits as marginal distribution at $k-1$ the following factorization:

$$q(\mathbf{x}_{0:k}|\mathbf{z}_{0:k}) = q(\mathbf{x}_k|\mathbf{z}_{0:k})q(\mathbf{x}_{0:k-1}|\mathbf{z}_{0:k-1}). \tag{3.4}$$

This will allow to obtain the particles at time k by augmenting the already existent at time $k-1$. With properly mathematical handling, see [2], the weight update equation could be expressed as

$$\omega_k^i \propto \frac{p(\mathbf{z}_k|\mathbf{x}_k^i)p(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i)p(\mathbf{x}_{0:k-1}^i|\mathbf{z}_{0:k-1})}{q(\mathbf{x}_k^i|\mathbf{x}_{0:k-1}^i,\mathbf{z}_{0:k})q(\mathbf{x}_{0:k-1}^i|\mathbf{z}_{0:k-1})} = \omega_{k-1}^i \frac{p(\mathbf{z}_k|\mathbf{x}_k^i)p(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i)}{q(\mathbf{x}_k^i|\mathbf{x}_{0:k-1}^i,\mathbf{z}_{0:k})}. \tag{3.5}$$

If finally, one allows the importance density to be dependent only on the last state at time $k-1$ the weight can be then described as

$$\omega_k^i \propto \omega_{k-1}^i \frac{p(\mathbf{z}_k|\mathbf{x}_k^i)p(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i)}{q(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i,\mathbf{z}_k)}. \tag{3.6}$$

Thus, the posterior density $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ will be approximated as

$$\hat{p}(\mathbf{x}_k|\mathbf{z}_{1:k}) \approx \sum_{i=1}^{N} \omega_k^i \delta(\mathbf{x}_k - \mathbf{x}_k^{(i)}). \tag{3.7}$$

It can be proved that as the number or particles N_s tends to infinity the approximation approaches the true posterior density $p(\mathbf{x}_k|\mathbf{z}_{1:k})$.

This first approach to particle filtering suffers from a major drawback, *Degeneracy*. That is, that after some iterations, only one particle will have a

significant weight while the others will be negligible. This is caused by the fact that the variance of the weights can only increase with time. This implies that most of the particles contribution to the estimation is zero. There are different developed strategies to avoid this issue. Below, I will address the four main ones described in [2].

3.2 Generic Particle Filter

A way of measuring the degeneracy of the previous algorithm is by calculating the number of particles that are effectively contributing to the estimation. Although this number can not be evaluated exactly, one could use the following estimate of it to obtain an approximated value:

$$\hat{N}_{\text{eff}} = \frac{1}{\sum_{i=1}^{N_s} (\omega_k^i)^2}. \tag{3.8}$$

With this approximation, one could avoid degeneracy by applying a resampling to the particles every time the number of effective particles $\hat{N}_{\text{eff}}$ falls below certain threshold. Let be the set of particles before the resampling step.The process of resampling consists in generating a new set of particles $\{\mathbf{x}_k^{i^*}\}_{i^*=1}^{N_s}$, where i^* is the new particles indexes, from the original set $\{\mathbf{x}_k^j\}_{j=1}^{N_s}$ by resampling N_s times from (3.7) so the probability function of the resampled set is equivalent to the sum of their normalized weight; that is

$$\mathbf{Pr}(\mathbf{x}_k^{i^*} = \mathbf{x}_k^j) = \omega_k^j. \tag{3.9}$$

Then, the new weights of the particles are reset to be

$$\omega_k^{i^*} = \frac{1}{N_s}. \tag{3.10}$$

Finally, let $i^* \equiv i$, then, $\omega_k^{i^*} = \omega_k^i$.

There are several forms to perform the resampling step, this work will use the one applied in [2] which is the *Systematic Resampling*. The *Generic Particle Filter* is an algorithm that computes the estimate of $\hat{N}_{\text{eff}}$ and after it falls below the threshold applies resampling. Although it is a better approach than the SIS filter, it still suffers from major limitations. The first one is that it makes it difficult to parallelize the particles estimation process because all the particles are combined in the resampling step. The second one is that as the particles with high weights are statistically selected repeatedly, the estimated distribution will contain many copies of the same points, causing a loss in diversity, also known as *sample impoverishment*. Particularly, this problem in small noise systems leads to *degeneracy* in very few iterations [2, 10]. A few alternatives to the SIS filter will be explained below.

3.3 Sampling Importance Resampling Filter

This method, proposed in 1993 by Gordon *et al.* is a Monte Carlo method applied to sequential Bayesian problems. It requires to have a previously known state evolution and measurement functions, $\mathbf{f}_k(\cdot,\cdot)$ and $\mathbf{h}_k(\cdot,\cdot)$. It also needs to obtain samples from the noise process $\mathbf{v}_{k-1}$ and the prior, as well as to evaluate the likelihood function $p(\mathbf{z}_k|\mathbf{x}_k)$ point-wise. It can be derived from the SIS filter formulation by assigning the importance density $q(\mathbf{x}_k|\mathbf{x}_{k-1}^i, \mathbf{z}_k)$ to be the prior density $p(\mathbf{x}_k|\mathbf{x}_{k-1}^i)$ and applying resampling in every time step. Considering this, one can formulate the update equation for the state and the weights as follows:

$$\mathbf{x}_k^i \sim p(\mathbf{x}_k|\mathbf{x}_{k-1}^i), \tag{3.11}$$

$$\omega_k^i \propto p(\mathbf{z}_k|\mathbf{x}_k^i). \tag{3.12}$$

This means, that the weights of the particles can be drawn proportionally from the likelihood function which constitutes an advantage of using this algorithm because the importance weights can be easily evaluated [2]. This makes the SIR filter a desirable alternative to the SIS filter. However, it still suffers from the same impoverishment problem of the Generic Particle Filter. Besides, as the importance sampling density is independent of the measurements $\mathbf{z}_k$, there is no a priori knowledge of the observations which can be very inefficient and makes the SIR filter susceptible to outliers. A different approach to resampling is introduced in the Auxiliary SIR filter.

3.4 Auxiliary Sampling Importance Resampling Filter

This particle filter method was introduced by Pitt and Shephard in 1999 in [38] as an alternative to the SIR filter which is based in altering the order in the sampling and resampling steps in order to minimize the loss of information based on the fact that it allows greater distinction between particles which leads to a better approximation of the target. In a way, the change of order creates a method that utilizes the future observation information to determine the actual contribution of each particle in the current observation. It allows to determine which particles should survive resampling. There are several modifications to the original implementation. In this work, I will use the same defined by [2].

The algorithm introduces a different importance density $q(\mathbf{x}_{k+1}, i|\mathbf{z}_{0:k+1})$ based on the future sample pair formed, predicted by the particle $\mathbf{x}_{k+1}^j$ and its index i^j, at the current time step k : $\{\mathbf{x}_{k+1}^j, i^j\}_{j=1}^{M_s}$. Then, using Baye's rule,

the proportionality for the joint density $p(\mathbf{x}_{k+1}, i|\mathbf{z}_{0:k+1})$ can be defined as

$$p(\mathbf{x}_{k+1}, i|\mathbf{z}_{0:k+1}) \propto p(\mathbf{z}_{k+1}|\mathbf{x}_{k+1})p(\mathbf{x}_{k+1}|\mathbf{x}_k^i)\omega_k^i. \tag{3.13}$$

Then, one can draw a sample from this joint density, omit the indexes and generate a new sample $\{\mathbf{x}_{k+1}^i\}_{j=1}^i$ approximating $q(\mathbf{x}_{k+1}, i|\mathbf{z}_{0:k+1})$ by (3.13)

$$q(\mathbf{x}_{k+1}, i|\mathbf{z}_{0:k+1}) \propto p(\mathbf{z}_{k+1}|\mu_{k+1})p(\mathbf{x}_{k+1}|\mathbf{x}_k^i)\omega_k^i, \tag{3.14}$$

where μ_{k+1}^i is some characterization associated with the conditional density $\mathbf{x}_{k+1}|\mathbf{x}_k^i$. Defining

$$q(\mathbf{x}_{k+1}|i, \mathbf{z}_{0:k+1}) \triangleq p(\mathbf{x}_{k+1}|\mathbf{x}_k^i) \tag{3.15}$$

and writing the joint density as

$$q(\mathbf{x}_{k+1}, i, \mathbf{z}_{0:k+1}) = q(i|\mathbf{z}_{0:k+1})q(\mathbf{x}_{k+1}|i, \mathbf{z}_{0:k+1}), \tag{3.16}$$

the first or partial weights can be approximated by the relation

$$q(i|\mathbf{z}_{0:k+1}) \propto p(\mathbf{z}_{k+1}|\mu_{k+1}^i)\omega_k^i. \tag{3.17}$$

Finally, the sample pair $\{\mathbf{x}_{k+1}^j, i^j\}_{j=1}^{M_s}$ is assigned with a new weight proportional to the ratio between the joint density and the importance density:

$$\omega^j = \frac{p(\mathbf{z}_{k+1}|\mathbf{x}_{k+1}^j)}{p(\mathbf{z}_{k+1}|\mu_{k+1}^{i^j})}. \tag{3.18}$$

Although, in systems of small process noise, the ASIR performs better than the SIS, the SIR filter outperforms it in systems with large process noise. The reason is that in small noise μ_{k+1}^i represents a well characterization of $p(\mathbf{x}_{k+1}|\mathbf{x}_k^i)$. Whereas, in large noise, the same single sample characterization is very poor and the filter's performance is considerable degraded.

3.5 Regularized Particle Filter

The main problem with the aforementioned filters is that the resampling step involves samples drawn from discrete distributions and this might lead to *particle collapse* which is basically the case of particle impoverishment where all the particles occupy the exact same point degrading the performance of the filters because of the poor posterior density representation. Considering this, there emerges the *Regularized Particle Filter* (RPF) which has the same formulation as the SIR filter except for the resampling step, that in the RPF, involves drawing samples from a continuous representation of the posterior density given by the approximation

$$p(\mathbf{x}_k|\mathbf{z}_{0:k}) \approx \sum_{i=1}^{N_s} \omega_k^i K_h(\mathbf{x}_k - \mathbf{x}_k^i), \tag{3.19}$$

where

$$K_h(\mathbf{x}) = \frac{1}{h^{n_x}} K(\frac{\mathbf{x}}{h}) \tag{3.20}$$

is the rescaled *Kernel density* $K(\cdot)$, $h > 0$ is a scalar parameter which defines the Kernel bandwidth, n_x is the dimension of the state vector $\mathbf{x}$ and $\omega_k^i, i = 1, ..N_s$ are the normalized weights. The Kernel density function is a symmetric pdf that satisfies:

$$\int \mathbf{x} K(\mathbf{x}) d\mathbf{x} = 0, \qquad \int \|\mathbf{x}\|^2 K(\mathbf{x}) d\mathbf{x} < \infty.$$

The intention is choosing the Kernel and the bandwidth that minimize the mean integrated squared error (MISE) between the estimation and the true posterior. When all the samples have the same weight and the density is Gaussian with unit covariance matrix, the optimal kernel to choose is the Epanechnikov and the optimal bandwidth can be defined as in [2]. The main

issue with this algorithm is that the Kernel approximation degrades its wellness as the states dimensionality increases. Furthermore, the samples are not guaranteed to asymptotically approximate the posterior.

In general, the SIR filter is still the preferred algorithm within the target tracking community because of its versatility and straightforward implementation. In addition, it can be easily mixed with different feature-based tracking algorithms because of its likelihood function definition.

3.6 Motion inclusion in the Particle Filter framework

The basic formulation of the particle filter avoids to include any motion information to perform either the prediction, update or both stages of filtering. Recently, researches have been working extensively to improve the particle filter implementation to make it robust to different situations such as, appearance model changes, particle impoverishment, interframe motion among others. Including motion information then, becomes a very useful tool to be able to deal with the aforementioned issues. In this section we will briefly describe three different approaches that include motion information into the particle filter implementation that have been developed in the last 20 years.

3.6.1 Including Velocity in the State Transition Model in a Particle Filter

In [63], Zhou *et al.* introduce a tracking algorithm that, using the particle filtering framework, is able to perform recognition and tracking simultaneously by incorporating motion information in the state transition model. The method stabilizes the tracker by incorporating three modifications to conventional tracking

algorithms. These modifications are: an adaptive appearance-based model for efficiently dealing with appearance changes; an adaptive velocity motion model with adaptive noise variance that derives velocity information from the previous state sample set using a first order linear approximation which allows them to define an adaptive state transition model; and, finally, an adaptive noise variance which helps to set an adaptive number of particles.

The adaptive appearance model is time-varying and models all the appearances up to the previous time instant $k-1$. It utilizes a mixture of Gaussians that helps define the likelihood function and is the base for the model update stage. The likelihood function uses the mixture centers, mixing probabilities and correspondent variances assuming that the distributions are independent of each other. For the model update stage, the authors assume the contributions of the past observations decrease exponentially with time for the current frame and then, they invoke the expectation-maximization (EM) algorithm described in [1]. It is beyond the scope of this thesis to include the mathematical formulation for this adaptive appearance model.

For the adaptive state transition model, the authors incorporate the previous particle configuration in the prediction scheme as follows. They take the complete sample set $\{\mathbf{x}_{k-1}^i\}_{i=1}^{N_s}$ and the appearance model of the previous frame to predict the shift in motion for the current frame by using a first-order linear approximation based on the constant brightness constraint. This can be interpreted as follows. Let $\mathbf{x}_k$ be a sample that satisfies $\tau_k(\mathbf{z}_k;\mathbf{x}_k) \approx \hat{\tau}_{k-1}$, where τ_k is the image patch of interest in the current frame and $\hat{\tau}_{k-1} \equiv \tau_{k-1}(\mathbf{z}_{k-1};\hat{\mathbf{x}}_{k-1})$ is the previous frame estimation. This can be defined as the current model by using a first-order Taylor series expansion around the sample $\tilde{\mathbf{x}}_k$ that is set to

be equal to the state estimated in the previous frame $\hat{\mathbf{x}}_{k-1}$:

$$\tilde{\tau}_k(\mathbf{z}_k, \mathbf{x}_k^{i'}) \approx \tau_k(\mathbf{z}_k, \tilde{\mathbf{x}}_k) + J_k(\mathbf{x}_k - \tilde{\mathbf{x}}_k) = \tau_k(\mathbf{z}_k, \tilde{\mathbf{x}}_k) + J_k v_k, \tag{3.21}$$

where J_k is the Jacobian matrix of the state and $v_k = \mathbf{x}_k - \tilde{\mathbf{x}}_k$ the adaptive velocity. Clearing v_k from (3.21)

$$v_k \approx -B_k(\tau_k(\mathbf{z}_k, \tilde{\mathbf{x}}_k) - \widetilde{\tau}_k(\mathbf{z}_k, \mathbf{x}_k^{i'})), \tag{3.22}$$

where B_k is a pseudo-inverse of the Jacobian matrix that can be estimated from the available $\{\mathbf{x}_{k-1}^i\}_{i=1}^{N_s}$ and their correspondent image patches. The following step consists in composing matrices based on the differences in motion vectors $\mathbf{x}_{k-1}^i$ and τ_{k-1}^i for each particle, using $\tau_k(\mathbf{z}_k, \tilde{\mathbf{x}}_k)$ and $\hat{\mathbf{x}}_{k-1}$ as references. Then, B_k is calculated by using Singular Value Decomposition (SVD) and taking only the first q values [63]. Note that as the appearance model is adaptive, B_k needs to be calculated in each timestep.

In order to obtain a good estimation of v_k one must run several iterations until the error between the predicted appearance and the update appearance model ϵ_k is minimized to some value.

Finally, the state transition model can be defined as:

$$\mathbf{x}_k = \hat{\mathbf{x}}_{k-1} + v_k + U_k, \tag{3.23}$$

where U_k is an adaptive noise defined as

$$U_k = R_k U_0, \tag{3.24}$$

where R_k is a scale factor that depends on the error ϵ_k and helps adjust the number of particles and U_0 is a standardized random vector [63].

3.6.2 Including Velocity Information in the Likelihood Function of SIR filters based on Block/Patch Matching

In [40] the authors propose a new method for tracking that involves a new graphical model that modifies the measurement process to include motion information into the likelihood function. This means that the current observation is assumed to be dependent on both the current and previous object configuration and the past observations. It also includes explicit motion information in the proposal density to deal with two main issues. In general, the proposal density is very hard to define and a common approach is to use the system dynamics as the proposal. However, this assumption implies that the model needs to be tight enough so it does not get degraded by outliers, but loose enough so it can cope with abrupt motion changes. Therefore, the method proposed by Odobez *et al.* introduces a new definition of the proposal distribution to deal with the issues aforementioned. Instead of using a template based appearance model, the authors measure the similarity between visual motion estimated from low-level information and the motion field induced by the state change.

The object is represented by a region R centered at the origin and that allows geometric transformations given a shape or a color distribution. These transformations are based on an affine transformation that includes translation $\mathbf{T}$, scaling s and aspect ratio r. The affine transformation $\alpha(\mathbf{T}, s, r)$ of the current and previous frame define the new state $\mathbf{x}_k = (\alpha_k, \alpha_{k-1})$.

The motion estimation is used for observations and for new state values. They utilize an affine displacement model $\vec{d}_\Theta$ parametrized by $\Theta = (a_i), i = 1, ..., 6$ and defined as

$$\vec{d}_\Theta(\mathbf{x}) = \begin{pmatrix} a_1 + a_2 x_1 + a_3 x_2 \\ a_4 + a_2 x_5 + a_6 x_2 \end{pmatrix}. \tag{3.25}$$

The estimation of Θ, $\hat{\Theta}$ is made by using a gradient-based multiresolution robust estimation method (See [40] for details). This allows to obtain a robust and accurate estimation of the motion model. Given these estimates and assuming that the targets coordinates are given in the objects center, one can measure the variation of the affine transformation α_{k-1}^m and use it to define the predicted transformation α^p:

$$\alpha_k^p = \alpha_{k-1} + \alpha_{k-1}^m, \tag{3.26}$$

where the values in α_{k-1}^m can be derived as some derivative estimates of $\mathbf{T}$, s, r.

The Data Likelihood model assumes two types of measurements, object $\mathbf{z}_k^o$ and patch gray level $\mathbf{z}_k^g$ measurements that are independent to each other given the measurements. In addition, the object measurements are assumed to be uncorrelated. Hence, the likelihood can be defined as:

$$p(\mathbf{z}_k|\mathbf{z}_{k-1}, \mathbf{x}_k, \mathbf{x}_{k-1}) = p(\mathbf{z}_k^o|\mathbf{x}_k)p(\mathbf{z}_k^g|\mathbf{z}_{k-1}^g, \mathbf{x}_k, \mathbf{x}_{k-1}). \tag{3.27}$$

Modeling the observations is done by using two instances: a visual object measurement approach based in both shape and color models and an image correlation measurement that consists in extracting measures from the parameter space and implementing warping on gray-level local patches $\tilde{\mathbf{z}}_{\mathbf{x}_k}^g$ according to the state values. Then, the likelihood function is said to have a proportionality as follows:

$$p(\mathbf{z}_k^o|\mathbf{x}_k)p(\mathbf{z}_k^g|\mathbf{z}_{k-1}^g, \mathbf{x}_k, \mathbf{x}_{k-1}) \propto p_{\mathbf{x}_a}(\alpha_{k-1}^m, \alpha_k, \alpha k - 1)p_{\mathbf{x}_b}(\tilde{\mathbf{z}}_{\mathbf{x}_k}^g, \tilde{\mathbf{z}}_{\mathbf{x}_{k-1}}^g), \tag{3.28}$$

where the right-hand side terms are defined as

$$p_{\mathbf{x}_a}(\alpha_{k-1}^m, \alpha_k, \alpha k - 1) = \mathcal{N}(\alpha_k^p, \alpha_k, \Lambda_{\xi_p}) \tag{3.29}$$

and

$$p_{\mathbf{x}_b}(\tilde{\mathbf{z}}^g_{\mathbf{x}_k}, \tilde{\mathbf{z}}^g_{\mathbf{x}_{k-1}}) = K^{-1} \exp\left[-\lambda_{cor} \mathbf{D}^2_{\mathbf{x}}(\tilde{\mathbf{z}}^g_{\mathbf{x}_k}, \tilde{\mathbf{z}}^g_{\mathbf{x}_{k-1}})\right], \tag{3.30}$$

where Λ_{ξ_p} is the Covariance of the measurements, K is a normalization constant that is computed between two consecutive patches, $\mathbf{D}_{\mathbf{x}}$ is the distance between two image patches and λ_{cor} is a correlation parameter set to 20 in [40]. Note that the distance $\mathbf{D}_{\mathbf{x}}$ is defined as the inverse of the normalized cross-correlation between two image patches:

$$\mathbf{D}_{\mathbf{x}}(\tilde{\mathbf{z}}_1, \tilde{\mathbf{z}}_2) = 1 - NCC(\tilde{\mathbf{z}}_1, \tilde{\mathbf{z}}_2). \tag{3.31}$$

The method uses a Cauchy distribution to model the process noise of the prior which is described by a second order AR model for each component of α. Finally, the proposal density $q(\mathbf{x}_k|\mathbf{x}^i_{0:k-1}, \mathbf{z}_{1:k})$ is defined using the fact that $\mathbf{x}_k = (\alpha_k, \alpha_{k-1})$ and letting $\alpha_{k-1} = \alpha^i_{k-1}$ which allows them to draw α_k from $q(\alpha_k|\alpha^i_{k-1}, \mathbf{z}_k, \mathbf{z}_{k-1})$ using the following:

$$q(\alpha_k|\alpha^i_{k-1}, \mathbf{z}_k, \mathbf{z}_{k-1}) = \mathcal{N}(\alpha_k; \alpha^p_k(\alpha^i_{k-1}), \Lambda_{\xi_p}). \tag{3.32}$$

It is important to note that this algorithm was developed based on face tracking and uses elliptical shapes as geometric regions which allows the affine transformation not to depend on rotation parameters. In more complicated situations and shapes, in order to have an efficient and accurate affine transformation model, it will be necessary to include the rotation information. It is also important to note that although this method might seem very useful for tracking, it requires high computational cost because it combines shape and color models as well as warping in order to define the likelihood function and process two consecutive frames to define them.

3.6.3 High Order Particle Filters

In 2011, Pan and Schonfeld [41] proposed a method that extends the particle filter formulation from the first order hidden Markov chain model to a higher order m, which proves to improve the accuracy and robustness of particle filters in visual tracking.

The new Markov Chain Model for the state-space model is defined so that, when the order of the chain is $m \geqq 1$ the current state $\mathbf{x}_k$ depends only on the past m states. That can be understood as:

$$p(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{x}_{k-2}, ...\mathbf{x}_0) = p(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{x}_{k-2}, ...\mathbf{x}_{k-m}). \tag{3.33}$$

In order to define the relationship between the states and measurements, the authors assume a graphical model. First, they assume that the high-order Markov model is described by an acyclic directed graph. This type of graph has an associated *Moral graph*, which is an undirected graph that takes a node of the directed acyclic graph and adds a new connection with another node with whom it has a common connection. The *Moral graph* helps to establish the independence and conditionality of the states and measurements of the mth-order Markov chain, which allows to derive the posterior densities given the following relations:

$$p(\mathbf{z}_k|\mathbf{x}_{0:k}, \mathbf{z}_{0:k-1}) = p(\mathbf{z}_k|\mathbf{x}_k) \tag{3.34}$$

$$p(\mathbf{x}_k|\mathbf{x}_{0:k}, \mathbf{z}_{0:k-1}) = p(\mathbf{x}_k|\mathbf{x}_{k-m:k-1}). \tag{3.35}$$

Considering the conditional independence obtained from the graphical

model, the posterior density can be expressed as follows:

$$\begin{aligned} p(\mathbf{x}_{0:k}|\mathbf{z}_{0:k}) &\propto p(\mathbf{z}_k|\mathbf{x}_{0:k}, \mathbf{z}_{0:k-1})p(\mathbf{x}_k|\mathbf{x}_{0:k}, \mathbf{z}_{0:k-1})p(\mathbf{x}_{0:k-1}|\mathbf{z}_{0:k-1}) \\ &= p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{x}_{k-m:k-1})p(\mathbf{x}_{0:k-1}|\mathbf{z}_{0:k-1}). \end{aligned} \tag{3.36}$$

The filtering stage of the algorithm is performed using a SIS framework. The new defined importance density is:

$$q(\mathbf{x}_{0:k}|\mathbf{z}_{0:k}) = q(\mathbf{x}_k|\mathbf{x}_{k-m:k-1}, \mathbf{z}_k)q(\mathbf{x}_{0:k-1}|\mathbf{z}_{0:k-1}). \tag{3.37}$$

This is obtained by augmenting the existing samples $\mathbf{x}^i_{0:k-1} \sim q(\mathbf{x}_{0:k-1}|\mathbf{z}_{0:k-1})$ with the new state $\mathbf{x}^i_k \sim q(\mathbf{x}_k|\mathbf{x}_{k-m:k-1}, \mathbf{z}_k)$ and applying the conditional independence given by the graphical model.

Thus, the weight update (3.3) is redefined as:

$$\begin{aligned} \omega^i_k &\propto \frac{p(\mathbf{z}_k|\mathbf{x}^i_k)p(\mathbf{x}^i_k|\mathbf{x}^i_{k-m:k-1})p(\mathbf{x}^i_{0:k-1}|\mathbf{z}_{0:k-1})}{q(\mathbf{x}^i_k|\mathbf{x}^i_{k-m:k-1}, \mathbf{z}_k)q(\mathbf{x}^i_{0:k-1}|\mathbf{z}_{0:k-1})} \\ &= \omega^i_{k-1}\frac{p(\mathbf{z}_k|\mathbf{x}^i_k)p(\mathbf{x}^i_k|\mathbf{x}^i_{k-m:k-1})}{q(\mathbf{x}^i_k|\mathbf{x}^i_{k-m:k-1}, \mathbf{z}_k)}, \end{aligned} \tag{3.38}$$

where $p(\mathbf{z}_k|\mathbf{x}_k)$ is the likelihood function, the transition probability is given by $p(\mathbf{x}^i_k|\mathbf{x}^i_{k-m:k-1})$ and the importance density is $q(\mathbf{x}^i_k|\mathbf{x}^i_{k-m:k-1}, \mathbf{z}_k)$. Hence, the filtered posterior density can be approximated by

$$p(\mathbf{x}_{k-m+1:k}|\mathbf{z}_{0:k}) \approx \sum_{i=1}^{N_s} \omega^i_k \delta(\mathbf{x}_{k-m+1:k} - \mathbf{x}^i_{k-m+1:k}) \tag{3.39}$$

Other approaches like the one in [22], try to include velocity information in the state update model using a particle filter framework. However, their formulation will not be included in this work.

The existence of these many previous algorithms which have attempted in one way or another to incorporate motion information into the particle filtering framework strongly motivate the main objective of this thesis, which is to

obtain a new robust and theoretically rigorous means to consider velocity information directly in the SIR filter likelihood function. In the following chapter, I will explain the methodology used to accomplish our main objective.

Chapter 4

Adding Velocity Information to the Particle Filter by State Vector Augmentation

In Chapter 3, I discussed extensively the subject of particle filtering, describing the main algorithms used frequently in the visual object tracking community as well as some recently developed variations that are of interest in this thesis. As I mentioned before, particle filters have become the *de facto* framework in target tracking because they are able to deal with non-Gaussian noise and non-linear systems. However, the technique has several disadvantages and it is often combined with other feature-based trackers in order to improve the performance of the tracker [43].

A major problem with particle filters in visual object tracking occurs in scenarios of structured clutter and a target which exhibits significant appearance changes through the sequence. Under these conditions it is desired to have an algorithm that is able to dynamically adapt and update the appearance model of the target when it is necessary. The question of how and when to perform appearance model updates has been widely addressed in the last decades [46, 56, 63].

Under the aforementioned conditions, a very common issue is that some particles can get large weights even when they have a poor state hypothesis because they partially match the clutter. As the number of particles is fixed,

the effect of particles with good state hypothesis that partially match the target is decreased and what is more, the *bad* particles are converted into many more after the resampling step. This has a prejudicial effect on the tracking performance.

Different strategies including marginalized particle filters and box particle filters have been implemented in order to improve the sampling efficiency [16,32,48] and efficiently deal with clutter [8,19,32]. Popular techniques including PCA decomposition, block/patch matching, machine learning, SVM, model update, fusion of trackers embedded in a particle filter framework for visual object tracking are reviewed in [58]. Although many of them deal with different challenging situations, *i.e.*, structured background clutter, occlusion, rapidly changing target appearance models, many of them do not include any motion information.

Motion information has been recognized as an important factor to include when doing video tracking [34, 41, 53, 63]. However, as in practical video tracking application the camera only produces one frame at each time step, it is not possible to obtain velocity measurements directly from it without incorporating other sensors. This means that there is not an explicit way to obtain velocity information from the measurements. In addition, equation (3.12) shows that particles with bad velocity hypotheses cannot be penalized using the weight calculation. In fact, velocity information is often completely omitted from the likelihood function calculation which is then based only on appearance variables.

Considering this, it can and does occur that a particle with a good appearance hypothesis and a good velocity hypothesis obtains the same weight

as a particle with same appearance hypothesis but a poor velocity hypothesis. Furthermore, a particle that matches the clutter might get a large weight of same or larger magnitude than a particle that matches the target, even when the velocity and appearance hypotheses of the first one are actually poor compared to those of the second one. This implies that it is necessary to have a way of preventing the *bad* particles from being propagated and multiplied in future time steps and enhancing the effect of the *good* particles.

All the situations mentioned before motivated the main contribution of this thesis which is based in including velocity information in the likelihood function of the SIR filter so the effect of particles with poor velocity hypothesis can be diminished. In this chapter I will cover a new formulation for the likelihood function of the SIR filter that will include indirect velocity information by introducing a state vector augmentation technique.

The outline of the chapter is as follows: first, I will cover the regular particle filter formulation. Then, I will define a commonly used likelihood function for the SIR filter in the context of template tracking. Finally, I will introduce the new state vector augmentation and define the new formulation of the likelihood function for incomporating the velocity information into it.

4.1 Standard SIR Filter Formulation

As I discussed in Chapters 2 and 3, the SIR filter is based on the Space State Model. The SIR filter formulation is based on First-order Markovian state dynamics [2]. In addition, it is assumed that the target kinematics obey a constant velocity model [31, 63] typically described by a state vector of the

generic form

$$\mathbf{x}_k = \begin{bmatrix} x_k & \dot{x}_k & y_k & \dot{y}_k & \gamma_k & \dot{\gamma}_k & \theta_k & \dot{\theta}_k \end{bmatrix}, \tag{4.1}$$

where $[x_k\ y_k]^T$ is the target centroid, $[\dot{x}_k\ \dot{y}_k]^T$ are the correspondent velocity in each direction, γ_k and θ_k are the magnification and rotation relative to an initial reference model and $\dot{\gamma}_k$ and $\dot{\theta}_k$ their correspondent time derivatives. Having the state update model defined in (2.2) and measurement model defined in (2.3) and the same formulation for the SIR filter described in Section 3.3 of Chapter 3 in equations (3.11) and (3.12). The posterior density can be approximated the same way as it was described in equation (3.7).

Let $\mathbf{z}_k$ be a sequence of images from a video converted to grayscale and let $\mathbf{T}$ be a template for the target, also converted to grayscale, either manually designated in the first image of the sequence or obtained by a segmentation process. In the case of this thesis, I will be manually designating the template based on the available ground truth data. If one defines the state update equation using the model in equation (4.1), a standard formulation takes the following form:

$$\mathbf{x}_k = \mathbf{A}\mathbf{x}_{k-1} + \mathbf{u}_k. \tag{4.2}$$

With everything spelled out explicitly, the same equation can be written as follows:

$$\begin{bmatrix} x_{k+1} \\ \dot{x}_{k+1} \\ y_{k+1} \\ \dot{y}_{k+1} \\ \gamma_{k+1} \\ \dot{\gamma}_{k+1} \\ \theta_{k+1} \\ \dot{\theta}_{k+1} \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} 1 & \Delta \\ 0 & 1 \end{bmatrix} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \begin{bmatrix} 1 & \Delta \\ 0 & 1 \end{bmatrix} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \begin{bmatrix} 1 & \Delta \\ 0 & 1 \end{bmatrix} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \begin{bmatrix} 1 & \Delta \\ 0 & 1 \end{bmatrix} \end{bmatrix} \begin{bmatrix} x_k \\ \dot{x}_k \\ y_k \\ \dot{y}_k \\ \gamma_k \\ \dot{\gamma}_k \\ \theta_k \\ \dot{\theta}_k \end{bmatrix} + \begin{bmatrix} 0 \\ u_k^x \\ 0 \\ u_k^y \\ 0 \\ u_k^\gamma \\ 0 \\ u_k^\theta \end{bmatrix}, \tag{4.3}$$

where Δ is the frame time and $u_k^{\cdot}$ are mutually uncorrelated i.i.d. noises that can be interpreted as models for the second derivatives of the appearance variables position, magnification and rotation.

In the method implemented in this thesis, the function $\mathbf{h}$ of equation (2.3) creates an image of same size as the frames of the video sequence filled with zeros and inserts the template $\mathbf{T}$ centered at $[x_k \ y_k]^T$ with a magnification given by γ_k, a rotation given by θ_k and adds the measurement noise $\mathbf{v}_k$. Note that there is no explicit background model in this formulation. For each particle, the hypothesized target appearance is

$$\mathbf{z}_k^i = \mathbf{h}(\mathbf{x}_k^i, 0). \tag{4.4}$$

Let $\mathbf{\Omega}_k^i$ be the spatial support of the template in $\mathbf{z}_k^i$ according to the magnification and rotation of the particle's hypothesis. Then, the likelihood function $p(\mathbf{z}_k|\mathbf{x}_k)$, used in the SIR filter to assign the weight to each particle, is often defined as:

$$p(\mathbf{z}_k|\mathbf{x}_k) = e^{-K(1-\rho_k^i)}, \tag{4.5}$$

where K is a tunable gain and ρ_k^i is a normalized cross correlation defined by:

$$\rho_k^i = \frac{\sum_{\mathbf{\Omega}_k^i}(\mathbf{z}_k - \overline{\mathbf{z}_k})(\mathbf{z}_k^i - \overline{\mathbf{z}_k^i})}{\sqrt{\sum_{\mathbf{\Omega}_k^i}(\mathbf{z}_k - \overline{\mathbf{z_k}})^2 \sum_{\mathbf{\Omega}_k^i}(\mathbf{z}_k^i - \overline{\mathbf{z}_k^i})^2}}, \tag{4.6}$$

where the bar variables denote the mean value of their correspondent variables.

Finally, the estimation of the target's centroid, magnification and rotation is addressed by taking the expected value with respect to equation (3.7)

as follows:

$$\hat{\mathbf{x}}_k = \begin{bmatrix} \hat{x}_k & \hat{y}_k & \hat{\gamma}_k & \hat{\theta}_k \end{bmatrix}^T = \sum_{i=1}^{N_s} \omega_k^i \begin{bmatrix} x_k & y_k & \gamma_k & \theta_k \end{bmatrix}^T. \tag{4.7}$$

The alternative method implemented only as a manner of comparison for the target's centroid calculation is taking the *Maximum A Posteriori* (MAP), *i.e.,* largest-weighted particle prior the resampling step as the estimated target centroid:

$$\begin{bmatrix} \hat{x}_k & \hat{y}_k \end{bmatrix}^T = \begin{bmatrix} x_k^{i^*} & y_k^{i^*} \end{bmatrix}^T, \qquad i^* = \arg\max_i \omega_k^i. \tag{4.8}$$

The SIR filter algorithm described above is used in this work as a comparison for the proposed method and main contribution of this thesis. In the followng section I will describe the proposed method by modifying the SIR filter equations just described to include the velocity information.

4.2 State Vector Augmentation for the SIR Filter

In the previous section I defined the SIR filter formulation that will be used in this work. By focusing our attention on the likelihood function described by equation (4.6) it is easy to note there is not an explicit inclusion of the velocity information. Therefore, there is not a way one could assign a larger weight to a particle that has both a good appearance and velocity hypothesis, than to a particle that only has a good appearance hypothesis.

If one considers that a particle j with state hypothesis $\mathbf{x}_{k-1}^j$ had a good appearance hypothesis, then ρ_{k-1}^j must have been large. Additionally, if particle j has a good velocity hypothesis, then the new state hypothesis $\mathbf{x}_k^j$ should also have a good appearance hypothesis which would mean that ρ_k^j will also be

large. By making both of these conditions necessary to assign a large weight to a particle, an indirect method to include velocity information into the likelihood function can be obtained as long as the hypothesis of particle j and the measurement of the previous time can be retained. A means to do this is using state vector augmentation.

In the state vector augmentation, the augmented system model is defined by a tilde. Let the augmented state vector be defined as:

$$\widetilde{\mathbf{x}}_k = [\mathbf{x}_k \quad \mathbf{x}_{k-1}]^T. \tag{4.9}$$

Then, the redefined state update model will be given by:

$$\widetilde{\mathbf{x}}_{k+1} = \begin{bmatrix} \mathbf{A} & 0 \\ \mathbf{I} & 0 \end{bmatrix} \widetilde{\mathbf{x}}_k + \begin{bmatrix} \mathbf{u}_k \\ 0 \end{bmatrix}, \tag{4.10}$$

where $\mathbf{I}$ is the $8x8$ identity matrix.

The following step is to redefine the measurement model. Let $\breve{\mathbf{z}}_k$ be a second realization of $\mathbf{z}_k$ generated at time step $k+1$ given by:

$$\breve{\mathbf{z}}_k = \mathbf{h}(\mathbf{x}_k, \mathbf{v}_{k+1}). \tag{4.11}$$

The difference between the two realizations is given by the noise difference:

$$\breve{\mathbf{z}}_k - \mathbf{z}_k = \mathbf{v}_{k+1} - \mathbf{v}_k. \tag{4.12}$$

. The augmented measurement vector is:

$$\widetilde{\mathbf{z}}_k = [\mathbf{z}_k \quad \breve{\mathbf{z}}_{k-1}]^T. \tag{4.13}$$

Therefore, the augmented measurement vector can be defined as follows

$$\widetilde{\mathbf{z}}_k = \begin{bmatrix} \mathbf{z}_k \\ \breve{\mathbf{z}}_{k-1} \end{bmatrix} = \widetilde{\mathbf{h}}(\widetilde{\mathbf{x}}_k, \mathbf{v}_k) \equiv \begin{bmatrix} \mathbf{h}(\mathbf{x}_k, \mathbf{v}_k) \\ \mathbf{h}(\mathbf{x}_{k-1}, \mathbf{v}_k) \end{bmatrix}. \tag{4.14}$$

The last step is to redefine the likelihood function according to the state vector augmentation of both the state and measurement model. For an augmented particle i with state vector $\widetilde{\mathbf{x}}_k^i = [\mathbf{x}_k^i \quad \mathbf{x}_{k-1}^i]^T$ and measurement hypothesis $\widetilde{\mathbf{z}}_k^i = [\mathbf{z}_k^i \quad \breve{\mathbf{z}}_{k-1}^i]$, let $\breve{\mathbf{\Omega}}_{k-1}^i$ be the spatial support of the template magnified and rotated in $\breve{\mathbf{z}}_{k-1}^i$. The likelihood function for this particle can be redefined as

$$p(\widetilde{\mathbf{z}}_k|\widetilde{\mathbf{x}}_k^i) = e^{-K(1-\widetilde{\rho}_k^i)}. \tag{4.15}$$

Finally, the normalized cross correlation between the hypothesis of the augmented particle and two consecutive frames from the sequence is given by:

$$\tilde{\rho}_k^i = \frac{\sum_{\mathbf{\Omega_k^i}}(\mathbf{z}_k - \overline{\mathbf{z_k}})(\mathbf{z}_k^i - \overline{\mathbf{z}_k^i}) + \sum_{\breve{\mathbf{\Omega}}_{k-1}^i}(\mathbf{z}_{k-1} - \overline{\mathbf{z}_{k-1}})(\breve{\mathbf{z}}_{k-1}^i - \overline{\breve{\mathbf{z}}_{k-1}^i})}{\sqrt{\sum_{\mathbf{\Omega_k^i}}(\mathbf{z}_k - \overline{\mathbf{z_k}})^2 + \sum_{\breve{\mathbf{\Omega}}_{\mathbf{k-1}}^{\mathbf{i}}}(\mathbf{z}_{k-1} - \overline{\mathbf{z_{k-1}}})^2}\sqrt{\sum_{\mathbf{\Omega_k^i}}(\mathbf{z}_k^i - \overline{\mathbf{z}_k^i})^2 + \sum_{\breve{\mathbf{\Omega}}_{k-1}^i}(\breve{\mathbf{z}}_{k-1}^i - \overline{\breve{\mathbf{z}}_{k-1}^i})^2}}, \tag{4.16}$$

where the bar variables denote the mean value of their correspondent variables. Note that as the actual video frame $\mathbf{z}_{k-1}$ is correlated with the hypothesis $\breve{\mathbf{z}}_{k-1}^i$, the second realization $\breve{\mathbf{z}}_{k-1}$ does not appear in (4.16). Additionally, it is important to understand that this last mentioned varible is only a theoretical concept that allows to define a way of considering a realization of the previous video frame in the current frame likelihood function $p(\widetilde{\mathbf{z}}_k|\widetilde{\mathbf{x}}_k^i)$.

The estimation of $\hat{\mathbf{x}}_k$ is finally obtained by computing (4.7) and the target's centroid is also calculated using (4.8) as a comparison.

Considering all the formulations and implications mentioned before, the proposed method of this thesis is described as a technique to incorporate indirectly velocity information into the likelihood function of the SIR filter. This

method presents a new definition of the particle filter that is more robust than the regular SIR filter because it prevents particles with poor velocity hypotheses from surviving the resampling step.

As mentioned before, when the target appearance changes too fast and there is structured clutter in the background, it is necessary to implement a target appearance model update [58]. In this thesis, I implemented the State Augmentation Technique in two set of synthetic data and two of the longwave infrared (IR) sequences described in [39]. For the two IR sequences, given the target appearance is not changing drastically and the size of the target is decreasing throughout the sequences, it is not necessary to update the target appearance models.

For the synthetic data, the two sequences were developed under different conditions of background. One of the sequences has a simple, uniform background that has a lot of contrast with the target. The other sequence is characterized by having structured clutter that has similarities with the target model in different areas. Considering the simplicity of the first mentioned synthetic data there is no need to use model update. However, for the second mentioned synthetic data, it is crucial to implement an appearance update model to be able to track the moving object. In the following subsection I will explain the implemented appearance model update technique that I implemented in the complex synthetic sequence to improve the tracking performance of both the regular SIR filter and the State Augmented SIR filter.

4.3 Template Update Strategy

The importance of online appearance model updating for target tracking has been shown by many authors [19, 33, 37, 46, 58]. Recently, all the popular visual target tracking algorithms have implemented appearance model update techniques to deal with the target appearance variations throughout the video sequences [58].

In [37], Matthew *et al.* propose an algorithm based on a gradient descent technique where the template was updated using a combination from the first frame template and the result from the most recent frame. Other work, like the one in [29] updates the appearance model using a fixed number of frames. However, it has been proved that this approach is sensitive to drifting [15] because the update might be performed more frequently or later than needed, allowing background leaking into the model.

Considering this, I implemented a simple target appearance update technique that helped improve the tracking performance of the SIR filter and the State Augmented SIR filter in one of the synthetic data sequences that I developed. The algorithm is based on the Normalized Cross-Correlation concept and uses a library of good appearance model candidates from where the template update is extracted in case it is needed.

4.3.1 Generalities of the Developed Update Strategy

The template update method is implemented in each frame after having obtained the hypotheses for all the particles. In the case of the standard SIR filter, for each frame, each particle hypothesizes the target's appearance according to equation (4.4). After all the particles have calculated their hypotheses, the es-

timated state $\hat{\mathbf{x}}_k$ is calculated according to (4.7). In the case of the State Vector Augmented SIR filter the only difference comes when each particle hypothesizes the target's appearance, which in this case is by using (4.14).

The following step is where the algorithm checks the goodness of the current frame estimation. It takes the estimated $\hat{\mathbf{x}}_k$ of the current frame $\mathbf{z}_k$ and calculates the size of the estimated target by taking the size of the initial template $\mathbf{T}$ and adjusting it according to the magnification $\hat{\gamma}_k$ and rotation $\hat{\theta}_k$. Then, it extracts a patch $\widehat{\mathbf{P}}_k$ from the current frame $\mathbf{z}_k$ of the size calculated from the estimation and centered at the estimated centroid $[\hat{x}_k \ \hat{y}_k]^T$. Then it takes a modified version of the template $\widetilde{\mathbf{T}}_k$ rescales it to the size of $\hat{\gamma}_k$ and rotates it according to $\hat{\theta}_k$. Finally it performs the normalized cross-correlation between the two of them according to:

$$\hat{\rho}_k = \frac{\sum_{\mathbf{\Omega}_k}(\widehat{\mathbf{P}}_k - \overline{\widehat{\mathbf{P}}_k})(\widetilde{\mathbf{T}}_k - \overline{\widetilde{\mathbf{T}}_k})}{\sqrt{\sum_{\mathbf{\Omega}_k}(\widehat{\mathbf{P}}_k - \overline{\widehat{\mathbf{P}}_k})^2 \sum_{\mathbf{\Omega}_k}(\widetilde{\mathbf{T}}_k - \overline{\widetilde{\mathbf{T}}_k})^2}}, \tag{4.17}$$

where $\mathbf{\Omega}_k$ is the spatial support of both $\widehat{\mathbf{P}}_k$ and $\widetilde{\mathbf{T}}_k$ in $\mathbf{z}_k$.

After obtaining the normalized cross-correlation coefficient it compares it with a predefined threshold. For the synthetic data used in this thesis this threshold was empirically chosen to be 0.4. If the coefficient is larger than the threshold it means that the current template is still good and the update it is not performed. In case the coefficient is smaller than the threshold the algorithm updates the current template by taking the candidate in the library with the highest frame number. Then the algorithm goes back to the frame where the new template was extracted and starts over from there. In the

following subsection I will explain the process of filling the template candidates library.

4.3.2 Template Library

The template library consists in three possible candidates that have achieved a high coefficient during the normalized cross-correlation process between the estimated target appearance and the current template. For each candidate, the library saves its coefficient value $\hat{\rho}_k$ and its frame number k.

At the beginning of the algorithm the template library is filled using the initial template $\mathbf{T}$ setting the frame number k of all the candidates to $k = 1$ and the coefficient value $\hat{\rho}_k$ to the threshold value 0.4. During the first frames of the video sequence, these initial conditions are replaced. The candidates are ordered according to their coefficient value. Once the algorithm has detected the first patch $\widehat{\mathbf{P}}_k$ that has a coefficient $\hat{\rho}_k$ larger than the threshold, this is set to be the first candidate in the library and both its frame number and coefficient value $\hat{\rho}_k$ are stored for comparison later. Then, the algorithm fills the other two candidates as follows: it takes any patch $\widehat{\mathbf{P}}_k$ that has a coefficient between the value of the first candidate's coefficient or 95% of its value and sets it to be the new second candidate. Then, it takes any patch $\widehat{\mathbf{P}}_k$ with a coefficient value $\hat{\rho}_k$ that is between 90% and 95% of the first candidate's coefficient value and sets it to be the new third candidate.

After these new candidates are set, in order to refresh the library, the algorithm just considers any path $\widehat{\mathbf{P}}_k$ with a coefficient higher than the threshold and compares its value with the three stored values. If that patch's coefficient is larger than the first candidate's or is at least 90% of its value then the patch

is considered a new candidate and it is stored in the correct position of the library.

4.4 Final Considerations

It is very important to note that although in this work, the chosen target appearance model is defined by a template, the proposed state vector augmentation can easily be extended to different appearance models based on histograms or other features, *e.g.,* HOG, SIFT, LBP or others.

In the following chapter I will discuss the experiments conducted in order to show how this new method is more effective for target tracking in video sequences with structured clutter and delivers comparable results in other situations where the standard SIR filter tends to succeed in the tracking process.

Chapter 5

Simulation and Results

In this chapter, I will use both synthetic and real-world infrared video sequences to experimentally demonstrate the SIR filter state vector augmentation technique that I proposed in Chapter 4. I compare the proposed algorithm with the standard SIR filter. Note that both algorithms use the NCC template matching filter and I implemented the template update strategy explained in Section 4.3.1 in both algorithms only in one of the synthetic video sequences. The performances of the algorithms are measured in terms of the Mean Absolute Tracking Error in pixels of the tracking centroid. This error is calculated by using the ground truth data.

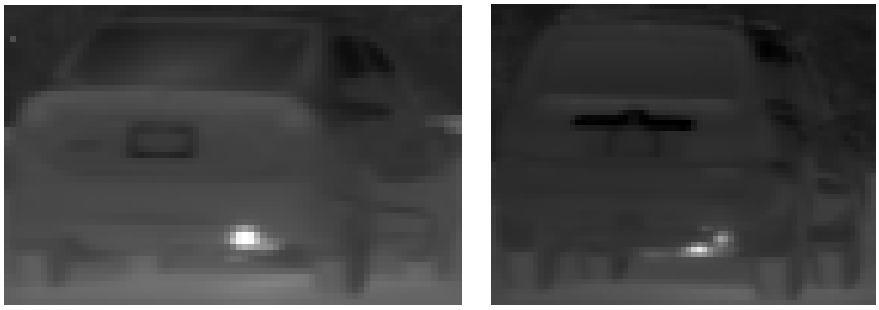
I assume that the initial target size, location and rotation angle are known *a priori* using manual designation [4]. According to this information, I construct the initial template extracting a piece $\mathbf{T}$ of the first frame $\mathbf{z}_k$. For each data sequence I have two independent tracker experiments running. In the first experiment (SIR), the tracker implements a Standard SIR filter in the pixel domain. In the second experiment (VSIR), the tracker implements the State Vector Augmented SIR filter proposed in Section 4.2. Note that both algorithms used a NCC matcher with a static template for all the data sequences except for the synthetic one with high cluttered background for which I included the template update strategy explained in Section 4.3.

(a) (b) (c)

Figure 5.1: Synthetic Data Sequence Base Target and Background Used. (a) Synthetic Target. (b) Simple Background. (c) Complex Background.

The synthetic data sequences were generated by inserting Figure 5.1(a) into two different backgrounds, one benign shown in Figure 5.1(b) and one complex shown in Figure 5.1(c), with trajectory defined by (4.3). The velocity drift noises for the vertical and horizontal coordinates were defined as Gaussian with variances of 0.63 and 0.75 respectively. In the case of the magnification and rotation drift noise, they were set to have a uniform distribution with zero mean and variances of 3.6×10^{-5} and 6.4×10^{-3} respectively for the simple sequence and of 4×10^{-3} and 8×10^{-2} respectively for the difficult one. The length of the benign background sequence was set to 150 frames and the length of the complex background sequence was set to 100 frames.

The two IR sequences evaluated are part of the longwave IR sequences described in [39]. They are part of a series of sequences called Brown Camp. Since the interest was to test the performance of the VSIR for single object tracking, the used IR sequences are subsequences of two cases of the aforementioned series. For them, the ground truth data was compiled manually. The noise variances were estimated by derivative approximation with finite differences on the ground truth data. Although this is a suboptimal approach, it



(a) (b)

Figure 5.2: Real Data Longwave Infrared Targets. (a) Target for Brown Camp 1 Case 3. (b) Target for Brown Camp 3 Case 7.

was used for expediency reasons. The pdfs of the noises were based on the ones chosen for the synthetic data sequences (Normal distribution for the velocities drifts and uniform distributions for the magnification and rotation drifts) but the means and variances were obtained from the ground truth data. The initial position velocity, magnification and rotation were extracted from the first frame of the ground truth data. The targets selected for the two real data sequences are shown in Figure 5.2. The target in Figure 5.2(a) belongs to the Subsequence Brown Camp 1 case 3 (`bc1_case3`) and the one in Figure 5.2(b) is part of the Subsequence Brown Camp 3 case 7 (`bc3_case7`). The length of the brown camp 1 case 3 subsequence is of 125 frames and the length of the brown camp 3 case 7 subsequence is 100 frames.

For all of the studied sequences I utilized a fixed number of particles set to 700. For all the sequences, when initializing the particles for the SIR and VSIR filters, the distribution of the noises for the horizontal and vertical position were set to be Gaussian and uniform for the magnification and rotation noises. For the IR sequences, I extracted the variances and means from the ground truth data using the method described before. For the synthetic

Table 5.1: Summarized Results for Synthetic Video Sequences

Case	Num Frames	Gain K (4.5), (4.15)	Num Runs	Template Update	**Mean Absolute Tracking Error (pixels)** Standar SIR E[·]	Standar SIR MAP	Proposed VSIR E[·]	Proposed VSIR MAP
Benign	150	10	10	No	**1.7505**	2.0090	1.9985	2.4028
Complex	100	10	25	No	14.8441	15.0561	**11.5527**	12.0532
Complex	100	10	25	Yes	2.5871	2.4296	**1.6907**	2.1461

sequences, I set the variances and means to be the same used to generate the videos. The gain K of equations (4.5), (4.15) was set to 10 for the synthetic data sequences and tuned between 75 and 100 for the IR sequences.

5.1 Results

The results of the validation experiments for synthetic sequences are presented in Section 5.1.1, while those for the infrared sequences are presented in Section 5.1.2.

5.1.1 Synthetic Sequences

Table 5.1 shows the tracking results for the synthetic video sequences. The tracking performance is quantified in terms of the centroid estimation accuracy measurements, *e.g.*, mean square error (MSE) using the expected value estimation and the MAP estimation. The table includes the results for both sequences, the number of frame in each one, the number of times I simulated for each sequence and specifies whether the template update strategy was implemented or not.

From Table 5.1 it is evident that incorporating velocity information into the likelihood function does not have a noticeable effect when the background is benign. This can be evident by looking at the frame by frame results obtained

for one of the runs shown in Figure 5.3. In this case, both filters have a good estimation and are able to track the target throughout the sequence.

For the complex background, the results of Table 5.1 show that incorporating velocity information has a substantial effect on the tracking performance. In fact, even when implementing a template update strategy the proposed VSIR algorithm provides better tracking results. In Figure 5.4, I show the graphical frame-wise tracking results when there is no template update strategy implemented. It is evident that both trackers fail to provide a good estimation of the target's position throughout the sequence. However, the VSIR filter shows a better estimation compared to the one obtained from the SIR filter.

The results of Table 5.1 and Figure 5.5 show the remarkable improvement in both tracking algorithms by including the Template Update Strategy, as expected, since there is a highly structured background and the target appearance is changing considerably through the sequence. Nevertheless, even in this case, the proposed VSIR algorithm outperforms the Standard SIR filter.

Notice that the background presents areas with structures that resemble the target appearance. Thus, when the target passes through these areas, false alarms cause the filters to drift. If a template update strategy is not implemented, the trackers eventually loose track. However, even when I implemented the template update strategy, in the cases where the updating appearance model presents background leaking in it the tracker will still show drifting but will have a better performance than without any update strategy.

In Figure 5.5, I selected three frames, 73, 77 and 81, to show the progression of both filters within these frames. From these images, it is evident

Table 5.2: Summarized Results for Longwave Infrared Video Sequences

Case	Num Frames	Gain K (4.5), (4.15)	Num Runs	Mean Absolute Tracking Error (pixels)			
				Standar SIR		Proposed VSIR	
				E[·]	MAP	E[·]	MAP
bc1_case3	180	75	15	3.8244	3.8861	**3.5183**	3.6090
bc3_case7	125	75	25	7.0381	7.1336	**2.8784**	2.9743
bc3_case7	125	100	25	5.6847	6.6511	**2.7767**	2.8202

that when the SIR filter starts drifting away, the target appearance model affects the estimation all the way through the end of the sequence. However, the VSIR filter is able to overcome the drifting thanks to the velocity information.

In general, our results show that there is better performance when taking the estimation according to the expected value than when considering the MAP estimation.

5.1.2 Infrared Sequences

The tracking results for the longwave infrared video sequences are shown in Table 5.2. As mentioned for the synthetic video sequences, the tracking performance is quantified in terms of the mean square error (MSE) using the expected value estimation and the MAP estimation. The table includes the results for both sequences, the number of frame in each one and the number of times I simulated for each sequence. Recall that as mentioned before I did not implement a Template Update Strategy in the sequences. One of the main reasons for not implementing an appearance model update was to isolated the performance differences between the Standard SIR and the proposed VSIR filters.

The results of Table 5.2 show that for the real data sequences, the proposed VSIR filter always exhibits a better performance than the standard SIR. Additionally, the results prove that the expected value estimation has a

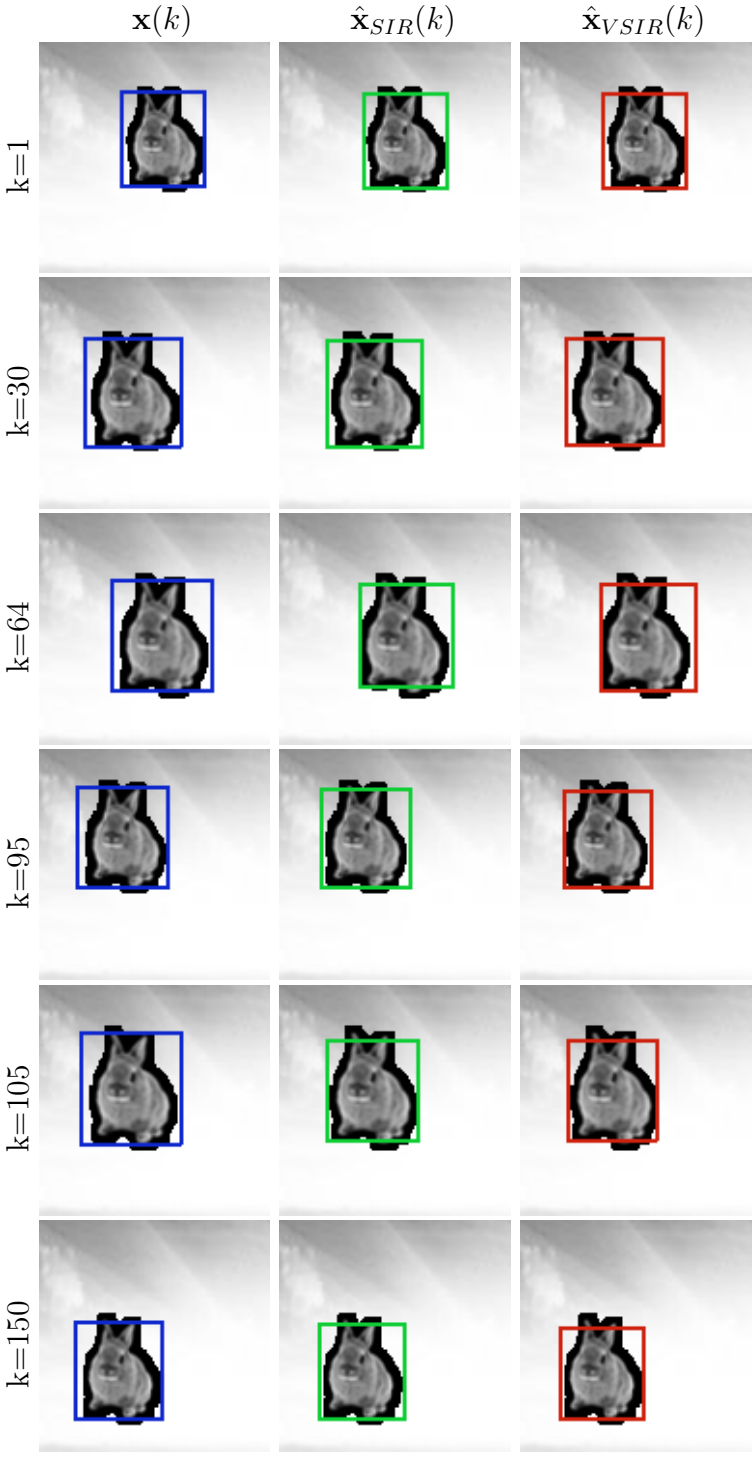


Figure 5.3: Frame-wise Tracking Comparison of the Two Evaluated Methods for the Benign Background Synthetic Sequence. The First Column is the Ground Truth Data. The Second and Third Columns show the Estimation Obtained from the SIR and Proposed VSIR Filters respectively.

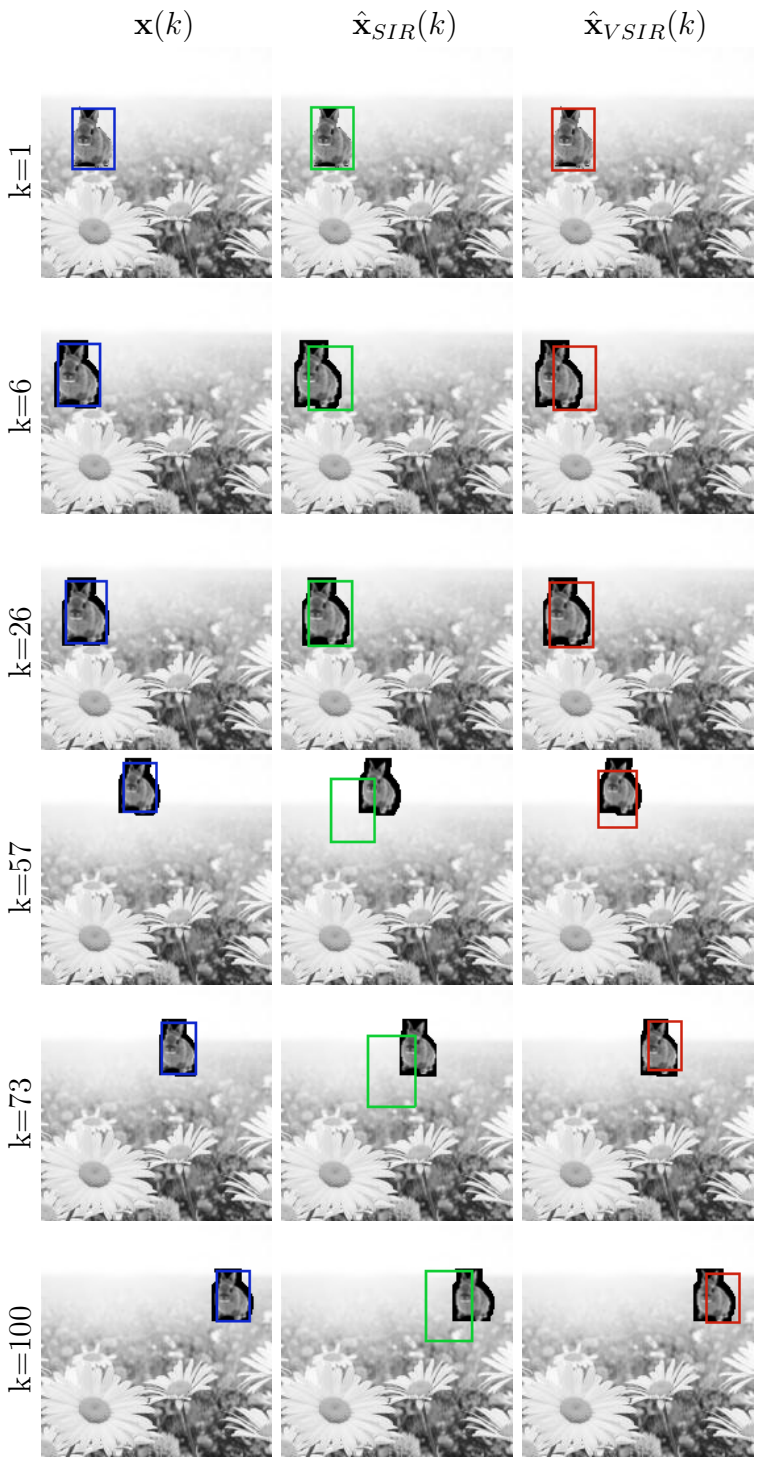


Figure 5.4: Frame-wise Tracking Comparison of the Two Evaluated Methods for the Complex Background Sequence without Template Update. The First Column is the Ground Truth Data. The Second and Third Columns show the Estimation Obtained from the SIR and Proposed VSIR Filters respectively.

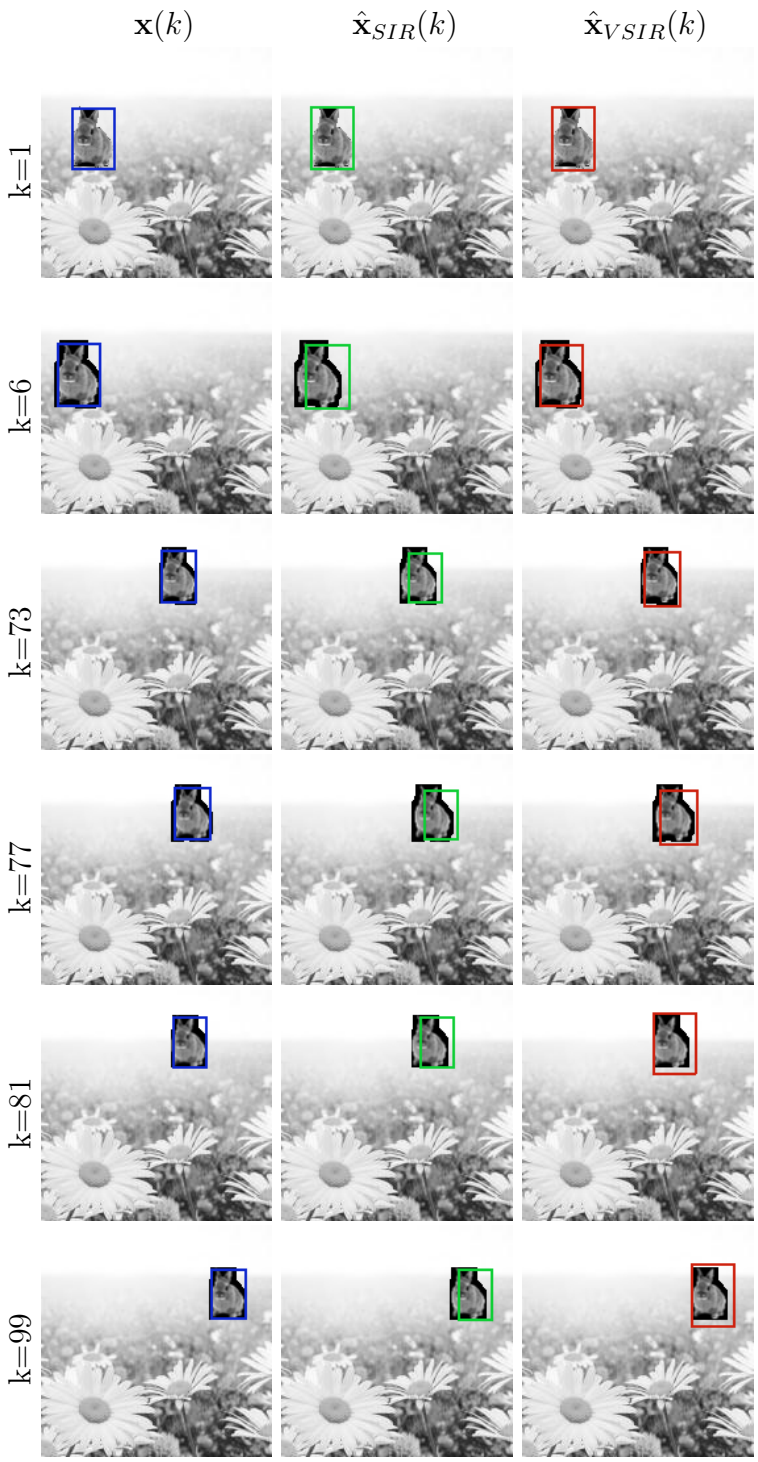


Figure 5.5: Frame-wise Tracking Comparison of the Two Evaluated Methods for the Complex Background Sequence with Template Update. The First Column is the Ground Truth Data. The Second and Third Columns show the Estimation Obtained from the SIR and Proposed VSIR Filters respectively.

lower Mean Absolute Error through the sequence. Another important thing to notice from the results given in the table is that although the standard SIR filter is able to track the moving object through the most complex sequence, `bc3_case7`, its estimation is very poor compared to the one obtained by the VSIR filter. Additionally, the results show that by augmenting the tunable gain in this sequence, the results obtained are better.

Figure 5.6 shows the frame-wise tracking performance for the infrared data sequence `bc1_case3`. The images show that as the the target decreases of size considerably, the standard SIR filter starts having greater difficulties than the proposed VSIR to estimated the position and appearance model of it. However, as these sequence is very simple, *i.e.*, there is no occlusion, and the target changes are very slow, both filters exhibit a good performance and are able to track the car.

Figure 5.7 shows the frame-wise tracking performance for the infrared data sequence `bc3_case7` when using a gain $K = 75$. Note that this sequence is more challenging than the `bc1_case3` one since it presents other moving objects in the scene and partial occlusion of the target. Additionally, the target is moving fast which cause a more abrupt size change causing a quicker lost of texture. From these images, it is evident that once the target is partially occluded by the following vehicle the standard SIR filter starts drifting away, showing a bad performance already in frame $k = 85$ and being completely off the target by frame $k = 109$ the tracker is already completely off the target. On the other hand, the proposed VSIR filter does not get affected by the partial occlusion, in fact is able to keep tracking all the way through the end of the sequence.

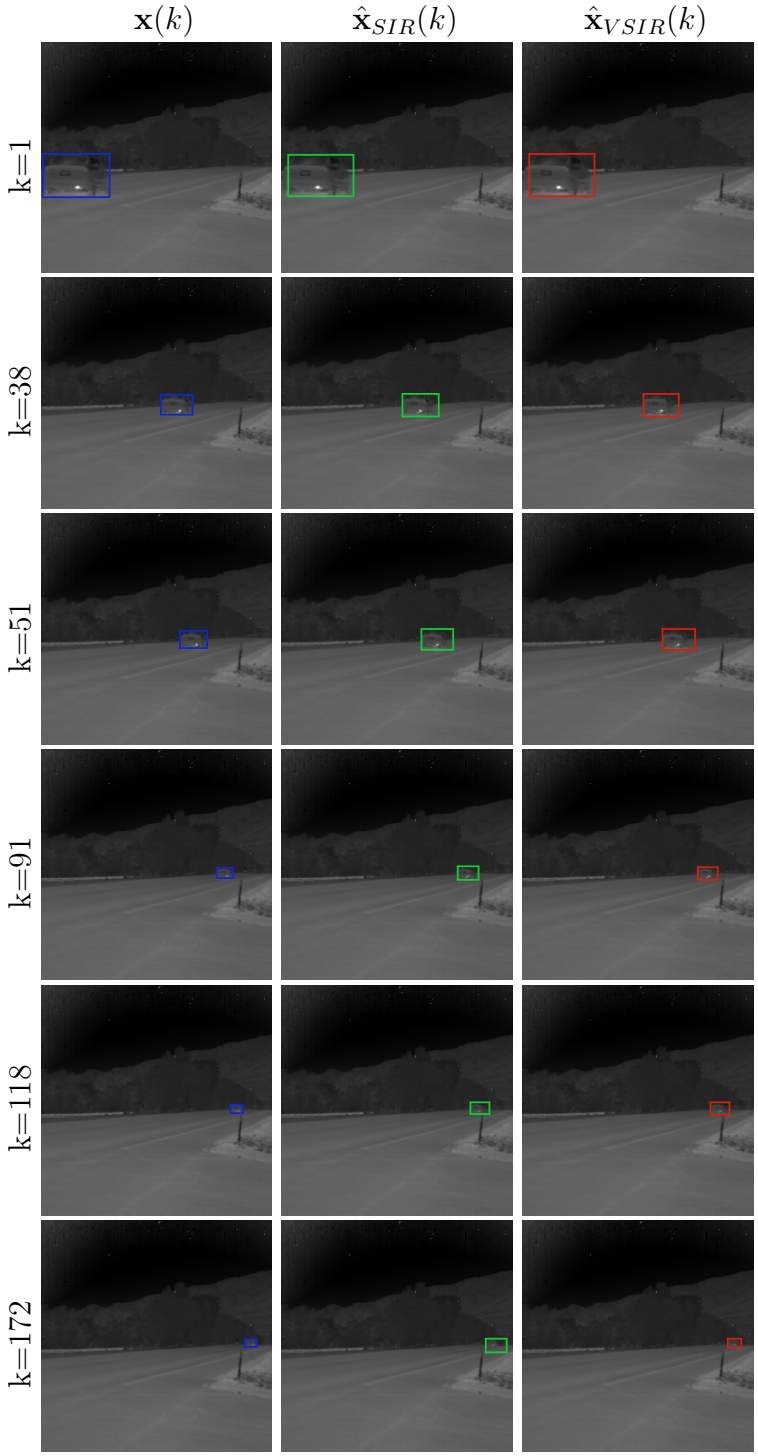


Figure 5.6: Frame-wise Tracking Comparison of the Two Evaluated Methods for the `bc1_case3` Sequence with Gain $K = 75$. The First Column is the Ground Truth Data. The Second and Third Columns show the Estimation Obtained from the SIR and Proposed VSIR Filters respectively.

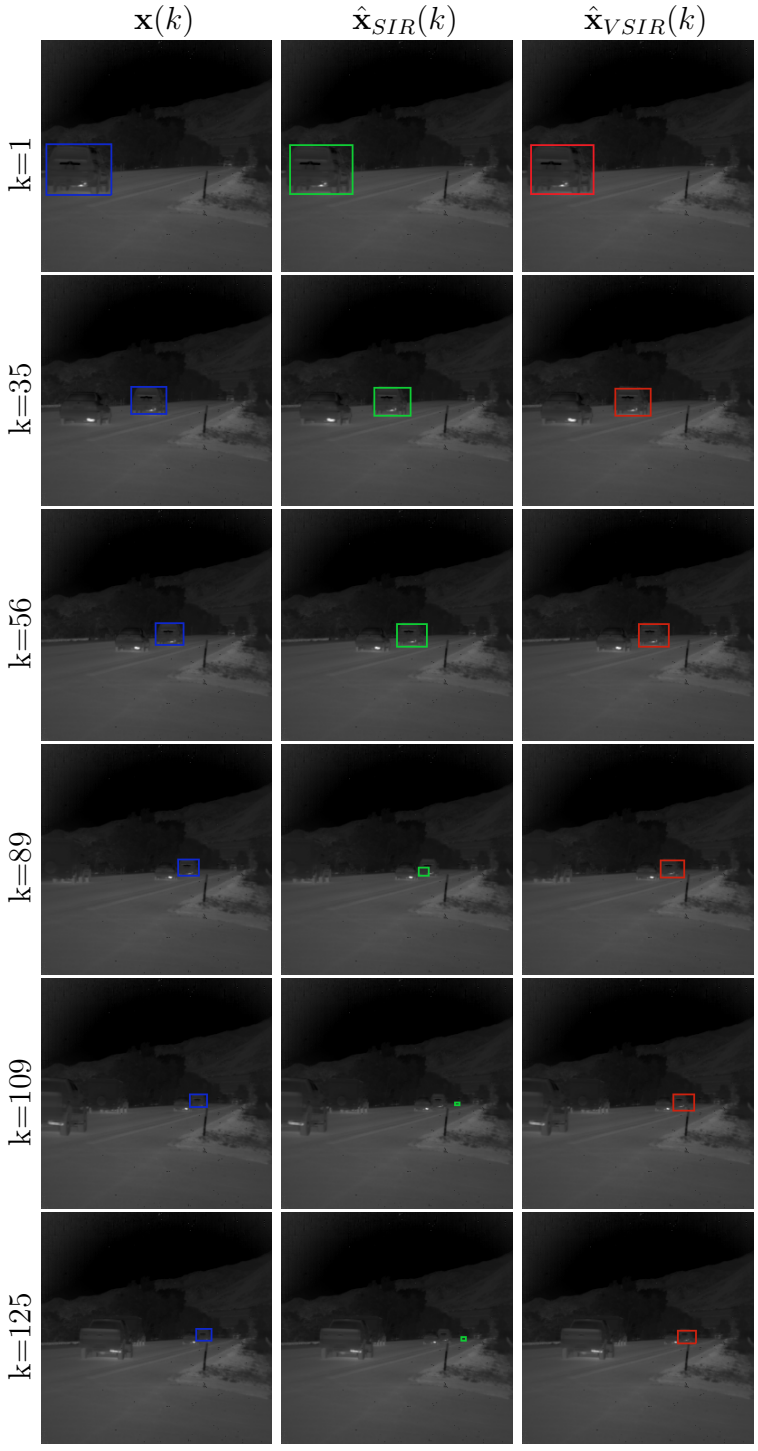


Figure 5.7: Frame-wise Tracking Comparison of the Two Evaluated Methods for the `bc3_case7` Sequence with Gain $K = 75$. The First Column is the Ground Truth Data. The Second and Third Columns show the Estimation Obtained from the SIR and Proposed VSIR Filters respectively.

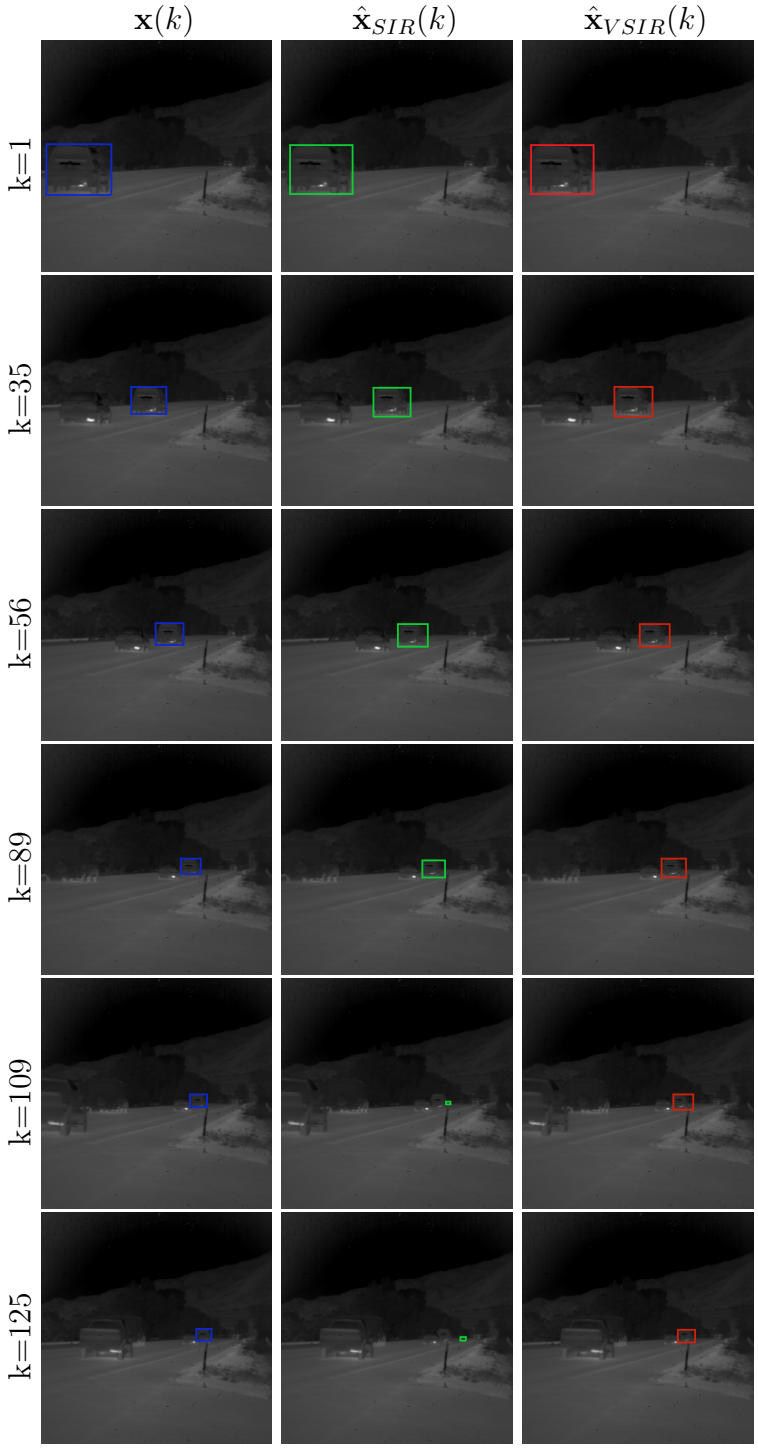


Figure 5.8: Frame-wise Tracking Comparison of the Two Evaluated Methods for the `bc3_case7` Sequence with Gain $K = 100$. The First Column is the Ground Truth Data. The Second and Third Columns show the Estimation Obtained from the SIR and Proposed VSIR Filters respectively.

Figure 5.8 shows the frame-wise tracking performance for the same data sequence when the gain is set to $K = 100$. In this case, the standard SIR manages to keep track of hte target for a longer period but, it is evident that once the target is partially occluded by the following vehicle the standard SIR filter starts drifting away later than in the previous case but, is still off the target by frame $k = 109$. The proposed VSIR filter performs similar to the previous case but the estimated magnification is closer to the ground truth data.

5.2 Final Considerations

In this chapter, I showed the tracking results obtained by implementing the standard SIR and proposed VSIR methods in two different type of sequences, synthetic and real longwave infrared. As expected, the proposed algorithm outperforms the standard SIR in all the challenging conditions and gives comparable results when the background is simple and the target appearance does not vary drastically. Although the algorithm achieves good results in the studied sequences, its performance needs to be tested with more complex sequences both in the visual and infrared spectrum and with targets with a highly changing appearance which will require to implement a robust appearance update model. Additionally, it requires an improvement in computing efficiency since it currently can track targets with approximately one frame per second without implementing any kind of parallel programming.

Chapter 6

Conclusions and Future Work

This thesis is focused on the target tracking problem and its main contribution is a State Vector Augmentation for the SIR filter to include the Velocity Information into its likelihood function. In Chapter 1 I introduced the problem. In Chapter 2, I covered general concepts regarding the target tracking problem including the two different main approaches: State Space and Feature Tracking. In Chapter 3, I reviewed the theory behind the particle filter, its most popular variations and some variations that allowed to provide a justification for including velocity information into the estimation of the particle filters to improve its performance. In Chapter 4, I covered generalities in the calculations of the SIR filter and introduced and defined the proposed State Vector Augmented SIR filter. Finally, in Chapter 5 I tested the performance of the proposed algorithm.

The original contributions of this work are listed below:

- I developed the theoretical formulation for the State Vector Augmented SIR filter algorithm.
- I reformulated the normalized cross correlation used in the Likelihood function of the SIR filter to include the velocity information in it.
- I developed an algorithm to generate synthetic data sequences with tar-

gets that can change both in magnification and rotation for testing the goodness of tracking algorithms in a controlled environment.

- I developed a simple template update strategy to deal with changes in the appearance model of the targets analyzed.

- I prove the effectiveness of the proposed algorithm with tracking results obtained from two longwave infrared sequences and two synthetic datasequences.

The most important contribution of this thesis is the mathematical formulation of the State Vector Augmentation for the likelihood function of the SIR filter (VSIR). The algorithm modifies the normalized cross correlation formula used in the likelihood function of the SIR filter of (4.6) to include the velocity information as shown in (4.16). The proposed method is able to track targets efficiently in situations of partial occlusion and rapid changes of the target appearance. The VSIR tracking performance is superior to that of the standard SIR filter in challenging conditions and exhibits a similar behavior in benign data sequences.

Despite its good performance, the algorithm needs to be tested in other visual and infrared video sequences that require the implementation of an appearance update model more robust than the one implemented in this work. Additionally, future work should include comparison of its performance with other state of the art target tracking methods, such as those covered by [33, 43, 58]. Future work should also include improving the algorithm for faster execution time, so the model can be fairly compared with different methods in the civilian target tracking community.

Bibliography

[1] D. B. R. A. P. Dempster, N. M. Laird, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.

[2] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking," *IEEE Trans. Signal Process.*, vol. 2, no. 2, p. 174188, Feb 2002.

[3] B. Babenko, M. H. Yang, and S. Belongie, "Visual tracking with online multiple instance learning," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, Jun 2009, pp. 983–990.

[4] A. Bal and M. S. Alam, "Automatic target tracking in flir image sequences using intensity variation function and template modeling," *IEEE Trans. Instrum., Meas.*, vol. 54, no. 5, p. 18461852, Oct 2005.

[5] Y. Bar-Shalom and T. E. Fortmann, *Tracking and Data Association*. Academic Press, Inc., 1988.

[6] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*. John Wiley & Sons, Inc., 2002.

[7] J. T. Barnett, B. D. Billard, and C. Lee, "Nonlinear morphological processors for point-target detection versus an adaptive linear spatial filter: a performance comparison," in *Proc. SPIE, Signal and Data Processing of Small Targets*, Orlando, FL, 1993, vol. 1954, pp. 12–24.

[8] M. Bruno, "Bayesian methods for multispectral target tracking in image sequences," *IEEE Transactions on Signal Processing*, vol. 52, no. 7, pp. 1848–1861, July 2004.

[9] T. J. Cham and J. M. Rehg, "A multiple hyphothesis approach to figure tracking," in *Proc. IEEE Conf. Comput. Vision Pattern Recog.*, vol. 2, Fort Collins,CO, June 1999, pp. 239–245.

[10] T. Clapp, "Statistical methods for the processing of communications data," Ph.D. dissertation, Dept. of Engineering, University of Cambridge, Cambridge, U.K., 2000.

[11] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 24, no. 5, pp. 603–619, May 2002.

[12] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 5, no. 25, pp. 564–577, May 2003.

[13] S. Dubuisson and C. Gonzales, "A survey of datasets for visual tracking," *Machine Vision and Applications*, vol. 27, no. 1, pp. 23–52, 2016.

[14] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification (2Nd Edition)*. Wiley-Interscience, 2000.

[15] B. Fan, Y. Du, Y. Cong, and Y. Tang, "Active drift correction template tracking algorithm," in *2012 19th IEEE Internaional Conference on Image Processing*, September 2012, pp. 397–400.

[16] A. Gning, B. Ristic, L. Mihaylova, and F. Abdallah, "An introduction to box particle filtering [lecture notes]," *IEEE Signal Processing Magazine*, vol. 30, no. 4, pp. 166–171, July 2013.

[17] J. Gong, G. Fan, L. Yu, J. P. Havlicek, and D. Chen, "Joint view-identity manifold for target tracking and recognition," in *2012 19th IEEE International Conference on Image Processing*, Sept 2012, pp. 1357–1360.

[18] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/non-gaussian bayesian state estimation," *IEE Proceedings F - Radar and Signal Processing*, vol. 140, no. 2, pp. 107–113, April 1993.

[19] S. Hare, A. Saffari, and P. H. S. Torr, "Struck: Structured output tracking with kernels," in *2011 International Conference on Computer Vision*, Nov 2011, pp. 263–270.

[20] K. Hariharakrishnan and D. Schonfeld, "Fast object tracking using adaptive block matching," *IEEE Transactions on Multimedia*, vol. 7, no. 5, pp. 853–859, Oct 2005.

[21] I. Haritaoglu, D. Harwood, and L. S. Davis, "W4: Real-time surveillance of people and their activities, ieee trans. pattern anal. machine intell." *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 22, no. 8, p. 809830, Aug 2000.

[22] Y. Huang and J. Llach, "Tracking the small object through clutter with adaptive particle filter," in *Proc. Intl. Conf. Audio, Language, Image Process.*, July 2008, pp. 357–362.

[23] Y.-W. Huang, C.-Y. Chen, C.-H. Tsai, C.-F. Shen, and L.-G. Chen, "Survey on block matching motion estimation algorithms and architectures with new results," *Journal of VLSI signal processing systems for signal, image and video technology*, vol. 42, no. 3, pp. 297–320, March 2006.

[24] M. Isard and A. Blake, "Condensation - - conditional density propagation for visual tracking," *International Journal of Computer Vision.*, vol. 29, no. 1, pp. 5–28, 1998.

[25] S. J. Julier and J. K. Uhlmann, "A new extension of the kalman filter to nonlinear systems," *SPIE*, vol. 3068, p. 182193, 1997.

[26] Z. Kalal, J. Matas, and K. Mikolajczyk, "P-n learning: Bootstrapping binary classifiers by structural constraints," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, June 2010, pp. 49–56.

[27] R. Kalman, "A new approach to linear filtering and prediction problems," *Trans. of ASME - Journal of Basic Enginering*, pp. 35–45, March 1960.

[28] B. V. K. V. Kumar and L. Hassebrook, "Performance measures for correlation filters," *Appl. Opt.*, vol. 29, no. 20, pp. 2997–3006, Jul 1990.

[29] S. Lankton, J. Malcolm, A. Nakhmani, and A. Tannenbaum, "Tracking through changes in scale," in *2008 15th IEEE Internaional Conference on Image Processing*, October 2008, pp. 241–244.

[30] L. Laymarie and M. D. Levine, "Tracking deformable objects in the plane using an active contour model," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 15, no. 6, p. 617634, June 1993.

[31] X. R. Li and V. Jilkov, "Survey of maneuvering target tracking. part i dynamic models," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, no. 4, pp. 1333–1364, October 2003.

[32] H. Liu and F. Sun, "Efficient visual tracking using particle filter with incremental likelihood calculation," *Information Sciences*, vol. 195, pp. 141 – 153, 2012.

[33] Q. Liu, X. Zhao, and Z. Hou, "Survey of single-target visual tracking methods based on online learning," *IET Computer Vision*, vol. 8, no. 5, pp. 419–428, October 2014.

[34] W. Liu, A. B. Chan, R. W. H. Lau, and D. Manocha, "Leveraging long-term predictions and online learning in agent-based multiple person tracking," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, no. 3, pp. 399–410, March 2015.

[35] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. IEEE Intl. Joint Conf. Artificial Intell.*, August 1981, pp. 674–679.

[36] D. Manjunatha and D. Sainarayanan, "Comparison and implementation of fast block matching motion estimation algorithms for video compression," *International Journal of Engineering Science and Technology*, vol. 3, no. 10, pp. 7608–7613, Oct 2011.

[37] I. Matthews, T. Ishikawa, and S. Baker, "The template update problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 6, pp. 810–815, June 2004.

[38] N. S. Michael K. Pitt, "Filtering via simulation: Auxiliary particle filters," *Journal of the American Statistical Association*, vol. 94, no. 446, pp. 590–599, 1999.

[39] C. Nguyen, J. Havlicek, G. Fan, J. Caulfield, and M. Pattichis, "Robust dual-band mwir/lwir infrared target tracking," in *2014 48th Asilomar Conference on Signals, Systems and Computers*, Nov 2014, pp. 78–83.

[40] J. M. Odobez, D. Gatica-Perez, and S. O. Ba, "Embedding motion in model-based stochastic tracking," *IEEE Trans. Image Process.*, vol. 15, no. 11, p. 35153531, Nov 2006.

[41] P. Pan and D. Schonfeld, "Visual tracking using high-order particle filtering," *IEEE Signal Processing Letters*, vol. 18, no. 1, pp. 51–54, Jan 2011.

[42] C. G. R. Cucchiara, M. Piccard, and A. Prati, "Detecting moving objects, ghosts, and shadows in video streams," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 25, no. 10, p. 13371342, Oct 2003.

[43] G. M. Rao and C. Satyanarayana, "Visual object target tracking using particle filter: A survey," *International Journal of Image, Graphics and Signal Processing*, vol. 5, no. 6, pp. 57–71, 2013.

[44] C. Rasmussen and G. D. Hager, "Probabilistic data association methods for tracking complex visual objects," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 23, no. 6, p. 560576, Jun 2001.

[45] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental learning for robust visual tracking," *International Journal of Computer Vision*, vol. 77, no. 1, pp. 125–141, 2008.

[46] S. Salti, A. Cavallaro, and L. Stefano, "Adaptive appearance modeling for video tracking: Survey and evaluation," *IEEE Transactions on Image Processing.*, vol. 21, no. 10, pp. 4334–4348, October 2012.

[47] C. Schmid, R. Mohr, and C. Bauckhage, "Evaluation of interest point detectors," *International Journal of Computer Vision*, vol. 37, no. 2, p. 151172, 2000.

[48] T. Schn, F. Gustafsson, and P. Nordlund, "Marginalized particle filters for mixed linear/nonlinear state-space models," *IEEE Transactions on Signal Processing*, vol. 53, no. 7, pp. 2279–2289, 2005.

[49] D. Serby, E. K. Meier, and L. V. Gool, "Probabilistic object tracking using multiple features," in *Proc. IEEE Intl. Conf. Pattern Recog.*, Cambridge, UK, August 2004, vol. 2, p. 184187.

[50] S. H. Shaikh, K. Saeed, and N. Chaki, *Moving Object Detection Using Background Subtraction.* Springer International Publishing, 2014.

[51] J. Shi and C. Tomasi, "Good features to track," in *Proc. IEEE Conf. Comput. Vision Pattern Recog.*, Jun 1994, p. 593600.

[52] C. Stauffer and W. Grimson, "Adaptive background mixture models for realtime tracking," in *Proc. IEEE Conf. Comput. Vision Pattern Recog.*, Fort Collins, CO, Jun 23-25 1999, vol. 2, pp. 246–252.

[53] J. Sullivan and J. Rittscher, "Guiding random particles by deterministic search," in *Computer Vision, 2001. ICCV 2001. Proceeding. Eighth IEEE International Conference on*, vol. 1, 2001, pp. 323–330.

[54] T. Tangsukson and J. P. Havlicek, "Am-fm image segmentation," in *Proc. IEEE Intl. Conf. Image Processing*, Vancouver, Canada, Sept 2000, p. 104107.

[55] V. T. Tom, T. Peli, M.Leung, and J. E. Bondaryk, "Morphology-based algorithm for point target detection in infrared backgrounds," in *Proc. SPIE, Signal and Data Processing of Small Targets*, Orlando, FL, 1993, vol. 1954, pp. 2–11.

[56] Q. Wang, F. Chen, W. Xu, and M. Yang, "Object tracking via partial least square analysis," *IEEE Transactions on Image Processing*, vol. 21, no. 10, pp. 4454–4465, 2004.

[57] T. Wang, I. Y. H. Gu, and P. Shi, "Object tracking using incremental 2d-pca learning and ml estimation," in *IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07*, vol. 1, April 2007, pp. I–933–I–936.

[58] Y. Wu, J. Lim, and M. H. Yang, "Online object tracking: A benchmark," in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, June 2013, pp. 2411–2418.

[59] Y. Yao and R. Chellappa, "Tracking a dynamic set of feature points," *IEEE Trans. Image Processing*, vol. 4, no. 10, p. 13821385, Oct 1995.

[60] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey." *ACM Computing Surveys*, vol. 38, no. 4, pp. 1 – 45, 2006.

[61] A. Yilmaz, K. Shafique, and M. Shah, "Target tracking in airborne forward looking infrared imagery," *Image and Vision Computing*, vol. 21, no. 7, pp. 623 – 635, 2003.

[62] A. Zheng, Y. Yuan, S. Jaiswal, and O. Au, "Image processing(icip), 2015 ieee international conference on," in *Motion estimation via hierarchical block matching and graph cut*, 2015, pp. 4371–4375.

[63] S. K. Zhou, R. Chellappa, and B. Moghaddam, "Visual tracking and recognition using appearance-adaptive models in particle filters," *IEEE Trans. Image Process.*, vol. 13, no. 11, pp. 1491–1506, 2004.