

© Copyright by Shawn Michael Herman, 2002

A PARTICLE FILTERING APPROACH TO JOINT PASSIVE RADAR TRACKING
AND TARGET CLASSIFICATION

BY

SHAWN MICHAEL HERMAN

B.S., University of Illinois, 1994

M.S., University of Illinois, 1996

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2002

Urbana, Illinois

ABSTRACT

In this thesis, we present a recursive Bayesian solution to the problem of joint tracking and classification for ground-based air surveillance. In our system, we specifically allow for complications due to multiple targets, false alarms, and missed detections. Most importantly, though, we utilize the full benefit of a joint approach by implementing our tracker using an aerodynamically valid flight model that requires aircraft-specific coefficients such as the wing area, minimum drag, and vehicle mass. Of course, these coefficients are provided to our tracker by our classifier.

The key feature that bridges the gap between tracking and classification is radar cross section (RCS), which we include in our measurement vector. By modeling the true deterministic relationship that exists between RCS and target aspect, we are able to gain both valuable class information and an estimate of target orientation. However, the lack of a closed-form relationship between RCS and target aspect prevents us from using the Kalman filter or any of its variants. Instead, we rely upon a sequential Monte Carlo-based approach known as particle filtering. In addition to allowing us to include RCS as a component in our measurement vector, the particle filter also simplifies the implementation of our nonlinear non-Gaussian flight model.

Thus, we believe that we are the first to provide a joint tracking/classification framework that realizes the full potential of such an approach. Our joint formulation consists of three key developments: (1) an aerodynamically valid flight model that relies upon aircraft-specific coefficients such as the wing area and the minimum value of drag, (2) an electromagnetically correct model for RCS that yields information pertaining to both class identity and target orientation, and (3) a particle filter-based implementation that takes into account realistic difficulties caused by multiple targets, false alarms, and missed detections.

ACKNOWLEDGMENTS

This work was supported in part by an NSF graduate fellowship and grants from DARPA (contract AFOSR F49620-98-1-0498) and the NATO Consultation, Command, and Control Agency (NC3A), Sensors and Surveillance Branch. I would like to offer special thanks to the following people for their support during the past few years: Dr. Paul Howland at NC3A for allowing this research to continue, Dick Lodwig and Dr. Bert Bradford at Lockheed Martin for sharing their expertise in FM-band passive radar, Prof. Aaron Lanterman at Georgia Tech for introducing me to the wonderful world of radar, and Dr. Ben Slocumb at Numerica, Inc., for providing the F-4E Phantom statistics and plenty of good advice. Finally, I would like to offer sincere gratitude to my advisor, Prof. Pierre Moulin, for his help, encouragement, and friendship over the past four years.

TABLE OF CONTENTS

CHAPTER 1 INTRODUCTION	1
CHAPTER 2 BACKGROUND IN ELECTROMAGNETICS	5
2.1 Ground-Based FM-Band Passive Radar	5
2.2 Radar Cross Section	6
2.3 Method of Moments	8
2.4 RCS Databases and Sampling Considerations	10
CHAPTER 3 RECURSIVE BAYESIAN FILTERING	15
3.1 Exact Recursive Bayesian Algorithms	20
3.1.1 The HMM filter	20
3.1.2 The Kalman filter	21
3.2 Suboptimal Recursive Bayesian Algorithms	22
3.2.1 Single Gaussian approximations	22
3.2.2 Sum of basis functions	28
CHAPTER 4 MONTE CARLO-BASED APPROACHES TO BAYESIAN FILTERING	31
4.1 Sampling from an Arbitrary Distribution	32
4.1.1 Rejection sampling	33
4.1.2 Metropolis-Hastings algorithm	34
4.1.3 Importance sampling	35
4.2 Sequential Importance Sampling	38
4.3 Degeneracy of the SIS Algorithm	39
4.3.1 Choice of the proposal distribution	41
4.3.2 Resampling	43
4.3.3 Generic particle filter	47
4.3.4 Auxiliary particle filter	48
CHAPTER 5 KALMAN FILTER-BASED TRACKING	51
5.1 State Models for the IMM-EKF	52
5.1.1 Constant velocity model	53
5.1.2 Constant acceleration model	53
5.1.3 Coordinated turn model	53
5.2 Measurement Model for the IMM-EKF	56
5.3 Data Association for the IMM-EKF	58
5.3.1 Probabilistic data association	58
5.3.2 PDA filter	61
5.3.3 JPDA filter	63

CHAPTER 6	PARTICLE FILTER-BASED TRACKING AND CLASSIFICATION	67
6.1	State Model for the Particle Filter	67
6.1.1	Reference frames and aerodynamic angles	68
6.1.2	Modeling the translational motion of an airplane	70
6.1.3	Modeling the rotational motion of an airplane	73
6.1.4	The complete flight model	76
6.2	Measurement Model for the Particle Filter	79
6.3	Data Association for the Particle Filter	81
6.3.1	PDA particle filter	82
6.3.2	Joint data association for particle filters	83
CHAPTER 7	EXPERIMENTAL RESULTS	87
7.1	Preliminaries	87
7.1.1	Performance metrics	90
7.1.2	Track maintenance	91
7.1.3	Generation of synthetic RCS data	92
7.2	Tracking Results for a Single Target	95
7.2.1	Tracking performance of the IMM extended Kalman filter	95
7.2.2	Tracking performance of the particle filters	98
7.3	Tracking Results for a Single Target in Clutter	105
7.3.1	Results for the IMM extended Kalman filter	107
7.3.2	Results for the particle filters	109
7.4	Classification Results for a Single Target	111
7.4.1	Simulations using three classes	112
7.4.2	Simulations using seven classes	113
7.5	Multitarget Tracking and Classification Results	114
CHAPTER 8	CONCLUSIONS	121
APPENDIX A	DEGENERACY OF IMPORTANCE WEIGHTS	123
APPENDIX B	DERIVATION OF EFFECTIVE SAMPLE SIZE	125
REFERENCES		131
VITA		137

LIST OF FIGURES

2.1: HH-polarized bistatic RCS for a VFY-218 fighter and a Falcon-20 commercial jet at 99.9 MHz and zero degrees incident and observed elevation. The color scale for both plots (from blue to red) is $[-26.55, 35.63]$ dBsm.	11
2.2: (a) Example of a positive elevation angle (EL) on the Cartesian axes. (b) Example of a positive azimuth angle (AZ) on the Cartesian axes.	11
3.1: Example of IMM algorithm with $N = 2$	25
4.1: A simple example of rejection sampling. The sample $x^{(i)}$ is drawn from a uniform proposal distribution, $\pi(x z) = \mathbb{U}_{[0, x_{max}]}(x)$. The pink bars indicate the probabilities of acceptance and rejection.	33
4.2: Pseudo-code description of the sequential importance sampling algorithm.	40
4.3: Example of how use of the prior $p(\mathbf{x}_k \mathbf{x}_{k-1}^{(i)})$ as proposal distribution can lead to degeneracy of the importance weights.	42
4.4: Pseudo-code description of the systematic resampling algorithm.	45
4.5: Illustration of systematic resampling for $N_p = 3$. The locations of the dashed lines indicate which states have been selected, in this case $\{\tilde{\mathbf{x}}_k^{(1)}, \tilde{\mathbf{x}}_k^{(3)}, \tilde{\mathbf{x}}_k^{(3)}\}$	46
4.6: Pseudo-code description of a generic particle filtering algorithm.	47
4.7: Pseudo-code description of the auxiliary particle filter with $\lambda_k^{(i)} \sim p(\mathbf{x}_k \mathbf{x}_{k-1}^{(i)})$	49
5.1: Example of IMM coordinated turn maneuver.	54
5.2: Illustration of measurement gates and the corresponding validation matrix.	64
6.1: (a) Example of a receiver-centered, Earth-fixed (inertial) frame. (b) Example of a body-centered frame. (c) Definition of the angle of attack α and the sideslip angle β	68
6.2: Illustration of forces relevant to flight. The cross-wind component is not shown. (It would be directed perpendicular to the page.)	71
6.3: Force diagram during a constant-speed coordinated turn.	75
6.4: Pseudo-code description of our state model for flight.	77
6.5: Pseudo-code description of the state integration routine.	78
6.6: Example of airplane response to turn-rate commands from the pilot. $\{\omega_k^*\}$ is the command sequence; $\{\omega_k\}$ is the sequence of actual turn-rates achieved by the aircraft.	79

7.1:	Top row: trajectories for the roll-roll maneuver and the roll-pitch maneuver. The projection of the path onto the ground plane is color-coded to indicate the time-varying rotation mode. The bottom row shows the locations of the target maneuvers relative to the receiver and the FM radio transmitters.	89
7.2:	The F-4E Phantom fighter and its dimensions. The kinematic coefficients used in our single-target experiments were chosen to simulate the flight characteristics of the F-4E.	90
7.3:	Bistatic RCS in dBsm for 0° incident elevation and 0° observed elevation. The upper left panel is the RCS generated by FISC for a VFY-218 fighter. Each of the other panels was created using the scattering center model from (7.4) and (7.5) and a different set of randomly generated parameters.	94
7.4:	Example of tracking performance of the IMM-EKF during the roll-pitch maneuver. Red lines are true values; blue lines are IMM-EKF estimates. Vertical black dotted lines indicate changes in the target's flight mode.	97
7.5:	Example of the tracking performance of the generic particle filter with $N_p = 600$. Red lines are true values; blue lines are particle filter estimates. Vertical black dotted lines indicate changes in the target's flight mode.	100
7.6:	Example of mode probabilities for the IMM-EKF and the generic particle filter ($N_p = 600$). Left column: roll-roll maneuver. Right column: roll-pitch maneuver. Vertical black dotted lines indicate changes in the target's flight mode.	101
7.7:	Example of the mode probabilities for the generic particle filter with $N_p = 600$ and $\sigma_n^2 = 2$. Left plot: roll-roll maneuver. Right plot: roll-pitch maneuver.	104
7.8:	Example of track loss for IMM-EKF due to clutter and missed detections. For the trial shown, $\beta_2 = 100$ and $P_D = 0.7$	109
7.9:	Left panel: ground truth trajectories for each target. Arrows are spaced 4 s apart and indicate the direction of travel. Right panel: APF track estimates. Black squares indicate scans in which the MAP class for the given track was incorrect.	115

LIST OF TABLES

6.1: Euler Angles as a Function of Rotation Mode and Pilot Input	74
7.1: Ranges of Values for Pilot Command Variables	90
7.2: EKF Track Accuracy versus IMM Process Noise Variances	96
7.3: PF Tracking Accuracy versus Particle Count	99
7.4: Transition Times and Mode Sequences for Both Trajectories	101
7.5: Average Run-Time for the Roll-Roll Maneuver versus Particle Count .	102
7.6: Generic PF Tracking Accuracy versus σ_n^2	104
7.7: Generic PF Tracking Accuracy versus Scattering Center Model	105
7.8: IMM-EKF Tracking Accuracy versus β_2 and P_D	108
7.9: PF Tracking Accuracy versus β_3 and P_D	109
7.10: Confusion Matrix for Generic PF ($N_p = 600$)	113
7.11: Confusion Matrix for Auxiliary PF ($N_p = 400$)	113
7.12: Confusion Matrix for Generic PF ($N_p = 800$)	113
7.13: Confusion Matrix for Auxiliary PF ($N_p = 600$)	114
7.14: IMM-EKF Multitarget Tracking Accuracy	115
7.15: PF Multitarget Tracking Accuracy versus σ_n^2	116
7.16: Multitarget Confusion Matrix for Generic PF ($N_p = 800$)	119
7.17: Multitarget Confusion Matrix for Auxiliary PF ($N_p = 600$)	119

LIST OF SYMBOLS

$\mathbf{E}^i, \mathbf{E}^s$	incident and scattered electric fields
$\mathbf{k}_i, \mathbf{k}_s$	incident and scattered wavevectors
\mathbf{J}_s	surface current
$s_{vv}, s_{vh}, s_{hv}, s_{hh}$	complex scattering coefficients
λ	wavelength
EL	elevation angle
AZ	azimuth angle
<hr/>	
Δt	sample period
\mathbf{x}_k	state vector at scan k
$\mathbf{x}_{1:k}$	sequence of state vectors $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_k\}$
\mathbf{p}_k	position vector at scan k
\mathbf{v}_k	velocity vector at scan k
\mathbf{a}_k	acceleration vector at scan k
\mathbf{z}_k	measurement vector at scan k
$\mathbf{z}_{1:k}$	sequence of measurement vectors $\{\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_k\}$
τ_k	delay measurement from scan k
$\sigma_{\tau_k}^2$	variance of delay measurement from scan k
d_k	Doppler shift measurement from scan k
$\sigma_{d_k}^2$	variance of Doppler shift measurement from scan k
σ_k	radar cross section measurement from scan k
$\sigma_{n_k}^2$	variance of I/Q scattering coefficient measurements from scan k
<hr/>	
F_k	state matrix at scan k
f_k	state function at scan k
H_k	measurement matrix at scan k
h_k	measurement function at scan k
\mathbf{u}_k	process noise vector at scan k
Q_k	process noise covariance at scan k
$q^{(CV)}, q^{(CA)}, q^{(CT)}$	IMM process noise strengths
\mathbf{w}_k	measurement noise at scan k
R_k	measurement noise covariance at scan k

$\hat{\mathbf{x}}_k$	estimate of state vector at scan k
$\hat{\mathbf{x}}_{k k}$	conditional mean of \mathbf{x}_k given $\mathbf{z}_{1:k}$
$\hat{\Sigma}_{k k}$	conditional covariance of \mathbf{x}_k given $\mathbf{z}_{1:k}$
K_k	Kalman gain matrix at scan k
\mathbf{y}_k	innovation vector at scan k
S_k	innovation covariance at scan k
γ	IMM model index
$\hat{\mathbf{x}}_{k k}^{(\gamma)}$	state vector of IMM model γ at scan k
$\hat{\Sigma}_{k k}^{(\gamma)}$	state covariance of IMM model γ at scan k
$\Lambda_k^{(\gamma)}$	likelihood of IMM model γ at scan k
$\mu_k^{(\gamma)}$	<i>a posteriori</i> probability of IMM model γ at scan k

β	clutter density
V_k	volume of measurement gate at scan k
P_D	probability of detection
P_G	probability the true measurement is contained within the gate
m_k	number of measurements contained in \mathbf{z}_k
$\Gamma_k^{(l)}$	l^{th} data association hypothesis at scan k
N_t	number of targets

p	true distribution
π	proposal distribution
$\mathbf{x}_k^{(i)}$	i^{th} particle
N_p	number of samples used by particle filter
N_{eff}	effective particle count
\hat{N}_{eff}	estimate of effective particle count
N_{thresh}	resample threshold
$w_k^{*(i)}$	optimal importance weight for $\mathbf{x}_k^{(i)}$
$w_k^{(i)}$	unnormalized importance weight for $\mathbf{x}_k^{(i)}$
$\bar{w}_k^{(i)}$	normalized importance weight for $\mathbf{x}_k^{(i)}$

ψ_k	target heading angle at scan k
θ_k	target elevation angle at scan k
ϕ_k	target bank angle at scan k
\mathcal{O}^{bi}	orthogonal transformation matrix (inertial frame to body frame)
\mathcal{O}^{bw}	orthogonal transformation matrix (wind frame to body frame)
$\hat{b}_x, \hat{b}_y, \hat{b}_z$	unit vectors for body frame coordinate axes
α_k	angle of attack at scan k
β_k	sideslip angle at scan k

ω_k	turn-rate at scan k
T_k	magnitude of thrust force at scan k
m	mass of target
S	effective area of wings
\mathbf{f}_k	force applied to target at scan k
L	magnitude of lift force
D	magnitude of drag force
ζ	class index
\mathcal{A}_c	set of aerodynamic coefficients $\{C_{L_\alpha}, C_{D_{\min}}, K_D, S, m\}$
ρ_k	density of air at scan k

γ_k	maneuver mode at scan k for PF flight model
P_{IMM}	Markov transition matrix
ω_k^*	turn-rate pilot command at scan k
$\Delta\theta_k^*$	pitch up/down pilot command at scan k
T_k^*	thrust pilot command at scan k
$\mathbb{U}_{\Delta\theta}^{PT}, \mathbb{U}_T^{PT}, \mathbb{U}_\omega^{CT}, \mathbb{U}_{\Delta\theta}^{CT}$	uniform distributions for pilot commands

N_{MC}	number of Monte Carlo trials performed
e_{RMS}	RMS error
N_L	number of trials considered lost
\bar{N}_g	average number of measurements per gate

LIST OF ABBREVIATIONS

APF	Auxiliary Particle Filter
CA	Constant Acceleration
CT	Coordinated Turn
CV	Constant Velocity
EKF	Extended Kalman Filter
FISC	Fast Illinois Solver Code
IMM	Interacting Multiple Model
JPDA	Joint Probabilistic Data Association
MC	Monte Carlo
MHT	Multiple Hypothesis Tracking
MLFMA	Multilevel Fast Multipole Algorithm
MoM	Method of Moments
PDA	Probabilistic Data Association
PDAF	Probabilistic Data Association Filter
PF	Particle Filter
PT	Pitch/Thrust
RCS	Radar Cross Section
SIS	Sequential Importance Sampling

CHAPTER 1

INTRODUCTION

In the most general sense, the two questions that a surveillance system tries to answer are “Where is it?” and “What is it?” The question of “Where?” is a matter of target tracking. The question of “What?” is a matter of classification. In recent years, it has been suggested in the literature that tracking and classification should proceed simultaneously, or *jointly*, whenever possible [1–6]. While it might seem that a joint approach to tracking and classification could do no worse than separate implementations of each, this is not necessarily the case. Furthermore, significant improvements in performance require the two-way exchange of useful information between the tracker and classifier. Broadly speaking, the kinematic output of the tracker (e.g., position and velocity) should improve the performance of the classifier, just as the target identity supplied by the classifier should improve track accuracy. As a means of introduction, we will consider two existing approaches to joint tracking/classification. We will show why both techniques fall short of delivering the full benefit of a joint approach. Then, we will demonstrate how the joint formulation presented in this thesis is able to deliver upon its potential.

Before discussing the two existing approaches to joint tracking/classification, it is helpful to introduce the actual application that we have in mind. In this work, we are primarily interested in designing a ground-based passive radar system for tracking and classifying aircraft. While the assumption that the radar is passive does limit the type and quality of data available to us, it does not impact our present discussion. For now, it is enough to focus on the broader task of air surveillance. It is important, though, to specify what we mean by classification. For our classifier, we are specifically interested in identifying the model of an aircraft (e.g., F-15 or T-38), as opposed to other systems whose labels correspond to broad classifications such as “fighter/airliner” or “friend/neutral/foe.”

As it happens, the distinction between specific class labels (e.g., F-15) and broad class labels (e.g., fighter/airliner) is the primary difference between current approaches to joint tracking/classification. In systems that label targets broadly, classification is often based upon class-dependent kinematic models [3, 4, 6]. For example, in [3], the target was assumed to belong to one of two possible classes. The first class was defined for highly agile airplanes such as fighters. The second class was defined for

lower speed, less maneuverable targets such as commercial airliners. The difficulty with designations such as these is that they are not mutually exclusive. Thus, there is nothing to stop a highly maneuverable target such as a fighter from flying slow and straight, in which case it will be classified by the system as an airliner.

The second approach to joint tracking/classification includes at least one feature within its set of measurements with the potential to discriminate between specific target types. For example, in [1] the authors suggested using high-resolution range profiles; in [2, 5], radar cross section (RCS) was chosen. By not relying on kinematic information alone, the classifier has the potential to be much more accurate. In all three papers, the claim was made that knowledge of the specific target type could be used to improve tracking performance. For example, in [2, 5], a rigid-body motion model was used to represent target dynamics. The forces and torques that cause the translational and rotational motion were modeled as independent zero-mean Gaussian random processes. It was then suggested that the covariances of these processes could be varied based upon the output of the classifier. Thus, the covariance matrix for an aircraft that was highly maneuverable would be “larger” than an aircraft with low maneuverability. However, this can cause the same sort of problem as broad class labels. Namely, what happens if a highly maneuverable target does not maneuver? In this case, the covariance matrices of the random forces and torques are poorly matched to the target’s motion, and the influence of the classifier could actually *degrade* tracking accuracy.

At this point, a natural question might be whether there is any aircraft-specific information that could improve tracking performance. The answer is yes, but it actually requires a new perspective on tracking. In [7, 8] the authors suggested a motion model that used actual aerodynamic equations for flight. In both papers, it was noted that, in flight, the predominant acceleration is caused by the lift force on the airframe. Furthermore, it was pointed out that the magnitude of the lift force was proportionate to the target’s speed squared while the direction of the force was nearly normal to the target’s velocity vector and wings. This is a substantial departure from the common assumption that target accelerations along each coordinate dimension are uncoupled and can be modeled as either white Gaussian noise or Brownian motion. There are two difficulties with the aerodynamically motivated models in [7, 8] though. First, to determine the magnitude of the lift force requires knowledge of a constant of proportionality that is aircraft-specific. Second, to determine the direction of the lift force requires knowledge of the plane’s angular orientation. Because neither paper addressed the problem of classification, the constant of proportionality between lift and speed squared had to be estimated adaptively. Furthermore, the authors just assumed that an estimate of the target’s orientation could be acquired somehow.

With this as motivation, we can now state the primary contribution of this work. In this thesis, we present a recursive Bayesian solution to the problem of joint tracking/classification for ground-based air surveillance. In our design, we utilize the full benefit of a joint approach by implementing our tracker using an aerodynamically valid flight model similar to [8]. However, in our case, aircraft-specific coefficients such as the wing area, minimum drag, and vehicle mass are available to the tracker because

of our joint formulation. In addition, we specifically allow for such complications as multiple targets, false alarms, and missed detections.

The key feature that makes our system possible is inclusion of radar cross section as one of our measurements. However, instead of modeling RCS using a Swerling model (i.e., as a random variable drawn from an exponential distribution [4]), we directly account for the complex relationship that exists between RCS and variables such as target geometry, position, and angular orientation. By treating RCS as a deterministic (albeit complex) function plus noise, we gain two crucial benefits. First, we are able to use the time evolution of a target's radar cross section as the primary feature for class discrimination. Second, we are able to track a target's time-varying angular orientation, which is required by our flight model. However, accommodating the complex nature of RCS does not come without a penalty. Namely, the lack of a closed-form expression for RCS prevents us from using an extended Kalman filter or any of its variants for state estimation. Instead, we rely upon a sequential Monte Carlo-based approach known as particle filtering to perform the needed inference [9–15]. In addition to allowing us to include RCS as a component of our measurement vector, we will also show that particle filtering simplifies the implementation of our nonlinear non-Gaussian flight model.

In summary, although others have suggested that tracking and classification should be performed jointly, we believe that we are the first to offer a solution that realizes the full potential of such a coupling. Our joint formulation consists of three key developments: (1) an aerodynamically valid flight model that relies upon aircraft-specific coefficients such as the wing area and the minimum value of drag, (2) an electromagnetically correct model for RCS that yields information pertaining to both class identity and target orientation, and (3) a particle filter-based implementation that takes into account realistic difficulties caused by multiple targets, false alarms, and missed detections.

This thesis is organized as follows. Chapter 2 presents relevant aspects of scattering theory including the definition of radar cross section. It highlights the complex relationship between target aspect and RCS. In Chapter 3, we introduce the stochastic framework in which we will formulate our joint tracking/classification problem. We also demonstrate how the inclusion of RCS as a data feature prevents us from using any variant of Kalman filtering. In Chapter 4, we introduce sampling-based approaches to sequential estimation, with an emphasis on importance sampling and the particle filter. We demonstrate that the flexibility of the particle filtering algorithm allows us to cast our joint tracking/classification problem in the desired recursive Bayesian framework. Next, in Chapter 5, we present a tracker based on the extended Kalman filter that will serve as a benchmark for comparison when we present our experimental results. Chapter 6 contains the full details of our system, including our flight model and our measurement equation for RCS. Next, extensive experimental results are presented in Chapter 7. The goal of this chapter is twofold. First, we demonstrate the significant improvement in tracking accuracy that can be achieved through a joint formulation with an aerodynamically correct flight model. Second, we explore the capabilities of particle filter-based systems in the presence of data uncertainty due to false alarms and

multiple targets. Finally, we conclude in Chapter 8. Chapters 2–4 provide the necessary background for our formulation. Chapters 6 and 7 contain the main contributions of this work.

CHAPTER 2

BACKGROUND IN ELECTROMAGNETICS

The interaction between signal and target in radar systems is properly addressed using electromagnetic (EM) scattering theory. In this section, we present an overview of the EM concepts pertinent to our application. As a means of motivating this discussion, we begin by introducing the operational scenario that we will simulate in Chapter 7: ground-based FM-band passive air surveillance. A discussion of FM-band passive radar then leads naturally to the consideration of radar cross section as a potential feature for classification.

2.1 Ground-Based FM-Band Passive Radar

The majority of modern radar systems are *active*. An active radar both transmits and receives electromagnetic signals for the purpose of detection and estimation. In contrast, a *passive* radar system does not transmit electromagnetic energy of its own; instead, it relies on “illuminators of opportunity” such as commercial radio or television broadcasts. Regardless of the source or nature of transmission, both types of radar attempt to measure scattering of these signals in order to detect and locate targets in the nearby airspace. We will refer to a single pass made by a radar in search of new or existing targets as a *scan*. For convenience, we assume that scans are performed every Δt seconds. As such, the search performed by the radar at time $k\Delta t$ will be referred to as scan k . The result of each scan is a set of detections and their associated measurements (e.g., delay and Doppler shift). Detections across consecutive scans that are determined to have originated from the same target are associated and referred to as a *track*. Measured data corresponding to a given track can then be filtered to produce an estimate of the unknown flight path.

Although most radar systems are active, passive radar can provide some advantages, especially in military contexts [2, 16, 17]. Because an active radar advertises its location through its transmissions, it can quickly become a target itself. A passive radar, on the other hand, offers the primary benefits of survivability and robustness against deliberate directional interference. Second, because commercial transmitters focus their

energy toward the Earth's surface, passive radar permits illumination of low-flying aircraft. Finally, the signals used by passive radar typically lie in the VHF/UHF frequency ranges (55–885 MHz). At these frequencies, stealth measures such as target shaping and the use of radar absorbing material (RAM) are much less effective. For example, it has been found that the ability of surface faceting to deflect signal energy away from the radar is only effective if the dimensions of a single facet are large compared to the wavelength. At wavelengths on the order of a single facet or larger, the strength of the radar return becomes dependent on the target's volume rather than its fine structure. Similarly, radar absorbing material must be matched to the wavelength of the incident radiation as well. For example, dielectric absorbers typically need to be applied in layers that are $0.01\text{--}0.1 \lambda$ thick, where λ is the wavelength of the radar signal. At VHF/UHF frequencies, this is generally too thick to be used on air vehicles because it would change their aerodynamic properties [18].

The main disadvantage of passive radar is a matter of signal design. As mentioned, a passive radar gathers information about targets using commercial broadcasts, most of which are far from optimal for use in tracking. To understand why, consider that an active radar estimates range by measuring the time delay between pulse transmission and reception of the target echo. In contrast, commercial broadcasts are not pulsed waveforms; they are continuous-wave signals that may be considered to be “always on.” In the case of a passive radar using FM broadcasts, target range is estimated by correlating the modulation waveform of the direct-path FM signal with the modulation waveform of the scattered signal. As such, the accuracy of this range measurement, or *range resolution*, is related to the maximum excursion of the frequency modulation. More generally, it can be shown that the achievable range resolution is inversely proportionate to the bandwidth of the illuminating signal [19]. Because a chirp waveform used by an active radar can have a bandwidth that is 10 000 times larger than that of an FM radio broadcast, the challenge inherent in passive radar is clear.

Having established that the tracking performance of passive radar may be substantially worse than that of its active counterpart, a logical question might be, “Are there any advantages to using passive radar for nonmilitary applications?” As it turns out, the FM frequency band is well-suited to target classification. Before we can explain why this is so, we first must introduce the concept of radar cross section.

2.2 Radar Cross Section

In radar applications, we are often interested in how well a target captures and reradiates electromagnetic energy. Radar cross section (RCS) is a measure of the magnitude of this reflection process expressed as an effective area. RCS is a function of the size, shape, and composition of the target as well as the polarization and frequency of the incident wave. RCS also depends on the position and orientation of the target with respect to both the source of the incident wave (e.g., the FM transmitter) and the mea-

surement location (i.e., the radar receiver). RCS is formally defined as

$$\sigma(\mathbf{k}_i, \mathbf{k}_s) = \lim_{R \rightarrow \infty} 4\pi R^2 \frac{|\mathbf{E}^s(\mathbf{k}_s)|^2}{|\mathbf{E}^i(\mathbf{k}_i)|^2}, \quad (2.1)$$

where $|\mathbf{E}^i|$ is the amplitude of the incident wave measured at the target, and $|\mathbf{E}^s|$ is the amplitude of the scattered wave measured at a distance R from the target [20]. The variables \mathbf{k}_i and \mathbf{k}_s denote the incident and scattered wavevectors, respectively, expressed in a coordinate system that is centered on the target.¹ The limiting process ensures that the scattered signal is measured in the far-field. The incident wave in the RCS calculation is commonly taken to be a plane wave. This assumption is justified when the distance between the transmitter and the target is large because spherical phase fronts can then be approximated as locally planar.

Because RCS is a function of size, shape, and composition of the target, it has potential as a feature for classification. However, a radar cannot measure RCS directly. Instead, it measures the power P_r of the received waveform, which is related to RCS through the radar range equation

$$\sigma = \frac{(4\pi)^3 R_t^2 R_r^2}{G_t P_t G_r \lambda^2} P_r, \quad (2.2)$$

where P_t is the transmitted power, λ is the wavelength, R_t is the transmitter-to-target range, and R_r is the receiver-to-target range.² G_t and G_r are antenna gains for the transmitter and receiver, respectively. The fraction in (2.2) can be viewed as the normalization term needed to remove the effects of transmission and propagation from the received power measurement, leaving a quantity (RCS) that depends on the target alone. It is reasonable to assume that P_t , G_t , and G_r are either known quantities or may be determined through suitable calibration. The two distances, R_t and R_r , must be approximated using an estimate of the target's position (which is available, of course, in a joint tracking/classification framework).

Now, we return to our claim that FM radio frequencies are well-suited for classification. As we have just seen, the RCS of a target varies as its position and orientation with respect to the transmitter or receiver changes. To ensure robust classification in the presence of noise, which may induce errors in our estimates of position and orientation, it is helpful that RCS vary “slowly” with small changes in these components of the target's state. The variation in RCS, as reflected by the number of nulls encountered as a target's aspect changes, is proportionate to the maximum dimension of the target in wavelengths. At FM-band frequencies (100 MHz), a fighter-sized aircraft is approximately five wavelengths long. In contrast, at X-band frequencies used by active radars (10 GHz), the same aircraft would be 500 wavelengths long! Therefore, although VHF frequencies yield poor range resolution, they have the potential for robust classification through measurement of RCS [21, 22].

¹For notational simplicity, we do not indicate the dependence of RCS on frequency and polarization in (2.1).

²The wavelength used in (2.2) corresponds to the carrier frequency of the broadcast, because that is where most of the transmitted power is concentrated.

2.3 Method of Moments

We have just shown that radar cross section may be a viable feature for classification in a passive radar system. However, in order to use RCS (as measured by received power), we will need to know the expected RCS for each target type that we wish to classify, over the entire range of frequencies and aspect angles that may be encountered. According to (2.1), this amounts to finding the scattered far-field $\mathbf{E}^s(\mathbf{k}_s)$, given the incident frequency and $\mathbf{E}^i(\mathbf{k}_i)$. To accomplish this, the current \mathbf{J}_s induced by the incident wave on the surface of the target must be found. Unfortunately, analytic expressions for \mathbf{J}_s can only be found for the simplest target geometries. Instead, we must rely on either empirical or numeric techniques. In this section, we will discuss one of the most popular approaches to numerically estimating the RCS of a target. Our goal is to highlight the complicated relationship that exists between target state (e.g., position and orientation) and RCS.

The method of moments (MoM) is a numeric technique that has found widespread use in the solution of scattering problems involving complex targets [23]. The method of moments applies to general linear-operator equations, such as

$$\mathcal{L}r = e, \quad (2.3)$$

where \mathcal{L} is a linear operator, r is an unknown response, and e is a known excitation. The unknown response r is expanded as a sum of basis functions, $r = \sum_{j=1}^N a_j r_j$. To solve for the N unknowns $\{a_1, \dots, a_N\}$, we write N linearly independent equations. These are obtained by taking the inner product of (2.3) with a set of N testing functions $\{y_1, \dots, y_N\}$,

$$\langle y_i, \mathcal{L}r \rangle = \sum_{j=1}^N a_j \langle y_i, \mathcal{L}r_j \rangle = \langle y_i, e \rangle, \quad i = 1, \dots, N. \quad (2.4)$$

This equation can be written in matrix notation as $Z\mathbf{a} = \mathbf{e}$ where $Z_{ij} = \langle y_i, \mathcal{L}r_j \rangle$, \mathbf{a} is a column vector of the unknown coefficients, and $e_i = \langle y_i, e \rangle$. Matrix inversion of Z yields the desired basis coefficients: $\mathbf{a} = Z^{-1}\mathbf{e}$.

In our electromagnetic scattering application, \mathcal{L} is an integrodifferential operator derived from one of the boundary conditions of Maxwell's equations, e is an incident field term (either electric or magnetic), and r is an approximation of \mathbf{J}_s . The shape of the target is provided to the MoM code by a CAD file that partitions the surface of the target into smooth patches or planar facets. Both the basis functions $\{r_i\}$ and the testing functions $\{y_i\}$ are then defined in terms of the facetization of the target. A popular choice of basis is defined on pairs of adjacent triangular facets [24]. The so-called Rao-Wilton-Glisson (RWG) basis is designed to maintain continuity of the normal component of \mathbf{J}_s across facet boundaries, thereby avoiding the need for fictitious line and point charges. A popular choice of testing function is $y_i = r_i$, commonly referred to as Galerkin's method. If a target is modeled as a closed surface, use of the RWG basis requires one function per triangular facet edge, yielding a number of un-

knowns proportionate to the number of facets in the CAD representation. Because the level of facetization is dictated by the length of the target in wavelengths, an accurate representation of the surface current on a fighter-sized aircraft at gigahertz frequencies may require millions of facets.³

Unfortunately, with this many unknowns, it is impractical to invert the matrix Z . In cases such as these, a conjugate gradient technique can be used to replace the $O(N^3)$ matrix inversion with an $O(KN^2)$ iterative approach requiring only matrix-vector multiplies. (K is the number of iterations required for convergence.) If the amount of computation is still unwieldy, the Multilevel Fast-Multipole Algorithm (MLFMA) can be used to reduce the number of operations in the matrix-vector multiply, resulting in $O(KN \log N)$ computation [25]. An example of one such MLFMA-based software package is FISC (Fast Illinois Solver Code) [26]. FISC finds \mathbf{J}_s as described above and then solves for the scattered electric field using the far-field radiation integral

$$\mathbf{E}^s(\mathbf{k}_s) \approx \frac{-j c \mu}{2 R \lambda} e^{-j k R} \iint_S \left(\mathbf{J}_s - (\hat{i}_s \cdot \mathbf{J}_s) \hat{i}_s \right) e^{j \mathbf{k}_s \cdot \mathbf{r}} dS, \quad (2.5)$$

where μ is the permeability of the medium, c is the speed of light, \mathbf{r} is the location in a target-centered coordinate system of an infinitesimal patch on the surface of the scatterer, and \hat{i}_s is the unit vector in the direction of \mathbf{k}_s . The wavenumber k is just the magnitude of \mathbf{k}_s , $k = 2\pi/\lambda$.

It is beneficial to pause at this point in our discussion and draw attention to the fact that there is no closed-form relationship between a target's aspect and its resulting RCS. Thus, even though an extended Kalman filter can handle nonlinear relationships between state (target aspect) and measurement (RCS), that relationship must still be made explicit. In the case of RCS, the closest we could come would involve storing the entire admittance matrix (Z^{-1}) and manipulating a discretized version of (2.5), an approach that is clearly impractical for real-time operation.

Up until now, we have overlooked polarization in our discussion. However, it must be taken into account in order to understand how EM solvers like FISC calculate RCS. Recall that \mathbf{E}^i specifies the amplitude, phase, and orientation of the incident electric field at each point in space. Because RCS is defined as a ratio of magnitudes and \mathbf{E}^s is related to \mathbf{E}^i through integrodifferential equations, the amplitude and phase of the incident field are arbitrary. As such, FISC sets $|\mathbf{E}^i| = 1$. The only parameter of the incident field that matters then is its orientation (and frequency). If we assume, in the RCS computation, that \mathbf{E}^i is a uniform plane wave traveling through free space, the phase fronts are then planes perpendicular to the direction of propagation, $\mathbf{E}^i \cdot \mathbf{k}_i = 0$. As such, \mathbf{E}^i can be decomposed into two orthogonal components,

$$\mathbf{E}^i = E_\theta^i \hat{i}_\theta + E_\phi^i \hat{i}_\phi, \quad (2.6)$$

where (θ, ϕ) represent the incident direction in a spherical coordinate system centered on the target and $(\hat{i}_\theta, \hat{i}_\phi)$ are spherical unit vectors. Because the system relating \mathbf{E}^i

³As a rule of thumb, the maximum length of a facet edge should be between 0.1 and 0.2 wavelengths.

and \mathbf{J}_s is linear, FISC only needs to compute \mathbf{J}_s for $\mathbf{E}^i = \hat{i}_\theta$ and $\mathbf{E}^i = \hat{i}_\phi$; the current distribution due to all other incident orientations can be synthesized from these two results. \hat{i}_θ and \hat{i}_ϕ are typically referred to as “vertical” and “horizontal” polarizations, respectively.

In an analogous fashion, the scattered electric field can be expressed as a sum of orthogonal components,

$$\mathbf{E}^s = E_{\theta'}^s \hat{i}_{\theta'} + E_{\phi'}^s \hat{i}_{\phi'}, \quad (2.7)$$

where the spherical angles (θ', ϕ') in (2.7) are primed because they correspond to the scattered direction, which is not necessarily the same as the incident direction. If $\mathbf{k}_s = -\mathbf{k}_i$, the spherical angles will be the same and the scenario is termed *monostatic* scattering. The general case is referred to as *bistatic* scattering. We see that, given the incident and scattered directions, there are only four coefficients that are needed to calculate \mathbf{E}^s for any \mathbf{E}^i satisfying $\mathbf{E}^i \cdot \mathbf{k}_i = 0$. These four quantities constitute entries in a scattering matrix S ,

$$\begin{bmatrix} E_{\theta'}^s \\ E_{\phi'}^s \end{bmatrix} = \underbrace{\begin{bmatrix} s_{vv} & s_{vh} \\ s_{hv} & s_{hh} \end{bmatrix}}_S \begin{bmatrix} E_{\theta}^i \\ E_{\phi}^i \end{bmatrix}, \quad (2.8)$$

where we have adopted the popular notation of using ‘v’ and ‘h’ to denote vertical and horizontal polarizations. Because each element of S varies with \mathbf{k}_i and \mathbf{k}_s , the scattering matrix is actually a collection of four complex-valued functions. The square magnitude of each coefficient yields the corresponding RCS (e.g., $\sigma_{vh} = |s_{vh}|^2$). It is specifically these scattering coefficients that FISC calculates.

RCS plots for two different aircraft are shown in Figure 2.1. Note that FISC specifies the direction of wave propagation using elevation and azimuth angles (EL, AZ) in a target-centered coordinate system. These angles are related to the spherical coordinate system by $\text{EL} = 90^\circ - \theta$ and $\text{AZ} = -\phi$. An example of positive elevation and azimuth angles is provided in Figure 2.2. In order to plot bistatic RCS in two dimensions, we have set both incident and scattered elevation angles to zero. The plots in Figure 2.1 depict σ_{hh} , the RCS term corresponding to the s_{hh} scattering coefficient.

2.4 RCS Databases and Sampling Considerations

In the previous section, we discussed how a software package such as FISC can be used to numerically approximate the scattering matrix for a complex target such as an aircraft. For each target class of interest, we would need to use an EM solver code such as FISC to generate a database of RCS values (off-line, prior to tracking) corresponding to various incident and scattered directions. We could then estimate RCS along an arbitrary flight path by interpolating between database entries.

The incident wave’s direction can then be specified by the pair $(\text{EL}^i, \text{AZ}^i)$ while the scattered wave direction is indicated by $(\text{EL}^s, \text{AZ}^s)$. When computing RCS values for more than one pair of incident and scattered directions, FISC imposes the require-

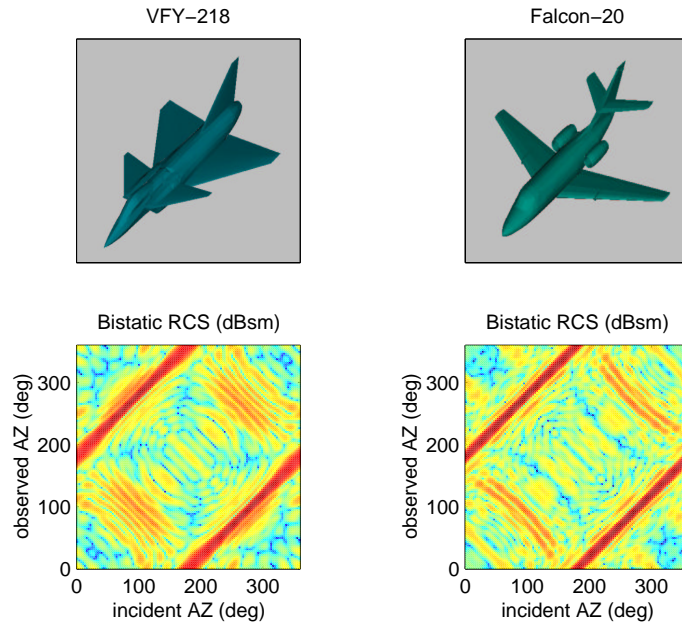


Figure 2.1: HH-polarized bistatic RCS for a VFY-218 fighter and a Falcon-20 commercial jet at 99.9 MHz and zero degrees incident and observed elevation. The color scale for both plots (from blue to red) is $[-26.55, 35.63]$ dBsm.

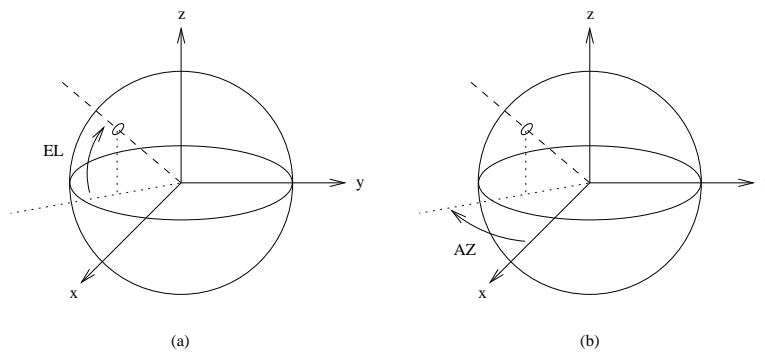


Figure 2.2: (a) Example of a positive elevation angle (EL) on the Cartesian axes. (b) Example of a positive azimuth angle (AZ) on the Cartesian axes.

ment that all angular values be separated by multiples of fixed elevation and azimuth increments. Define Δ_f as the fixed frequency increment, Δ_θ as the fixed elevation increment, and Δ_ϕ as the fixed azimuth increment. In this section, we will consider how to choose these three values, with the intent of minimizing the size of each database by using the largest increments possible.

To gain insight into this choice of increments, we will assume that our target can be modeled as a collection of ideal scattering centers, each with complex amplitude a_n and position $\mathbf{r}_n = [x_n, y_n, z_n]'$ in a target-centered coordinate system with unit vectors \hat{i}_x , \hat{i}_y , and \hat{i}_z defining the axes [27]. This model is appropriate when the wavelength of the incident field is small relative to the target dimensions. If we also assume that each scattering center is visible from all observation angles, the monostatic scattering from the target can be written as

$$G(f, \theta, \phi) = \sum_n a_n e^{-j2\mathbf{k}\cdot\mathbf{r}_n}. \quad (2.9)$$

If we substitute the expression for the wavevector

$$\mathbf{k} = \frac{2\pi}{\lambda} \left(\sin \theta \cos \phi \hat{i}_x + \sin \theta \sin \phi \hat{i}_y + \cos \theta \hat{i}_z \right) \quad (2.10)$$

into (2.9), we arrive at

$$G(f, \theta, \phi) = \sum_n a_n e^{-j2\pi(Xx_n + Yy_n + Zz_n)}, \quad (2.11)$$

where

$$X = \frac{2f}{c} \sin \theta \cos \phi, \quad Y = \frac{2f}{c} \sin \theta \sin \phi, \quad Z = \frac{2f}{c} \cos \theta, \quad (2.12)$$

and c is the speed of light. The capitalized notation highlights the fact that (x, y, z) and (X, Y, Z) form a Fourier transform pair. Because building an RCS database amounts to uniform sampling of $G(f, \theta, \phi)$, a reasonable choice for stepsizes would be ones that satisfy the Nyquist criterion for G .⁴

To enforce the Nyquist criterion, we need to make some assumptions about the spatial extent of the target as represented by the scattering centers $\{\mathbf{r}_n\}$. We assume that our target is oriented so that its nose points along \hat{i}_x , and the right wing (from the pilot's perspective) extends along \hat{i}_y . Furthermore, the center of the smallest rectangular volume enclosing the target is taken to be the origin of the (x, y, z) coordinate system. Finally, we also assume that all of the scattering centers $\{\mathbf{r}_n\}$ are contained within this rectangular bounding box. With these assumptions, the Nyquist criterion requires that the stepsizes for uniform sampling satisfy

$$\Delta_X \leq \frac{1}{L}, \quad \Delta_Y \leq \frac{1}{W}, \quad \text{and} \quad \Delta_Z \leq \frac{1}{H}, \quad (2.13)$$

⁴Building a bistatic RCS database actually amounts to uniformly sampling the function $G(f, \theta^i, \phi^i, \theta^s, \phi^s)$. However, it can be shown that the stepsizes that satisfy the Nyquist criterion in the monostatic scenario are lower bounds for any combination of incident and scattered directions.

where L , W , and H are the length, wingspan, and height, respectively, of the target. In order to translate the constraints on Δ_X , Δ_Y , and Δ_Z into values for Δ_f , Δ_θ , and Δ_ϕ , we take differentials of the relationships in (2.12),

$$\begin{bmatrix} \Delta_X \\ \Delta_Y \\ \Delta_Z \end{bmatrix} = \frac{2}{c} \begin{bmatrix} \sin \theta \cos \phi & f \cos \theta \cos \phi & -f \sin \theta \sin \phi \\ \sin \theta \sin \phi & f \cos \theta \sin \phi & f \sin \theta \cos \phi \\ \cos \theta & -f \sin \theta & 0 \end{bmatrix} \begin{bmatrix} \Delta_f \\ \Delta_\theta \\ \Delta_\phi \end{bmatrix}. \quad (2.14)$$

Our task then is to find values for Δ_f , Δ_θ , and Δ_ϕ so that (2.13) is satisfied for any choice of (f, θ, ϕ) . Clearly, the presence of f as a scaling factor for several terms in (2.14) makes it impossible to do this for all choices of $f \geq 0$. In order to proceed, we must incorporate f within the angular increments. Also, in the interest of simplifying the system of inequalities, we will set the two angular increments equal to each other. Defining $\tilde{\Delta} \triangleq f\Delta_\theta = f\Delta_\phi$, we can rewrite equations (2.13) and (2.14) as

$$\left| \Delta_f \sin \theta \cos \phi + \tilde{\Delta} \cos(\theta + \phi) \right| \leq \frac{c}{2L}, \quad (2.15)$$

$$\left| \Delta_f \sin \theta \sin \phi + \tilde{\Delta} \sin(\theta + \phi) \right| \leq \frac{c}{2W}, \quad (2.16)$$

$$\left| \Delta_f \cos \theta - \tilde{\Delta} \sin \theta \right| \leq \frac{c}{2H}. \quad (2.17)$$

This set of inequalities is a compact representation of the Nyquist criterion for uniform sampling in frequency-aspect space. To solve for the actual increments, we have to specify the desired trade-off between Δ_f and $\tilde{\Delta}$, because one can be made larger at the expense of the other. As a numeric example, consider the case where $\Delta_f = \tilde{\Delta}$ and $L = \max\{L, W, H\}$. In this case, (2.15) happens to impose the most stringent requirement on the sampling increment. Numeric evaluation yields

$$1.618 \Delta_f \leq \frac{c}{2L}. \quad (2.18)$$

If we take $c = 3 \times 10^8$ m/s, $f = 100$ MHz, and $L = 15$ m (for a fighter-sized aircraft), we have $\Delta_f \leq 6.18$ MHz. Using the largest increment possible and remembering that we chose $\tilde{\Delta} = \Delta_f$ for this example, the frequency-aspect increments for our RCS database would be $\Delta_f = 6.18$ MHz and $\Delta_\theta = \Delta_\phi = \tilde{\Delta}/f = 3.54^\circ$.

CHAPTER 3

RECURSIVE BAYESIAN FILTERING

In the introduction, we stated our desire to seek a recursive Bayesian solution to our joint tracking/classification problem. Before presenting this solution, we must first define what is meant by “recursive Bayesian filtering.” We begin our discussion in the most general terms, presenting concepts using random variables and density functions, in a development that parallels [28]. Once the recursive Bayesian filtering problem is understood in fundamental terms, we then expand our presentation by adopting a specific system model.

Many problems in statistical signal processing and automatic control can be cast in the framework of estimating the evolving “state” of a system given measurements that are stochastically related to it. We will provide a more precise definition of the system state later. For now, define \mathbf{x}_k as the state of the system at time $t = k\Delta t$, where Δt is the sampling interval. Thus, $\mathbf{x}_{1:k} \triangleq \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_k\}$ is a sequence of state values at $t = \Delta t, 2\Delta t, \dots, k\Delta t$. Because of either noise in the state evolution process or uncertainty as to the exact nature of the process itself, \mathbf{x}_k is generally regarded as a random variable. We assume that information concerning this unknown state sequence is conveyed through a measurement process. Define \mathbf{z}_k as the measurement produced by the system at $t = k\Delta t$ and, similar to our state notation, let $\mathbf{z}_{1:k} \triangleq \{\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_k\}$ denote a sequence of measurements. Here again, because of either noise in the measurement process or uncertainty in the underlying state sequence, \mathbf{z}_k is also regarded as a random variable.

With these definitions, we can now be more precise about our state estimation (or filtering) problem. Namely, for the purpose of monitoring or controlling a given stochastic system, we wish to estimate its evolving state \mathbf{x}_k using all measurements $\mathbf{z}_{1:k}$ collected up to the current time. Because \mathbf{x}_k is a random variable, all information provided by $\mathbf{z}_{1:k}$ is conveyed by the posterior density, $p(\mathbf{x}_k | \mathbf{z}_{1:k})$.¹ A *Bayesian* filtering algorithm is any technique that produces an estimate of $p(\mathbf{x}_k | \mathbf{z}_{1:k})$ for each $k = 1, 2, \dots$

¹In a slight abuse of notation, density functions will be identified by their arguments whenever this does not cause confusion (e.g., $p(\mathbf{x}_k | \mathbf{z}_{1:k})$ instead of the cumbersome $p_{\mathbf{x}_k | \mathbf{z}_{1:k}}(\mathbf{x}_k | \mathbf{z}_{1:k})$). Furthermore, we will use the same notation to refer to both random variables and their realizations, such as the use of $\mathbf{z}_{1:k}$ in $p(\mathbf{x}_k | \mathbf{z}_{1:k})$.

That is, instead of producing an estimate of \mathbf{x}_k directly, a Bayesian filtering algorithm supplies an estimate of the posterior $p(\mathbf{x}_k|\mathbf{z}_{1:k})$. We note that given the posterior density, it is straightforward (conceptually) to produce any desired statistic of \mathbf{x}_k . For instance, a minimum mean-square error (MMSE) estimate of the current state could be found by computing the conditional mean: $\hat{\mathbf{x}}_k = \int \mathbf{x}_k p(\mathbf{x}_k|\mathbf{z}_{1:k}) d\mathbf{x}_k$. A *recursive* Bayesian filtering algorithm imposes the constraint that the estimate of $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ should be generated solely from the previous posterior density $p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1})$ and the most recent measurement \mathbf{z}_k . In this way, the problem of storing $\mathbf{z}_{1:k}$, the entire measurement sequence, is avoided. Thus, we can summarize with the following definition.

Definition 1 *Recursive Bayesian Filtering*

Given $p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1})$ and the most recent measurement \mathbf{z}_k calculate (or estimate) the posterior density $p(\mathbf{x}_k|\mathbf{z}_{1:k})$.

In general, it is not possible to perform Bayesian filtering recursively (i.e., without storing the entire measurement sequence $\mathbf{z}_{1:k}$). To see why this is so, it is convenient to think of a two-step process that takes $p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1})$ and \mathbf{z}_k as inputs and returns the desired posterior $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ as the output. The first step of this process is commonly referred to as *prediction*, and it maps the previous posterior $p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1})$ into the one-step prediction density $p(\mathbf{x}_k|\mathbf{z}_{1:k-1})$. The second step, the *measurement update*, combines the most recent observation \mathbf{z}_k and $p(\mathbf{x}_k|\mathbf{z}_{1:k-1})$ from the prediction step to produce the desired posterior $p(\mathbf{x}_k|\mathbf{z}_{1:k})$. Formulas for these two steps follow immediately from Bayes' rule.

Definition 2 *Bayesian Prediction*

$$p(\mathbf{x}_k|\mathbf{z}_{1:k-1}) = \int_{\mathbf{x}_{k-1}} p(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{z}_{1:k-1}) p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1}) d\mathbf{x}_{k-1} \quad (3.1)$$

Definition 3 *Bayesian Measurement Update*

$$p(\mathbf{x}_k|\mathbf{z}_{1:k}) = \frac{p(\mathbf{z}_k|\mathbf{x}_k, \mathbf{z}_{1:k-1}) p(\mathbf{x}_k|\mathbf{z}_{1:k-1})}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})} \quad (3.2)$$

Upon examining these two definitions, we see immediately why Bayesian filtering cannot be performed recursively, in general. Specifically, the presence of the terms $p(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{z}_{1:k-1})$ in (3.1) and $p(\mathbf{z}_k|\mathbf{x}_k, \mathbf{z}_{1:k-1})$ in (3.2) require that we store all previous measurements $\mathbf{z}_{1:k-1}$. As an aside, we note that the denominator in (3.2) is not troublesome because it is a constant and can be found (theoretically) by integrating the numerator, $p(\mathbf{z}_k|\mathbf{z}_{1:k-1}) = \int p(\mathbf{z}_k|\mathbf{x}_k, \mathbf{z}_{1:k-1}) p(\mathbf{x}_k|\mathbf{z}_{1:k-1}) d\mathbf{x}_k$. Thus, in order to perform Bayesian filtering *recursively*, we must impose constraints on our stochastic system that allow the conditioning on $\mathbf{z}_{1:k-1}$ in $p(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{z}_{1:k-1})$ and $p(\mathbf{z}_k|\mathbf{x}_k, \mathbf{z}_{1:k-1})$ to be dropped.

The first assumption we make is that the state sequence is Markov,

$$p(\mathbf{x}_k|\mathbf{x}_{0:k-1}) = p(\mathbf{x}_k|\mathbf{x}_{k-1}). \quad (3.3)$$

Conceptually, for a Markov process, the value of the random variable \mathbf{x}_{k-1} provides as much information about \mathbf{x}_k as the value of the process at $t = (k-1)\Delta t$ and all previous time instants. In this sense, the typical notion of “state” is actually seen to follow from the definition of a Markov process; namely, the state of a system \mathbf{x}_k is any collection of variables that make the path history $\mathbf{x}_{0:k-1}$ inconsequential for estimating future values of the sequence. The second assumption we make is that observations occur through a memoryless channel. In other words, conditioned on \mathbf{x}_k , \mathbf{z}_k is assumed to be independent of the rest of the state sequence and all other measurements. This can be expressed as

$$p(\mathbf{z}_{1:k}|\mathbf{x}_{1:k}) = \prod_{i=1}^k p(\mathbf{z}_i|\mathbf{x}_i). \quad (3.4)$$

With these two assumptions, we are able to prove the following two lemmas, which will allow us to compute $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ from $p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1})$ without the past measurement sequence $\mathbf{z}_{1:k-1}$.

Lemma 1 Recursive Bayesian Prediction

Given a Markov state sequence $\{\mathbf{x}_k\}$ observed through a memoryless channel,

$$p(\mathbf{x}_k|\mathbf{z}_{1:k-1}) = \int_{\mathbf{x}_{k-1}} p(\mathbf{x}_k|\mathbf{x}_{k-1}) p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1}) d\mathbf{x}_{k-1}. \quad (3.5)$$

Proof:

By the Theorem of Total Probability and Bayes’ rule,

$$\begin{aligned} p(\mathbf{x}_k|\mathbf{z}_{1:k-1}) &= \int_{\mathbf{x}_{1:k-1}} p(\mathbf{x}_k, \mathbf{x}_{1:k-1}|\mathbf{z}_{1:k-1}) d\mathbf{x}_{1:k-1} \\ &= \int_{\mathbf{x}_{1:k-1}} \frac{p(\mathbf{z}_{1:k-1}|\mathbf{x}_{1:k}) p(\mathbf{x}_{1:k})}{p(\mathbf{z}_{1:k-1})} d\mathbf{x}_{1:k-1}. \end{aligned} \quad (3.6)$$

Next, we use the Markov assumption to express $p(\mathbf{x}_{1:k})$ as $p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{1:k-1})$. In addition, the conditional independence of the observation sequence allows us to drop the conditioning on \mathbf{x}_k in (3.6). This leaves us with

$$p(\mathbf{x}_k|\mathbf{z}_{1:k-1}) = \int_{\mathbf{x}_{1:k-1}} \frac{p(\mathbf{z}_{1:k-1}|\mathbf{x}_{1:k-1}) p(\mathbf{x}_{1:k-1}) p(\mathbf{x}_k|\mathbf{x}_{k-1})}{p(\mathbf{z}_{1:k-1})} d\mathbf{x}_{1:k-1} \quad (3.7)$$

$$= \int_{\mathbf{x}_{k-1}} \int_{\mathbf{x}_{1:k-2}} p(\mathbf{x}_{k-1}, \mathbf{x}_{1:k-2}|\mathbf{z}_{1:k-1}) p(\mathbf{x}_k|\mathbf{x}_{k-1}) d\mathbf{x}_{1:k-2} d\mathbf{x}_{k-1} \quad (3.8)$$

$$= \int_{\mathbf{x}_{k-1}} p(\mathbf{x}_k|\mathbf{x}_{k-1}) p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1}) d\mathbf{x}_{k-1}, \quad (3.9)$$

where we used Bayes’ rule to go from (3.7) to (3.8).

In Lemma 1, we showed that when the state sequence is Markov and the observations occur through a memoryless channel, the one-step prediction density $p(\mathbf{x}_k|\mathbf{z}_{1:k-1})$ can be computed from the previous posterior $p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1})$ without storing $\mathbf{z}_{1:k-1}$.

Now, we demonstrate how the one-step prediction density can be updated upon reception of a new observation \mathbf{z}_k .

Lemma 2 *Recursive Bayesian Update*

Given a measurement sequence $\{\mathbf{z}_k\}$ that is conditionally independent given the state sequence $\{\mathbf{x}_k\}$,

$$p(\mathbf{x}_k | \mathbf{z}_{1:k}) = \frac{p(\mathbf{z}_k | \mathbf{x}_k) p(\mathbf{x}_k | \mathbf{z}_{1:k-1})}{p(\mathbf{z}_k | \mathbf{z}_{1:k-1})}. \quad (3.10)$$

Proof:

We follow the proof given in [28]. Using Bayes' rule,

$$p(\mathbf{x}_k | \mathbf{z}_{1:k}) = \frac{p(\mathbf{z}_k, \mathbf{z}_{1:k-1} | \mathbf{x}_k) p(\mathbf{x}_k)}{p(\mathbf{z}_{1:k})} \quad (3.11)$$

$$= \frac{p(\mathbf{z}_k | \mathbf{x}_k) p(\mathbf{z}_{1:k-1} | \mathbf{x}_k) p(\mathbf{x}_k)}{p(\mathbf{z}_{1:k})}, \quad (3.12)$$

where we used the conditional independence of \mathbf{z}_k given \mathbf{x}_k to go from (3.11) to (3.12). Applying Bayes' rule to $p(\mathbf{z}_{1:k-1} | \mathbf{x}_k)$ yields

$$p(\mathbf{x}_k | \mathbf{z}_{1:k}) = \frac{p(\mathbf{z}_k | \mathbf{x}_k) p(\mathbf{x}_k | \mathbf{z}_{1:k-1}) p(\mathbf{z}_{1:k-1})}{p(\mathbf{z}_{1:k})} \quad (3.13)$$

$$= \frac{p(\mathbf{z}_k | \mathbf{x}_k) p(\mathbf{x}_k | \mathbf{z}_{1:k-1})}{p(\mathbf{z}_k | \mathbf{z}_{1:k-1})}, \quad (3.14)$$

where the marginal of \mathbf{x}_k , present in both the numerator and denominator, has been canceled in (3.13).

In Lemma 2, we note that the Markov assumption was not needed. Rather, the conditional independence of the measurement process was enough to derive $p(\mathbf{x}_k | \mathbf{z}_{1:k})$ from $p(\mathbf{x}_k | \mathbf{z}_{1:k-1})$, given only the current observation \mathbf{z}_k . In summary, while Bayesian filtering cannot be performed recursively in general, it can be accomplished for stochastic systems with Markov state sequences and conditionally independent measurement processes. We are now in a position to provide the motivation for our specific choice of system model.

Consider a stochastic system model, formulated in discrete time,

$$\mathbf{x}_k = f_k(\mathbf{x}_{k-1}, \mathbf{u}_k), \quad (3.15)$$

where \mathbf{x}_k is an $n_x \times 1$ state vector, and \mathbf{u}_k is an $n_u \times 1$ process noise vector. As such, each (potentially time-varying) function f_k maps $\mathbb{R}^{n_x} \times \mathbb{R}^{n_u}$ to \mathbb{R}^{n_x} . The sequence $\{\mathbf{u}_k\}$ can be used to model either a random perturbation of the state process or uncertainty in our knowledge of f_k (or both). This is a powerful formulation that allows us to approximate a differential equation of arbitrary order as a first-order vector difference equation. Next, consider a stochastic measurement equation of the form

$$\mathbf{z}_k = h_k(\mathbf{x}_k, \mathbf{w}_k), \quad (3.16)$$

where \mathbf{z}_k is an $n_z \times 1$ observation vector corresponding to state \mathbf{x}_k , and \mathbf{w}_k is an $n_w \times 1$ measurement noise vector. The observation function h_k , which is also allowed to vary with time, maps $\mathbb{R}^{n_x} \times \mathbb{R}^{n_w}$ to \mathbb{R}^{n_z} . Similar to the process noise, the sequence $\{\mathbf{w}_k\}$ can be used to model either a random perturbation of the observation process or uncertainty in our knowledge of h_k (or both). Furthermore, we note that while f_k and h_k are assumed to be known for all k , no assumptions are made as to their functional description (e.g., whether they are linear or nonlinear). The model described by (3.15) and (3.16) is often referred to as a *dynamic system*.

We wish to perform Bayesian filtering on the dynamic system described by (3.15) and (3.16). In order to avoid the growing memory problem inherent in computing $p(\mathbf{x}_k | \mathbf{z}_{1:k})$ for all k , we insist upon a recursive solution to the filtering problem. The requirements remain the same as before: namely, that $\{\mathbf{x}_k\}$ be Markov and $\mathbf{z}_{1:k}$ be conditionally independent given $\mathbf{x}_{1:k}$. However, by adopting a dynamic system model, these two assumptions can be restated solely in terms of the noise sequences, $\{\mathbf{u}_k\}$ and $\{\mathbf{w}_k\}$, and the random variable \mathbf{x}_0 corresponding to the initial state. We provide the result in the following proposition.

Proposition 1 *Recursive Bayesian Filtering for Dynamic Systems*

The dynamic system described by (3.15) and (3.16) admits a recursive solution to the Bayesian filtering problem, as described in Lemmas 1 and 2, if the following conditions hold.

1. *The noise vectors $\{\mathbf{u}_k\}$ form an independent sequence.*
2. *The noise vectors $\{\mathbf{w}_k\}$ form an independent sequence.*
3. *The sequences $\{\mathbf{u}_k\}$ and $\{\mathbf{w}_k\}$ and the random variable \mathbf{x}_0 are all mutually independent.*

The proof follows by applying properties (3.3) and (3.4) to the dynamic system model.

In this section, we have defined the recursive Bayesian filtering problem and provided its solution. Lemmas 1 and 2 show how $p(\mathbf{x}_k | \mathbf{z}_{1:k})$ can be calculated from $p(\mathbf{x}_{k-1} | \mathbf{z}_{1:k-1})$ and \mathbf{z}_k using the two-step procedure of prediction and measurement update. For the specific case of a dynamic system model, Proposition 1 provides a sufficient set of criteria for the existence of a recursive solution. We note that the three criteria are satisfied by many stochastic models of interest, including the one we will propose for joint tracking/classification. At first glance, it might seem that this settles the matter. Unfortunately, the solution provided by Lemmas 1 and 2 is conceptual; analytical solutions are only available for a handful of systems. More specifically, the integral in (3.5) and the integration needed to compute $p(\mathbf{z}_k | \mathbf{z}_{1:k-1})$ in (3.10) generally do not have closed-form solutions. Furthermore, even if the posterior $p(\mathbf{x}_k | \mathbf{z}_{1:k})$ can be found, estimates such as $\hat{\mathbf{x}}_k = \int \mathbf{x}_k p(\mathbf{x}_k | \mathbf{z}_{1:k}) d\mathbf{x}_k$ are likely to be intractable. In the next section, we discuss two cases where a closed-form solution to the prediction and measurement update equations can be found.

3.1 Exact Recursive Bayesian Algorithms

3.1.1 The HMM filter

In the previous section, we showed that if $\mathbf{x}_{1:k}$ was Markov and $\mathbf{z}_{1:k}$ conditionally independent given $\mathbf{x}_{1:k}$, a conceptual solution to the recursive Bayesian filtering problem could be found. Unfortunately, the two-step process of (3.5) and (3.10) does not, in general, admit an analytic solution. However, if \mathbf{x}_k is drawn from a state space that is both discrete and finite, the integrals in (3.5) and (3.10) become (finite) sums, which are easily computed. More specifically, the prediction equation becomes

$$\Pr(\mathbf{x}_k = i | \mathbf{z}_{1:k-1}) = \sum_{j=1}^N \Pr(\mathbf{x}_k = i | \mathbf{x}_{k-1} = j) \Pr(\mathbf{x}_{k-1} = j | \mathbf{z}_{1:k-1}) \quad (3.17)$$

for $i = 1, 2, \dots, N$, where N is the dimension of the state space. The term $\Pr(\mathbf{x}_k = i | \mathbf{x}_{k-1} = j)$ is an element from the Markov transition matrix. Likewise, the update equation simplifies to

$$\Pr(\mathbf{x}_k = i | \mathbf{z}_{1:k}) = \frac{p(\mathbf{z}_k | \mathbf{x}_k = i) \Pr(\mathbf{x}_k = i | \mathbf{z}_{1:k-1})}{\sum_{j=1}^N p(\mathbf{z}_k | \mathbf{x}_k = j) \Pr(\mathbf{x}_k = j | \mathbf{z}_{1:k-1})}, \quad (3.18)$$

where the normalization term in the denominator has been expanded to indicate how it would be computed. Thus, the recursion in Equations (3.17) and (3.18) requires only the ability to evaluate the transition probabilities $\Pr(\mathbf{x}_k = i | \mathbf{x}_{k-1} = j)$ and the likelihood function $p(\mathbf{z}_k | \mathbf{x}_k = i)$. Furthermore, the likelihood only needs to be evaluated up to a normalization constant because of its presence in both the numerator and denominator of (3.18).

State estimation algorithms based upon the recursion in (3.17)–(3.18) are often referred to as *hidden Markov model (HMM) filters*. These types of filters are widely used in speech recognition where it is assumed that all words are formed from the concatenation of basic language units called *phonemes* [29]. An example of a phoneme is the ‘i’ vowel sound in the word “hi.” Each language has its own distinct set, with approximately 40 for American English. The unknown (or hidden) state sequence $\mathbf{x}_{1:k}$ is the set of phonemes uttered by the speaker. The observation sequence $\mathbf{z}_{1:k}$ models both the human vocal mechanism and the measurement process (e.g., noise). Often, the likelihood $p(\mathbf{z}_k | \mathbf{x}_k = i)$ is taken to be a Gaussian mixture. As such, the HMM filter applies directly to the speech recognition problem.²

While it is true that HMM filters have revolutionized the field of speech recognition, they are only applicable to discrete state spaces or those that can be reasonably approximated as such. For our application, joint tracking/classification of airborne targets, this assumption is not valid. In the next section, we will present a second exact recursive Bayesian algorithm, but this one will apply to continuous state spaces.

²Actually, most speech recognizers find the maximum *a posteriori* estimate of $\mathbf{x}_{1:N_k}$ given $\mathbf{z}_{1:N_k}$, where N_k is the total number of samples. In this case, a common approach is to use the Viterbi algorithm, which replaces (3.17) by the computation $\max_j \{\Pr(\mathbf{x}_k = i, \mathbf{x}_{k-1} = j | \mathbf{z}_{1:k-1})\}$ for each i [30].

3.1.2 The Kalman filter

In this section, we maintain the assumptions necessary for recursive Bayesian filtering. Thus, $\mathbf{x}_{1:k}$ is Markov, and $\mathbf{z}_{1:k}$ is conditionally independent given $\mathbf{x}_{1:k}$ (or, for a dynamic system, $\{\mathbf{u}_k\}$, $\{\mathbf{w}_k\}$, and \mathbf{x}_0 are both individually and mutually independent for all k). However, instead of assuming that \mathbf{x}_k is drawn from a discrete state space, we instead assume that the system model takes the form

$$\mathbf{x}_k = F_k \mathbf{x}_{k-1} + \mathbf{u}_k, \quad (3.19)$$

$$\mathbf{z}_k = H_k \mathbf{x}_k + \mathbf{w}_k, \quad (3.20)$$

where F_k and H_k are appropriately dimensioned matrices. We recognize immediately that (3.19)–(3.20) represents a dynamic system model where the functions f_k and h_k , though still permitted to vary with time, are now linear. With this stochastic model, we make the following proposition.

Proposition 2 *Exact Recursive Bayesian Filtering for Linear, Gaussian Systems*

Given a dynamic system that satisfies (3.19)–(3.20), where $\{\mathbf{x}_k\}$ is Markov and $\{\mathbf{z}_k\}$ is conditionally independent given $\{\mathbf{x}_k\}$. If the initial state \mathbf{x}_0 and the noise processes $\{\mathbf{u}_k\}$ and $\{\mathbf{w}_k\}$ are each Gaussian for all k , then the Kalman filter provides the exact recursive Bayesian solution for $p(\mathbf{x}_k | \mathbf{z}_{1:k})$.

To see why this might be so, it is helpful to consider the actual Kalman filtering equations. We will denote a Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance matrix Σ as $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$. Similarly, evaluation of the same Gaussian density function at \mathbf{x} will be denoted as $\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \Sigma)$. Then, as specified in Proposition 2, we assume $\mathbf{u}_k \sim \mathcal{N}(\mathbf{0}, Q_k)$, $\mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, R_k)$, and $\mathbf{x}_0 \sim \mathcal{N}(\boldsymbol{\mu}_0, \Sigma_0)$, where the statistics Q_k , R_k , $\boldsymbol{\mu}_0$, and Σ_0 are assumed to be known for all k .³ Then, with the notation

$$\hat{\mathbf{x}}_{k|k} \triangleq \text{E}[\mathbf{x}_k | \mathbf{z}_{1:k}], \quad (3.21)$$

$$\hat{\Sigma}_{k|k} \triangleq \text{E}[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k})(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k})' | \mathbf{z}_{1:k}], \quad (3.22)$$

we can now state the Kalman filter equations [31, 32]:

$$\hat{\mathbf{x}}_{k|k-1} = F_k \hat{\mathbf{x}}_{k-1|k-1}, \quad (3.23)$$

$$\hat{\Sigma}_{k|k-1} = F_k \hat{\Sigma}_{k-1|k-1} F_k' + Q_k, \quad (3.24)$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + K_k (\mathbf{z}_k - H_k \hat{\mathbf{x}}_{k|k-1}), \quad (3.25)$$

$$\hat{\Sigma}_{k|k} = \hat{\Sigma}_{k|k-1} - K_k H_k \hat{\Sigma}_{k|k-1}, \quad (3.26)$$

³The noise processes $\{\mathbf{u}_k\}$ and $\{\mathbf{w}_k\}$ do not need to be zero-mean for the Kalman filter to be Bayesian optimal. However, because their mean values must be known for all k , it would be straightforward to recast the system in terms of zero-mean pseudo-noises and known control inputs.

where

$$K_k = \hat{\Sigma}_{k|k-1} H_k' S_k^{-1}, \quad (3.27)$$

$$S_k = H_k \hat{\Sigma}_{k|k-1} H_k' + R_k. \quad (3.28)$$

K_k is the Kalman gain, and S_k is the covariance of the innovation term, $\mathbf{z}_k - H_k \hat{\mathbf{x}}_{k|k-1}$.

At first glance, it might not seem that (3.23)–(3.26) have much to do with the recursive Bayesian filtering solution presented in Lemmas 1 and 2. However, the precise structure of the stochastic model (linear and Gaussian) permits a significant simplification of the prediction/update procedure. Because \mathbf{x}_0 and \mathbf{u}_1 are jointly Gaussian, \mathbf{x}_1 will be Gaussian. Then, because \mathbf{w}_1 is Gaussian, $\mathbf{z}_1 | \mathbf{x}_1$ will be Gaussian. Generalizing this line of reasoning, because Gaussian distributions are preserved under linear transformations, it can be shown that both the one-step prediction density $p(\mathbf{x}_k | \mathbf{z}_{1:k-1})$ and the posterior $p(\mathbf{x}_k | \mathbf{z}_{1:k})$ are Gaussian for all k . Because a Gaussian distribution is completely specified by its first two moments, it suffices to propagate the conditional mean $E[\mathbf{x}_k | \mathbf{z}_{1:k}]$ and the conditional covariance matrix $\text{var}[\mathbf{x}_k | \mathbf{z}_{1:k}]$. Inspection of (3.23)–(3.26) reveals that the Kalman filtering algorithm accomplishes precisely this. Equations (3.23)–(3.24) are the analytic solution for the Bayesian prediction step, while Equations (3.25)–(3.26) are the analytic solution for the Bayesian measurement update. Thus, we see that the well-known Kalman filter provides the exact recursive Bayesian solution for linear Gaussian systems.

3.2 Suboptimal Recursive Bayesian Algorithms

In the previous section, we discussed two filtering problems for which exact recursive Bayesian solutions exist. Unfortunately, for system models that are neither discrete nor linear-Gaussian, it is typically impossible to implement the recursive Bayesian solution provided by Lemmas 1 and 2. Instead, approximations to the optimal filter are often used. Because the state space for tracking problems is inherently continuous (or mixed, in the case of joint tracking/classification), most suboptimal algorithms in the tracking literature are based on the Kalman filter. Broadly speaking, these algorithms can be divided into two categories: (1) those that approximate the posterior as Gaussian, and (2) those that approximate the posterior as a sum of basis functions. In this section, we will present examples of both approaches to suboptimal recursive Bayesian filtering. In the end, though, we will find that none are suitable for FM-band passive radar tracking and classification.

3.2.1 Single Gaussian approximations

The majority of suboptimal Bayesian filters fall in the category that we term *single Gaussian approximations*. All of the filters in this class approximate the true posterior as a Gaussian distribution. We begin with the most popular of these, the extended Kalman filter.

Extended Kalman filter

The extended Kalman filter (EKF) applies the Kalman filtering recursion to the case of nonlinear state and measurement equations [33]. The dynamic system most often considered is

$$\mathbf{x}_k = f_k(\mathbf{x}_{k-1}) + \mathbf{u}_k, \quad (3.29)$$

$$\mathbf{z}_k = h_k(\mathbf{x}_k) + \mathbf{w}_k, \quad (3.30)$$

where the noise processes are assumed to be Gaussian, though f_k and h_k can now be nonlinear. The extended Kalman filter is derived by linearizing f_k and h_k using the first-order Taylor series approximations

$$f_k(\mathbf{x}_{k-1}) \approx f_k(\hat{\mathbf{x}}_{k-1|k-1}) + \tilde{F}_k(\mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k-1|k-1}), \quad (3.31)$$

$$h_k(\mathbf{x}_k) \approx h_k(\hat{\mathbf{x}}_{k|k-1}) + \tilde{H}_k(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1}), \quad (3.32)$$

where \tilde{F}_k and \tilde{H}_k are the Jacobian matrices,

$$\tilde{F}_k \triangleq \left. \frac{\partial f_k(\mathbf{x})}{\partial \mathbf{x}} \right|_{\mathbf{x}=\hat{\mathbf{x}}_{k-1|k-1}}, \quad (3.33)$$

$$\tilde{H}_k \triangleq \left. \frac{\partial h_k(\mathbf{x})}{\partial \mathbf{x}} \right|_{\mathbf{x}=\hat{\mathbf{x}}_{k|k-1}}. \quad (3.34)$$

Using the linear approximations in (3.31) and (3.32) in place of f_k and h_k in the standard Kalman filter derivation yields the following recursive algorithm:

$$\hat{\mathbf{x}}_{k|k-1} = f_k(\hat{\mathbf{x}}_{k-1|k-1}), \quad (3.35)$$

$$\hat{\Sigma}_{k|k-1} = \tilde{F}_k \hat{\Sigma}_{k-1|k-1} \tilde{F}_k' + Q_k, \quad (3.36)$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + K_k(\mathbf{z}_k - h_k(\hat{\mathbf{x}}_{k|k-1})), \quad (3.37)$$

$$\hat{\Sigma}_{k|k} = \hat{\Sigma}_{k|k-1} - K_k \tilde{H}_k \hat{\Sigma}_{k|k-1}. \quad (3.38)$$

The definitions for K_k and S_k are identical to (3.27) and (3.28) except H_k is replaced by the Jacobian \tilde{H}_k . The EKF algorithm approximates the posterior density $p(\mathbf{x}_k | \mathbf{z}_{1:k})$ as Gaussian with mean and covariance equal to $\hat{\mathbf{x}}_{k|k}$ from (3.37) and $\hat{\Sigma}_{k|k}$ from (3.38), respectively.

If the approximations provided by the first-order Taylor series expansion in (3.31) and (3.32) are not good enough, additional terms can be retained. This leads to higher order variants of the EKF algorithm. For example, the *truncated second-order filter* starts with a quadratic approximation for f_k and h_k and then retains all second-order moments during the ensuing derivation. The *Gaussian second-order filter* also begins with a quadratic approximation of the nonlinearities. However, whereas the truncated second-order filter ignores all central moments of \mathbf{x}_k above second order, the Gaussian second-order filter accounts for the fourth central moments by approximating them using the values they would assume if the posterior actually was Gaussian [33–35]. In

either case, the additional complexity of these higher-order filters has prevented their widespread use [36].

Interacting multiple model algorithm

In many tracking applications, targets maneuver in different fashions at different times. For example, an aircraft might fly with a constant velocity and heading initially before executing a tight turn. In cases such as these, it may be difficult to specify a single dynamic model that accurately represents the target's behavior at all times. One potential solution is to vary some of the parameters of the Kalman filter in an attempt to more accurately match the evolving target dynamics. For example, in [37], the authors adopted the system model $\mathbf{x}_k = F_k \mathbf{x}_{k-1} + G_k (\mathbf{u}_k + \boldsymbol{\lambda}_k)$, where $\{\boldsymbol{\lambda}_k\}$ was a sequence of unknown pilot command vectors drawn from a finite set, $\{\boldsymbol{\lambda}^{(1)}, \dots, \boldsymbol{\lambda}^{(N)}\}$. The authors then formulated $\{\mathbf{z}_k\}$ as observations from a hidden Markov model with $\{\boldsymbol{\lambda}_k\}$ as the hidden sequence. Under simplifying assumptions, their algorithm reduces to a single Kalman filter with $\hat{\boldsymbol{\lambda}}_k = \sum_{i=1}^N \boldsymbol{\lambda}^{(i)} \Pr(\boldsymbol{\lambda}_k = \boldsymbol{\lambda}^{(i)} | \mathbf{z}_{1:k})$ as supplementary input. In [38], the proposed filter switched back and forth between a constant-velocity model and a constant-acceleration model based upon a fading memory average of $(\mathbf{z}_k - H_k \hat{\mathbf{x}}_{k|k-1})' S_k^{-1} (\mathbf{z}_k - H_k \hat{\mathbf{x}}_{k|k-1})$. This algorithm attempts to estimate the unknown acceleration during maneuvers as opposed to confining it (through the use of fixed command inputs) to a predefined set. In either case, the restriction that a single state model (F_k or f_k) must be used for an entire sample period is a fundamental limitation of both approaches.

Alternatively, the motion of a target that switches between different maneuver modes can be modeled as a Jump Markov System (JMS),⁴

$$\mathbf{x}_k = f_k^{(\gamma_k)}(\mathbf{x}_{k-1}) + \mathbf{u}_k^{(\gamma_k)}, \quad (3.39)$$

$$\mathbf{z}_k = h_k^{(\gamma_k)}(\mathbf{x}_k) + \mathbf{w}_k, \quad (3.40)$$

where $\{\gamma_k\}$ is a finite-state Markov chain taking values in $\{1, 2, \dots, N\}$ according to the transition matrix P_{IMM} . The components of P_{IMM} are defined as $P_{IMM}(i, j) = \Pr(\gamma_k = j | \gamma_{k-1} = i)$ for any k . As seen from the superscripts in (3.39) and (3.40), a jump Markov system can have as many as N distinct state models, each with its own measurement function and process noise covariance. While this addresses the problem of modeling the motion of a target that maneuvers in different fashions at different times, optimal filtering algorithms for jump Markov systems have computational complexity that increases exponentially with k . For example, if the system was linear Gaussian when conditioned upon $\{\gamma_k\}$, a separate Kalman filter would be needed for every possible model sequence. Clearly, this can be impractical for even a modest number of samples. The *Interacting Multiple Model* (IMM) algorithm addresses this computational difficulty by merging the state estimates produced by each model at the beginning of each sample interval [39, 40]. Because this system will be used as a

⁴In general, the statistics of the measurement noise \mathbf{w}_k are also allowed to vary with γ_k , but this is unnecessary for our formulation.

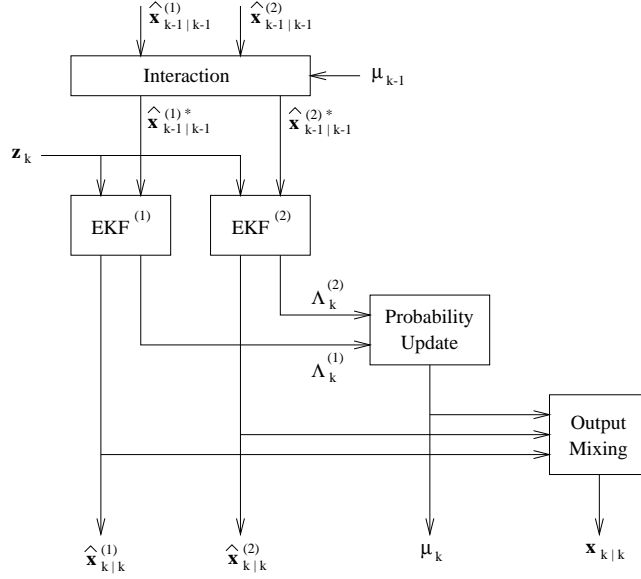


Figure 3.1: Example of IMM algorithm with $N = 2$.

performance benchmark in Chapter 7, we describe its implementation now.

The IMM algorithm uses N separate extended Kalman filters (or standard Kalman filters, if $f_k^{(\gamma)}$ and $h_k^{(\gamma)}$ are linear). We will use superscripts to denote the state estimates and covariances for each filter. Figure 3.1 provides a block diagram for the case where $N = 2$. Looking at the figure, we see that each iteration of the IMM algorithm proceeds in four steps. First, the previous outputs from each filter “interact” through the previous model probabilities $\mu_{k-1}^{(\gamma)}$ to produce mixed inputs $\hat{\mathbf{x}}_{k-1|k-1}^{(\gamma)*}$ for the current scan. This is the key step that keeps computation from growing exponentially. Next, each EKF operates as normal, producing the updated outputs $\hat{\mathbf{x}}_{k|k}^{(\gamma)}$. The likelihoods from each filter, denoted $\Lambda_k^{(\gamma)}$, are then used to generate the updated model probabilities $\mu_k^{(\gamma)}$. Finally, the updated probabilities are used to average the outputs of the individual filters. The details of each step are presented next.

1. Interaction (or Input Mixing)

Denote the *a posteriori* probability that the target was moving according to model γ at scan $k - 1$ as $\mu_{k-1}^{(\gamma)}$. Then, we have

$$\mu_{k-1}^{(i,\gamma)} \triangleq \Pr(\gamma_{k-1} = i | \gamma_k = \gamma) = \frac{P_{IMM}(i, \gamma) \mu_{k-1}^{(i)}}{C_{k-1}^{(\gamma)}}, \quad (3.41)$$

where

$$C_{k-1}^{(\gamma)} = \sum_{i=1}^N P_{IMM}(i, \gamma) \mu_{k-1}^{(i)} \quad (3.42)$$

and $\gamma = 1, \dots, N$. Note that $\mu_{k-1}^{(i,\gamma)}$ pertains to the “reverse” of a typical transition. With these probabilities, the previous state estimates and covariance matri-

ces are mixed according to

$$\hat{\mathbf{x}}_{k-1|k-1}^{(\gamma)*} = \sum_{i=1}^N \mu_{k-1}^{(i,\gamma)} \hat{\mathbf{x}}_{k-1|k-1}^{(i)}, \quad (3.43)$$

$$\begin{aligned} \hat{\Sigma}_{k-1|k-1}^{(\gamma)*} &= \sum_{i=1}^N \mu_{k-1}^{(i,\gamma)} \left(\hat{\Sigma}_{k-1|k-1}^{(i)} + \hat{\mathbf{x}}_{k-1|k-1}^{(i)} (\hat{\mathbf{x}}_{k-1|k-1}^{(i)})' \right) \\ &\quad - \hat{\mathbf{x}}_{k-1|k-1}^{(\gamma)*} (\hat{\mathbf{x}}_{k-1|k-1}^{(\gamma)*})'. \end{aligned} \quad (3.44)$$

Each mixed input is then provided to its respective extended Kalman filter.

2. Kalman Prediction and Update

The Kalman prediction and update proceed as specified in the regular EKF (see Equations (3.35)–(3.38)). The likelihood

$$\Lambda_k^{(\gamma)} = \mathcal{N} \left(\mathbf{z}_k; h_k^{(\gamma)}(\hat{\mathbf{x}}_{k|k-1}^{(\gamma)}), S_k^{(\gamma)} \right) \quad (3.45)$$

is also stored for use in the next step. Note that (3.45) assumes that the measurement noise is Gaussian.

3. Probability Update

Using the likelihoods from the previous step, the model probabilities are updated according to

$$\mu_k^{(\gamma)} = \frac{1}{C} \Lambda_k^{(\gamma)} C_{k-1}^{(\gamma)}, \quad \text{where} \quad C = \sum_{i=1}^N \Lambda_k^{(i)} C_{k-1}^{(i)}. \quad (3.46)$$

4. Output Mixing

The state estimate and covariance matrix for the IMM algorithm is obtained as a weighted sum of the outputs from the individual filters. With the updated model probabilities as the weights, we have

$$\hat{\mathbf{x}}_{k|k} = \sum_{\gamma=1}^N \mu_k^{(\gamma)} \hat{\mathbf{x}}_{k|k}^{(\gamma)}, \quad (3.47)$$

$$\hat{\Sigma}_{k|k} = \sum_{\gamma=1}^N \mu_k^{(\gamma)} \left(\hat{\Sigma}_{k|k}^{(\gamma)} + \hat{\mathbf{x}}_{k|k}^{(\gamma)} (\hat{\mathbf{x}}_{k|k}^{(\gamma)})' \right) - \hat{\mathbf{x}}_{k|k} (\hat{\mathbf{x}}_{k|k})'. \quad (3.48)$$

Note that the IMM algorithm requires N times the computation of an extended Kalman filter. Nonetheless, its excellent performance against maneuvering targets has made it very popular in the radar community. As such, it will serve as a benchmark when we present our experimental results.

Unscented Kalman filter

The *unscented transform* is motivated by the intuition that it may be easier to approximate a probability distribution than an arbitrary nonlinear function. Thus, instead of

linearizing f_k or h_k , the unscented transform deterministically generates a set of points whose sample mean and sample covariance are equal to $\hat{\mathbf{x}}_{k-1|k-1}$ and $\hat{\Sigma}_{k-1|k-1}$, respectively. The nonlinear function is then applied to each of the sample points, yielding a transformed sample from which the predicted mean and covariance are calculated [41, 42]. The *unscented Kalman filter* is so named because it implements the Kalman recursion using the sample points provided by the unscented transform.

The unscented Kalman filter has two advantages over the EKF. First, the estimate of the conditional mean provided by the unscented KF can be shown to be correct up to the second order (of its Taylor series expansion). Because the estimate of the conditional mean provided by the EKF is only correct to the first order, the unscented KF can sometimes be more accurate than the EKF [41]. Second, the unscented KF does not use any Jacobians (\tilde{F}_k or \tilde{H}_k). This is beneficial because these matrices may be cumbersome to derive.

Shortcomings of single Gaussian approximations

All of the single Gaussian approximations that we have presented suffer from the same shortcoming. By only maintaining estimates of the conditional mean and covariance, they implicitly approximate the posterior $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ as Gaussian. If the true density is either heavy-tailed or multimodal, a Gaussian description will be inaccurate. As an example, consider the effect of *target glint* on radar measurements. If the target is modeled as a collection of scattering centers, target glint is the error in state estimation due to interference between these phase centers as the target’s aspect with respect to the radar varies. In [12], it was noted that glint results in heavy-tailed, non-Gaussian measurement errors, which severely degrade the performance of the Kalman filter. A second example where a Gaussian approximation to the posterior is inappropriate is the case of tracking in the presence of clutter. In [13], the goal was to track outlines of foreground objects in video scenes that contained substantial background clutter. One specific experiment involved the task of tracking a person walking in front of other people. In the presence of such clutter, the true posterior density is multimodal. However, because of its Gaussian approximation, the Kalman filter is unable to accommodate simultaneous alternative hypotheses until they can be disambiguated by future measurements. Instead, the authors reported that the Kalman filter tended to get “distracted” by background clutter, never to recover.

Because our FM-band passive radar must track multiple targets in the presence of clutter (i.e., false alarms), the true posterior will generally be multimodal. As such, the single Gaussian approximations are poor matches for our application. There is an additional difficulty with the EKF and the IMM algorithm. Because delay and Doppler shift are nonlinear functions of position and velocity, the measurement model h_k for our application will be nonlinear as well. As discussed earlier, filtering within the Kalman framework then requires knowledge of the Jacobian matrix \tilde{H}_k . The Jacobian is required in order to map uncertainty in the measurement space into uncertainty in the state space during the Kalman update. Thus, if we wish to include RCS as a fea-

ture in our EKF data vector, we need to compute partial derivatives such as $\partial\sigma_k/\partial\mathbf{p}_k$, where \mathbf{p}_k is the position vector extracted from state \mathbf{x}_k . Because there is no closed-form expression for RCS, it will generally be impossible to find a closed-form expression for any partial derivative involving RCS (see Equations (2.1) and (2.5)). Instead, these derivatives would need to be approximated numerically, but the computational demands of such an approach would be impractical for real-time applications such as ours. Therefore, the need to compute Jacobians prevents the use of RCS within the Kalman framework. However, because the majority of class information is conveyed through RCS in VHF-band radar, this limitation immediately excludes both the EKF and the IMM algorithm as potential filtering solutions for our joint tracker/classifier.

3.2.2 Sum of basis functions

In the previous section, we considered several filtering algorithms that approximated the posterior density using a single Gaussian distribution. For applications such as multitarget tracking or tracking in the presence of clutter, the true posterior is often multimodal. In these cases, the Gaussian assumption reduces the amount of information available significantly. In fact, for multimodal systems, the EKF operates more as a maximum likelihood estimator than a minimum variance estimator, and the resulting estimate of the conditional mean may actually just follow one of the peaks of the true posterior. As an alternative to the single Gaussian assumption, a sum of basis functions can be used to approximate the true posterior. Although many different bases have been proposed in the literature, one particular example is noteworthy.

The *Gaussian sum filter* uses a weighted sum of Gaussian functions to approximate a posterior distribution of arbitrary complexity [43, 44],

$$\sum_{i=1}^N w_{k|k}^{(i)} \mathcal{N}(\mathbf{x}_k; \boldsymbol{\mu}_{k|k}^{(i)}, \boldsymbol{\Sigma}_{k|k}^{(i)}) \approx p(\mathbf{x}_k | \mathbf{z}_{1:k}), \quad (3.49)$$

where $\boldsymbol{\mu}_{k|k}^{(i)}$ and $\boldsymbol{\Sigma}_{k|k}^{(i)}$ are the mean and covariance, respectively, of the i^{th} Gaussian function given $\mathbf{z}_{1:k}$, and $\{w_{k|k}^{(i)}\}_{i=1}^N$ is a set of nonnegative weights that sum to one. The use of Gaussian functions has two distinct advantages over other choices for the basis. First, the Gaussian functions lend a certain mathematical tractability to the associated filtering algorithms. Second, the Gaussian sum is a valid density function for any N . Starting with a Gaussian sum approximation for $p(\mathbf{x}_k | \mathbf{z}_{1:k})$ and a state model of the form $\mathbf{x}_k = f_k(\mathbf{x}_{k-1}) + \mathbf{u}_k$, the one-step prediction density is approximated as

$$\begin{aligned}
p(\mathbf{x}_{k+1}|\mathbf{z}_{1:k}) &= \int p(\mathbf{x}_{k+1}|\mathbf{x}_k) p(\mathbf{x}_k|\mathbf{z}_{1:k}) d\mathbf{x}_k \\
&\approx \int p_{\mathbf{u}_{k+1}}(\mathbf{x}_{k+1} - f_{k+1}(\mathbf{x}_k)) \cdot \sum_{i=1}^N w_{k|k}^{(i)} \mathcal{N}(\mathbf{x}_k; \boldsymbol{\mu}_{k|k}^{(i)}, \Sigma_{k|k}^{(i)}) d\mathbf{x}_k \\
&= \sum_{i=1}^N w_{k|k}^{(i)} \int \mathcal{N}(\mathbf{x}_{k+1}; f_{k+1}(\mathbf{x}_k), Q_{k+1}) \mathcal{N}(\mathbf{x}_k; \boldsymbol{\mu}_{k|k}^{(i)}, \Sigma_{k|k}^{(i)}) d\mathbf{x}_k
\end{aligned} \tag{3.50}$$

$$\approx \sum_{i=1}^N w_{k+1|k}^{(i)} \mathcal{N}(\mathbf{x}_{k+1}; \boldsymbol{\mu}_{k+1|k}^{(i)}, \Sigma_{k+1|k}^{(i)}). \tag{3.51}$$

In general, the N integrals in (3.50) are intractable. In order to arrive at (3.51), f_{k+1} is replaced by a first-order Taylor series expansion about $\boldsymbol{\mu}_{k|k}^{(i)}$. The integrand is then a quadratic function of \mathbf{x}_k , allowing the integration to be performed analytically.

The measurement update of the Gaussian sum filter proceeds in a similar way. Assuming the form $\mathbf{z}_k = h_k(\mathbf{x}_k) + \mathbf{w}_k$, we have

$$\begin{aligned}
p(\mathbf{x}_{k+1}|\mathbf{z}_{1:k+1}) &= \frac{p(\mathbf{z}_{k+1}|\mathbf{x}_{k+1})p(\mathbf{x}_{k+1}|\mathbf{z}_{1:k})}{C_{k+1}} \\
&\approx \frac{p_{\mathbf{w}_{k+1}}(\mathbf{z}_{k+1} - h_{k+1}(\mathbf{x}_{k+1}))}{C_{k+1}} \sum_{i=1}^N w_{k+1|k}^{(i)} \mathcal{N}(\mathbf{x}_{k+1}; \boldsymbol{\mu}_{k+1|k}^{(i)}, \Sigma_{k+1|k}^{(i)}) \\
&= \frac{1}{C_{k+1}} \sum_{i=1}^N w_{k+1|k}^{(i)} \mathcal{N}(\mathbf{z}_{k+1}; h_{k+1}(\mathbf{x}_{k+1}), R_{k+1}) \\
&\quad \cdot \mathcal{N}(\mathbf{x}_{k+1}; \boldsymbol{\mu}_{k+1|k}^{(i)}, \Sigma_{k+1|k}^{(i)})
\end{aligned} \tag{3.52}$$

$$\approx \sum_{i=1}^N w_{k+1|k+1}^{(i)} \mathcal{N}(\mathbf{x}_{k+1}; \boldsymbol{\mu}_{k+1|k+1}^{(i)}, \Sigma_{k+1|k+1}^{(i)}), \tag{3.53}$$

where C_{k+1} is a normalization constant. To arrive at (3.53), the measurement function h_{k+1} in (3.52) is replaced by a first-order Taylor series approximation about $\boldsymbol{\mu}_{k+1|k}^{(i)}$. Each product in the sum in (3.52) is then quadratic in \mathbf{x}_{k+1} , and Equation (3.53) follows.

Thus, we see that the Gaussian sum filter is indeed able to maintain a multimodal approximation to the true posterior. As such, it has the potential to outperform the EKF when used for tracking in the presence of clutter or multiple targets. However, our development has highlighted a crucial shortcoming that it *does* share with the EKF, namely the need to compute the Jacobian matrix for h_k . Recall, in going from (3.52) to (3.53), h_{k+1} was replaced with a first-order Taylor series approximation. As discussed in the previous section, in general, there are no closed-form relationships for partial derivatives involving RCS. Thus, incorporation of RCS into the measurement vector prevents use of the Gaussian sum filter, just as it ruled out the EKF and its variants.

At this point, it would seem that the demanding nature of our RCS-based joint tracking/classification application has exhausted our Bayesian filtering options. Fortunately, this is not the case. There are a whole class of recursive Bayesian algorithms

that we have not considered yet, but they require a decidedly “non-Kalman” approach. These filters are based upon Monte Carlo (or random sampling) techniques, and their properties are the subject of the next chapter.

CHAPTER 4

MONTE CARLO-BASED APPROACHES TO BAYESIAN FILTERING

In the last chapter, we presented the Bayesian approach to filtering. In order to avoid having to store the past measurement sequence $\mathbf{z}_{1:k-1}$, we insisted upon a recursive solution. We then showed that such a solution existed if $\mathbf{x}_{1:k}$ was Markov and $\mathbf{z}_{1:k}$ was conditionally independent given $\mathbf{x}_{1:k}$. Unfortunately, the two-step recursive Bayesian solution provided by Lemmas 1 and 2, while conceptually appealing, proved to be intractable in all but two cases: (1) discrete, finite state spaces or (2) linear, Gaussian system models. Because our state space is continuous and our measurement model is nonlinear, we then considered several suboptimal extensions of the Kalman filtering algorithm. We found that they all suffered from at least one of two severe drawbacks: they either assumed a Gaussian distribution for $p(\mathbf{x}_k|\mathbf{z}_{1:k})$, or they assumed knowledge of the Jacobian $\partial h_k/\partial \mathbf{x}_k$. The first assumption is inappropriate for tracking applications involving clutter or multiple targets because $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ can then be multimodal. The second assumption fails because we wish to include radar cross section in \mathbf{z}_k for the purpose of performing target classification. Because there is no closed-form relationship between the components of \mathbf{x}_k (such as position and orientation) and RCS, it is impractical to compute (or estimate) $\partial h_k/\partial \mathbf{x}_k$.

It might seem as though our joint tracking/classification problem does not admit a practical recursive Bayesian solution. As it turns out, what is really at fault is the notion of extending the Kalman filtering algorithm to an application for which it was never intended. Instead, we need a radically different approach for joint tracking/classification. In this chapter, we will explore Monte Carlo (or sampling)-based approaches to recursive Bayesian filtering. We will find that their flexibility allows them to succeed where the Kalman filter and its variants failed.

4.1 Sampling from an Arbitrary Distribution

In this section, we provide the necessary theoretical background for the particle filtering algorithm introduced later in this chapter. Our presentation draws on overviews given in [45, 46]. We begin our discussion of Monte Carlo (MC)-based Bayesian filtering techniques by considering a slightly different problem, namely the computation of the high-dimensional integral

$$I(g) = \int g(\mathbf{x}) p(\mathbf{x}|\mathbf{z}) d\mathbf{x}, \quad (4.1)$$

where $g(\cdot)$ is any $p(\mathbf{x}|\mathbf{z})$ -integrable function. Because of integrals similar to (4.1), the recursive Bayesian filtering solution provided by Lemmas 1 and 2 was deemed impractical. More specifically, because the integration in (4.1) occurs over an $n_{\mathbf{x}}$ -dimensional space and $p(\mathbf{x}|\mathbf{z})$ may well be multivariate or nonstandard, (4.1) is typically intractable. Instead of seeking an analytic solution, an approximation, which we denote $I_{N_p}(g)$, can be obtained by Monte Carlo integration,

$$I_{N_p}(g) \triangleq \frac{1}{N_p} \sum_{i=1}^{N_p} g(\mathbf{x}^{(i)}), \quad (4.2)$$

where $\{\mathbf{x}^{(i)} : i = 1, \dots, N_p\}$ are drawn independently from $p(\mathbf{x}|\mathbf{z})$. The validity of this approximation is guaranteed by the strong law of large numbers (SLLN), which states that the average of many independent random variables with common mean and finite variance converges to their common mean:

$$\lim_{N_p \rightarrow \infty} I_{N_p}(g) = I(g), \quad \text{with probability one.} \quad (4.3)$$

Furthermore, if the variance σ_g^2 of $g(\mathbf{x})$ with respect to $p(\mathbf{x}|\mathbf{z})$ is finite, a central limit theorem also holds:

$$\sqrt{N_p} (I_{N_p}(g) - I(g)) \longrightarrow \mathcal{N}(0, \sigma_g^2), \quad \text{in distribution.} \quad (4.4)$$

Equation (4.4) is quite useful because it indicates how the error in our Monte Carlo approximation varies with N_p . The advantage of Monte Carlo integration is clear. Whereas a Riemann approximation, which samples the state space in a deterministic manner, has an accuracy of $O(N_p^{-1/n_{\mathbf{x}}})$ for an $n_{\mathbf{x}}$ -dimensional integral, the Monte Carlo approximation, which samples the state space randomly, has an accuracy of $O(N_p^{-1/2})$, independent of $n_{\mathbf{x}}$ [46]. It is in this sense that Monte Carlo integration is said to “beat the curse of dimensionality” [47]. Unfortunately, N_p is only half the story in (4.4); σ_g^2 may grow appreciably as the dimension of the state space increases. Thus, although Monte Carlo integration is theoretically preferable in high-dimensional spaces to deterministic techniques, both types of approximation may be inaccurate, depending on how σ_g^2 scales with $n_{\mathbf{x}}$. Furthermore, as a random sampling technique, the approximation $I_{N_p}(g)$ requires the ability to draw independent samples $\{\mathbf{x}^{(i)}\}$ from

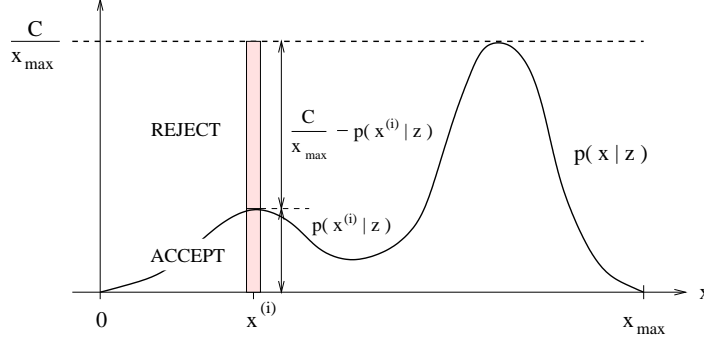


Figure 4.1: A simple example of rejection sampling. The sample $x^{(i)}$ is drawn from a uniform proposal distribution, $\pi(x|z) = \mathbb{U}_{[0, x_{\max}]}(x)$. The pink bars indicate the probabilities of acceptance and rejection.

$p(\mathbf{x}|\mathbf{z})$. As such, a nonstandard $p(\mathbf{x}|\mathbf{z})$ can present a substantial difficulty for MC integration. We consider this problem next.

One of the main tasks of any Monte Carlo method is the generation of independent samples from a target distribution such as $p(\mathbf{x}|\mathbf{z})$. In general, it is not possible to draw these samples directly. Instead, one of three indirect approaches is typically used. We briefly introduce each as a means of motivating the particle filtering algorithm. We will find that these indirect samples are either dependent or drawn from some distribution other than $p(\mathbf{x}|\mathbf{z})$ [48]. We will use $\pi(\mathbf{x}|\mathbf{z}) \neq p(\mathbf{x}|\mathbf{z})$ to denote a *proposal* distribution from which we *are* able to draw samples. Also, for all three sampling techniques, we assume that $p(\mathbf{x}|\mathbf{z})$ can be evaluated for any \mathbf{x} .

4.1.1 Rejection sampling

For a given $\pi(\mathbf{x}|\mathbf{z})$, suppose that we can find a constant C such that

$$C\pi(\mathbf{x}|\mathbf{z}) \geq p(\mathbf{x}|\mathbf{z})$$

for all \mathbf{x} . Then, the rejection sampling algorithm is as follows [49]:

1. Draw $\mathbf{x}^{(i)}$ from $\pi(\mathbf{x}|\mathbf{z})$ and compute the ratio

$$r = \frac{p(\mathbf{x}^{(i)}|\mathbf{z})}{C\pi(\mathbf{x}^{(i)}|\mathbf{z})} \leq 1.$$

2. Draw $u \sim \mathbb{U}_{[0,1]}$, where $\mathbb{U}_{[0,1]}$ is the uniform distribution on $[0, 1]$.
3. If $u \leq r$, accept $\mathbf{x}^{(i)}$; otherwise reject it.

It can be shown that the accepted samples follow the distribution $p(\mathbf{x}|\mathbf{z})$. A simple example involving a true distribution with support $\{x : p(x|z) > 0\} = [0, x_{\max}]$ is shown in Figure 4.1. The proposal distribution is taken to be uniform, $\pi(x|z) = \mathbb{U}_{[0, x_{\max}]}(x)$. In this case, a sample $x^{(i)} \sim \mathbb{U}_{[0, x_{\max}]}$ is proposed and accepted with probability $r = x_{\max} p(x^{(i)}|z)/C$.

In the general case, the probability of acceptance is

$$\Pr(\text{accept}|\mathbf{z}) = \int \Pr(\text{accept}|\mathbf{x}, \mathbf{z}) \pi(\mathbf{x}|\mathbf{z}) d\mathbf{x} = \frac{1}{C}. \quad (4.5)$$

Thus, as C grows, rejection sampling becomes increasingly inefficient at generating samples from $p(\mathbf{x}|\mathbf{z})$. For C to be chosen as conservatively as possible, the maximum value of $p(\mathbf{x}|\mathbf{z})/\pi(\mathbf{x}|\mathbf{z})$ must be found. Because $p(\mathbf{x}|\mathbf{z})$ is nonstandard, this may be a challenging nonlinear optimization problem in itself. On the other hand, if we heuristically increase C to ensure that it is larger than the (unknown) maximum, the efficiency of the algorithm will suffer. Overall, because of the relative inefficiency indicated by (4.5), rejection sampling is unsuitable for real-time applications such as radar that require samples from $p(\mathbf{x}|\mathbf{z})$ at each scan.

4.1.2 Metropolis-Hastings algorithm

The Metropolis-Hastings algorithm was suggested by Metropolis et al. [50] and later extended by Hastings [51]. The main idea behind the Metropolis-Hastings algorithm is to simulate a Markov chain in the state space of \mathbf{x} so that the stationary distribution of the chain is $p(\mathbf{x}|\mathbf{z})$. Note that in Markov chain analysis, one is typically given a transition function and asked to find the corresponding stationary distribution. With the Metropolis-Hastings algorithm, we consider the inverse problem: given a stationary distribution, find a transition function that reaches this equilibrium point efficiently. Starting with any initial sample $\mathbf{x}^{(0)}$, the Metropolis-Hastings algorithm proceeds as follows [46]:

1. Draw $\tilde{\mathbf{x}} \sim \pi(\cdot|\mathbf{x}^{(i)})$ and compute the ratio

$$r(\tilde{\mathbf{x}}, \mathbf{x}^{(i)}) = \min \left\{ 1, \frac{p(\tilde{\mathbf{x}}|\mathbf{z}) \pi(\mathbf{x}^{(i)}|\tilde{\mathbf{x}})}{p(\mathbf{x}^{(i)}|\mathbf{z}) \pi(\tilde{\mathbf{x}}|\mathbf{x}^{(i)})} \right\}.$$

2. Draw $u \sim \mathbb{U}_{[0,1]}$.
3. If $u \leq r(\tilde{\mathbf{x}}, \mathbf{x}^{(i)})$, $\mathbf{x}^{(i+1)} = \tilde{\mathbf{x}}$; otherwise $\mathbf{x}^{(i+1)} = \mathbf{x}^{(i)}$.

Note that for $\tilde{\mathbf{x}} \neq \mathbf{x}$, the actual transition function of the algorithm is $\pi(\tilde{\mathbf{x}}|\mathbf{x})r(\tilde{\mathbf{x}}, \mathbf{x})$ (i.e., the proposal probability times the acceptance probability). The only serious restriction on our choice of the proposal distribution is that $\pi(\tilde{\mathbf{x}}|\mathbf{x}) > 0$ if and only if $\pi(\mathbf{x}|\tilde{\mathbf{x}}) > 0$. It can be shown that the chain induced by the Metropolis-Hastings algorithm is *reversible*¹ and has $p(\mathbf{x}|\mathbf{z})$ as its invariant distribution [46].

We notice that, unlike rejection sampling, the Metropolis-Hastings algorithm does not require the maximization of $p(\mathbf{x}|\mathbf{z})/\pi(\mathbf{x}|\mathbf{z})$. In fact, because $p(\cdot|\mathbf{z})$ appears in both the numerator and denominator of $r(\tilde{\mathbf{x}}, \mathbf{x})$, we only need to evaluate it up to a normalizing constant. Nonetheless, the Metropolis-Hastings algorithm has two drawbacks

¹In the Markov chain literature, chains that satisfy the *detailed balance* condition are referred to as *reversible*. Detailed balance is a condition that ensures invariance with respect to the transition function $\pi(\tilde{\mathbf{x}}|\mathbf{x})r(\tilde{\mathbf{x}}, \mathbf{x})$.

that make it unsuitable for many real-time applications. First, it typically requires a substantial burn-in period (possibly on the order of thousands of samples) before the chain reaches its stationary distribution and samples can be collected. Even more troubling, though, is that the resulting samples are clearly dependent. In order to reduce the correlation between consecutive $\mathbf{x}^{(i)}$, researchers sometimes choose to keep every n^{th} sample (e.g., $\{\mathbf{x}^{(50)}, \mathbf{x}^{(100)}, \mathbf{x}^{(150)}, \dots\}$ for $n = 50$).

4.1.3 Importance sampling

Thus far, we have considered two approaches to generating samples from an arbitrary distribution. Rejection sampling is typically inefficient and presumes knowledge of $\max_{\mathbf{x}} \{p(\mathbf{x}|\mathbf{z})/\pi(\mathbf{x}|\mathbf{z})\}$ while the Metropolis-Hastings algorithm requires a substantial burn-in period and generates dependent samples. As such, both techniques are unsuitable for real-time applications. In this section, we introduce a third approach, importance sampling [52], and demonstrate that it is well-matched to our intended application. In the next section, we will introduce the sequential extension of the importance sampling algorithm, which is equivalent to particle filtering.

Recall that rejection sampling required knowledge of a constant C such that $C \geq p(\mathbf{x}|\mathbf{z})/\pi(\mathbf{x}|\mathbf{z})$ for all \mathbf{x} . Importance sampling has no such constraint; instead it merely requires that $\pi(\mathbf{x}|\mathbf{z}) > 0$ whenever $p(\mathbf{x}|\mathbf{z}) > 0$. This requirement is necessary; otherwise, there would be regions of the state space where $p(\mathbf{x}|\mathbf{z})$ was nonzero that could not be “reached” by samples drawn from $\pi(\mathbf{x}|\mathbf{z})$. With this assumption, the importance algorithm can be motivated by simply rewriting the integral from (4.1),

$$I(g) = \int g(\mathbf{x}) \frac{p(\mathbf{x}|\mathbf{z})}{\pi(\mathbf{x}|\mathbf{z})} \pi(\mathbf{x}|\mathbf{z}) d\mathbf{x} \quad (4.6)$$

$$= E_{\pi}[g(\mathbf{x})w^*(\mathbf{x})|\mathbf{z}], \quad (4.7)$$

where

$$w^*(\mathbf{x}) = p(\mathbf{x}|\mathbf{z})/\pi(\mathbf{x}|\mathbf{z}), \quad (4.8)$$

and $E_{\pi}[\cdot|\mathbf{z}]$ is the expectation taken with respect to $\pi(\mathbf{x}|\mathbf{z})$. By design, it should be easy to sample from the proposal distribution. Thus, after drawing N_p independent samples $\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(N_p)}\}$ according to $\pi(\mathbf{x}|\mathbf{z})$, the expectation in (4.7) can be approximated via MC integration as

$$\hat{I}_{N_p}^*(g) \triangleq \frac{1}{N_p} \sum_{i=1}^{N_p} g(\mathbf{x}^{(i)})w^{*(i)}, \quad (4.9)$$

where $w^{*(i)}$ is shorthand for $w^*(\mathbf{x}^{(i)})$. The set $\{w^{*(1)}, w^{*(2)}, \dots, w^{*(N_p)}\}$ is referred to as the *importance weights*. An explicit definition follows from (4.8):

$$w^{*(i)} \triangleq \frac{p(\mathbf{x}^{(i)}|\mathbf{z})}{\pi(\mathbf{x}^{(i)}|\mathbf{z})} = \frac{p(\mathbf{z}|\mathbf{x}^{(i)})p(\mathbf{x}^{(i)})}{p(\mathbf{z})\pi(\mathbf{x}^{(i)}|\mathbf{z})}, \quad (4.10)$$

where we applied Bayes' rule to rewrite $p(\mathbf{x}^{(i)}|\mathbf{z})$. Just as in (4.2), the estimate $\hat{I}_{N_p}^*(g)$ is unbiased. Furthermore, if the variance of $g(\mathbf{x})$ with respect to $\pi(\mathbf{x}|\mathbf{z})$ is finite, $\hat{I}_{N_p}^*(g)$ can be shown to converge with probability one to $I(g)$ as $N_p \rightarrow \infty$ by the strong law of large numbers. Unfortunately, there is a practical difficulty with (4.10) that must be addressed. So far in this chapter, we have assumed that $p(\mathbf{x}|\mathbf{z})$ could be evaluated, even if it was impossible to draw samples from it. For the types of stochastic models discussed in Chapter 3 (i.e., Markov state sequences observed through memoryless channels), this is generally not the case. While it is usually straightforward to evaluate the likelihood $p(\mathbf{z}|\mathbf{x})$ and the prior $p(\mathbf{x})$ for such models, the calculation of the normalization term $p(\mathbf{z})$ in the denominator of (4.10) is often intractable.

In order to avoid the need to evaluate $p(\mathbf{z})$, we use Bayes' rule to rewrite (4.6).

$$I(g) = \frac{1}{p(\mathbf{z})} \int g(\mathbf{x}) \frac{p(\mathbf{z}|\mathbf{x}) p(\mathbf{x})}{\pi(\mathbf{x}|\mathbf{z})} \pi(\mathbf{x}|\mathbf{z}) d\mathbf{x} \quad (4.11)$$

$$= \frac{\int g(\mathbf{x}) \frac{p(\mathbf{z}|\mathbf{x}) p(\mathbf{x})}{\pi(\mathbf{x}|\mathbf{z})} \pi(\mathbf{x}|\mathbf{z}) d\mathbf{x}}{\int \frac{p(\mathbf{z}|\mathbf{x}) p(\mathbf{x})}{\pi(\mathbf{x}|\mathbf{z})} \pi(\mathbf{x}|\mathbf{z}) d\mathbf{x}} \quad (4.12)$$

$$= \frac{E_\pi[g(\mathbf{x})w(\mathbf{x})|\mathbf{z}]}{E_\pi[w(\mathbf{x})|\mathbf{z}]}, \quad (4.13)$$

where

$$w(\mathbf{x}) = p(\mathbf{z}|\mathbf{x}) p(\mathbf{x}) / \pi(\mathbf{x}|\mathbf{z}) \quad (4.14)$$

can be computed. We see that the normalization term $p(\mathbf{z})$ from (4.11) has been expanded as the integral in the denominator of (4.12). In this case, a set of N_p independent samples drawn from $\pi(\mathbf{x}|\mathbf{z})$ can be used to estimate both expectations in (4.13), thereby yielding a new approximation for $I(g)$.

$$\hat{I}_{N_p}(g) \triangleq \frac{\frac{1}{N_p} \sum_{i=1}^{N_p} g(\mathbf{x}^{(i)}) w^{(i)}}{\frac{1}{N_p} \sum_{j=1}^{N_p} w^{(j)}} = \sum_{i=1}^{N_p} g(\mathbf{x}^{(i)}) \bar{w}^{(i)}, \quad (4.15)$$

where $w^{(i)}$ is shorthand for $w(\mathbf{x}^{(i)})$ and

$$\bar{w}^{(i)} \triangleq \frac{w^{(i)}}{\sum_{j=1}^{N_p} w^{(j)}} = \frac{w(\mathbf{x}^{(i)})}{\sum_{j=1}^{N_p} w(\mathbf{x}^{(j)})}. \quad (4.16)$$

From (4.16), we notice that the $\{\bar{w}^{(i)}\}$ are normalized to sum to one. To avoid confusion, from here forward we shall refer to $\{w^{*(i)}\}$ as the ‘‘true’’ importance weights and the $\{w^{(i)}\}$ and $\{\bar{w}^{(i)}\}$ as the unnormalized and normalized importance weights, respectively. In addition, all references to importance sampling will imply $\hat{I}_{N_p}(g)$ in (4.15), unless otherwise noted (as opposed to $\hat{I}_{N_p}^*(g)$ in (4.9)). Because $\hat{I}_{N_p}(g)$ is a ratio of estimates, it will typically be biased. However, the strong law of large numbers still applies, and thus $\hat{I}_{N_p}(g) \rightarrow I(g)$, with probability one, as $N_p \rightarrow \infty$. Comparing (4.9) and (4.15) reveals that the unknown true importance weights $\{w^{*(i)}\}$ are being

approximated by $\{N_p \bar{w}^{(i)}\}$ in $\hat{I}_{N_p}(g)$.

Before proceeding, it is insightful to rewrite the approximation $\hat{I}_{N_p}(g)$ provided by importance sampling in a slightly different way. Denoting the Dirac delta measure $\delta(\mathbf{x} - \mathbf{x}^{(i)})$ as $\delta_{\mathbf{x}^{(i)}}(\mathbf{x})$, we can express (4.15) as

$$\hat{I}_{N_p}(g) = \sum_{i=1}^{N_p} \left(\int g(\mathbf{x}) \delta_{\mathbf{x}^{(i)}}(\mathbf{x}) d\mathbf{x} \right) \bar{w}^{(i)} = \int g(\mathbf{x}) \sum_{i=1}^{N_p} \left(\bar{w}^{(i)} \delta_{\mathbf{x}^{(i)}}(\mathbf{x}) \right) d\mathbf{x}. \quad (4.17)$$

Then, defining the empirical (random) measure $\hat{p}_{N_p}(\mathbf{x}|\mathbf{z})$ generated by the samples $\{\mathbf{x}^{(i)}\}$ drawn from $\pi(\mathbf{x}|\mathbf{z})$ as

$$\hat{p}_{N_p}(\mathbf{x}|\mathbf{z}) \triangleq \sum_{i=1}^{N_p} \bar{w}^{(i)} \delta_{\mathbf{x}^{(i)}}(\mathbf{x}), \quad (4.18)$$

we can relate (4.1) and (4.17) as

$$\hat{I}_{N_p}(g) = \int g(\mathbf{x}) \hat{p}_{N_p}(\mathbf{x}|\mathbf{z}) d\mathbf{x} \approx \int g(\mathbf{x}) p(\mathbf{x}|\mathbf{z}) d\mathbf{x}, \quad (4.19)$$

where the approximation improves as $N_p \rightarrow \infty$. Equation (4.19) is quite useful because it indicates the sense in which importance sampling can be considered a density estimation technique. Thus, we can think of the outcome of the importance sampling algorithm as either an MC approximation for the integral $\int g(\mathbf{x}) p(\mathbf{x}|\mathbf{z}) d\mathbf{x}$ or an empirical estimate $\hat{p}_{N_p}(\mathbf{x}|\mathbf{z})$ of the posterior $p(\mathbf{x}|\mathbf{z})$. As it happens, both are different sides of the same coin. Because the focus of Bayesian filtering is the posterior $p(\mathbf{x}|\mathbf{z})$, we will adopt the latter interpretation.

In summary, the importance sampling algorithm provides us with a means of constructing an empirical approximation to the true posterior, without requiring the ability to sample from $p(\mathbf{x}|\mathbf{z})$. Unlike rejection sampling and the Metropolis-Hastings algorithm where $\{\mathbf{x}^{(i)}\}$ are distributed according to the true posterior,² the samples produced by importance sampling are distributed according to the proposal distribution $\pi(\mathbf{x}|\mathbf{z})$. However, with the calculation of the normalized importance weights $\{\bar{w}^{(i)}\}$, the importance sampling algorithm yields an empirical estimate of $p(\mathbf{x}|\mathbf{z})$, as indicated in (4.19). Furthermore, importance sampling is the only technique of the three that is appropriate for real-time applications because it is efficient (i.e., all samples are kept) and does not require a lengthy burn-in period. Having established the fundamentals of importance sampling, we now need to extend the algorithm for the sequential estimation of densities $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ for $k = 1, 2, \dots$, as opposed to the single-state form $p(\mathbf{x}|\mathbf{z})$ that we have been considering.

²More specifically, for rejection sampling, samples that are *accepted* are distributed according to $p(\mathbf{x}|\mathbf{z})$. Likewise, for the Metropolis-Hastings algorithm, samples drawn after the initial burn-in period are distributed according to $p(\mathbf{x}|\mathbf{z})$.

4.2 Sequential Importance Sampling

In this section, we introduce the sequential extension of the importance sampling algorithm. The sequential importance sampling (SIS) algorithm will allow us to approximate posterior densities of the form $p(\mathbf{x}_k | \mathbf{z}_{1:k})$, as required for Bayesian filtering. For the SIS algorithm to be recursive, we will have to constrain the form of the proposal distribution. The following discussion draws upon material from the tutorial by Arulampalam et al. [36].

In Section 4.1.3, we showed how importance sampling can be used to generate a random measure $\hat{p}_{N_p}(\mathbf{x} | \mathbf{z})$ that approximates the true posterior. From (4.18), we see that $\hat{p}_{N_p}(\mathbf{x} | \mathbf{z})$ is completely specified by the set $\{\mathbf{x}^{(i)}, \bar{w}^{(i)}\}_{i=1}^{N_p}$, where $\bar{w}^{(i)}$ is the normalized importance weight corresponding to $\mathbf{x}^{(i)}$. Thus, $\hat{p}_{N_p}(\mathbf{x} | \mathbf{z})$ consists of a set of (random) support points and their associated weights. In the context of Bayesian filtering, each support point is actually a randomly generated *sequence* of states, which we denote $\mathbf{x}_{0:k}^{(i)}$. In this case, our random measure $\hat{p}_{N_p}(\mathbf{x}_{0:k} | \mathbf{z}_{1:k})$ is specified by the set $\{\mathbf{x}_{0:k}^{(i)}, \bar{w}_k^{(i)}\}_{i=1}^{N_p}$, where the unnormalized weights introduced in (4.14) now satisfy

$$w_k^{(i)} = \frac{p(\mathbf{z}_{1:k} | \mathbf{x}_{0:k}^{(i)}) p(\mathbf{x}_{0:k}^{(i)})}{\pi(\mathbf{x}_{0:k}^{(i)} | \mathbf{z}_{1:k})}. \quad (4.20)$$

Thus, upon receipt of the current measurement \mathbf{z}_k , the sequential extension of the importance sampling algorithm must transform the random measure $\{\mathbf{x}_{0:k-1}^{(i)}, \bar{w}_{k-1}^{(i)}\}_{i=1}^{N_p}$ into $\{\mathbf{x}_{0:k}^{(i)}, \bar{w}_k^{(i)}\}_{i=1}^{N_p}$. If the proposal distribution is chosen to factor as

$$\pi(\mathbf{x}_{0:k} | \mathbf{z}_{1:k}) = \pi(\mathbf{x}_k | \mathbf{x}_{0:k-1}, \mathbf{z}_{1:k}) \pi(\mathbf{x}_{0:k-1} | \mathbf{z}_{1:k-1}), \quad (4.21)$$

then a sample $\mathbf{x}_{0:k}^{(i)} \sim \pi(\mathbf{x}_{0:k} | \mathbf{z}_{1:k})$ can be obtained by the following simple procedure:

1. Draw $\mathbf{x}_k^{(i)} \sim \pi(\mathbf{x}_k | \mathbf{x}_{0:k-1}^{(i)}, \mathbf{z}_{1:k})$.
2. Set $\mathbf{x}_{0:k}^{(i)} = \{\mathbf{x}_{0:k-1}^{(i)}, \mathbf{x}_k^{(i)}\}$.

Without the factorization in (4.21), drawing a sample from $\pi(\mathbf{x}_{0:k} | \mathbf{z}_{1:k})$ would require that the entire previous state history $\mathbf{x}_{0:k-1}$ be resampled. Clearly, as k grows, this will become infeasible for any real-time application. However, with the factorization in (4.21), we only need to draw N_p state vectors during each measurement update. This factorization of the proposal distribution also simplifies the form of the importance weights. Substituting (4.21) into (4.20) yields

$$w_k^{(i)} = \frac{p(\mathbf{z}_{1:k} | \mathbf{x}_{0:k}^{(i)}) p(\mathbf{x}_{0:k}^{(i)})}{\pi(\mathbf{x}_k^{(i)} | \mathbf{x}_{0:k-1}^{(i)}, \mathbf{z}_{1:k}) \pi(\mathbf{x}_{0:k-1}^{(i)} | \mathbf{z}_{1:k-1})} \quad (4.22)$$

$$= \frac{p(\mathbf{z}_k | \mathbf{x}_k^{(i)}) p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)})}{\pi(\mathbf{x}_k^{(i)} | \mathbf{x}_{0:k-1}^{(i)}, \mathbf{z}_{1:k})} \cdot \frac{p(\mathbf{z}_{1:k-1} | \mathbf{x}_{0:k-1}^{(i)}) p(\mathbf{x}_{0:k-1}^{(i)})}{\pi(\mathbf{x}_{0:k-1}^{(i)} | \mathbf{z}_{1:k-1})} \quad (4.23)$$

$$= \frac{p(\mathbf{z}_k | \mathbf{x}_k^{(i)}) p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)})}{\pi(\mathbf{x}_k^{(i)} | \mathbf{x}_{0:k-1}^{(i)}, \mathbf{z}_{1:k})} \cdot w_{k-1}^{(i)}, \quad (4.24)$$

where we used the Markov property of $\mathbf{x}_{0:k}$ and the conditional independence of $\mathbf{z}_{1:k}$ given $\mathbf{x}_{0:k}$ to go from (4.22) to (4.23).

Because we wish to perform recursive Bayesian filtering, the proposal distribution in the denominator of (4.24) must be modified further. First, because we do not wish to store the previous measurement sequence $\mathbf{z}_{1:k-1}$, the proposal distribution should be redefined as conditional on the current measurement \mathbf{z}_k only. Second, because we are interested in obtaining an estimate for the filtering density $p(\mathbf{x}_k|\mathbf{z}_{1:k})$, it would be preferable if we did not have to store the entire previous state history $\mathbf{x}_{0:k-1}^{(i)}$ in order to draw $\mathbf{x}_k^{(i)}$. This implies dropping the conditioning on $\mathbf{x}_{0:k-2}^{(i)}$ in the proposal distribution of (4.24). Applying these two modifications recursively to $\pi(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})$ yields the proposal distribution

$$\pi(\mathbf{x}_{0:k}|\mathbf{z}_{1:k}) = \pi(\mathbf{x}_0^{(i)}) \prod_{j=1}^k \pi(\mathbf{x}_j^{(i)}|\mathbf{x}_{j-1}^{(i)}, \mathbf{z}_j). \quad (4.25)$$

The unnormalized importance weights $\{w_k^{(i)}\}$ from (4.24) then assume the following simple form:

$$w_k^{(i)} = w_{k-1}^{(i)} \cdot \frac{p(\mathbf{z}_k|\mathbf{x}_k^{(i)})p(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)})}{\pi(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)}, \mathbf{z}_k)}. \quad (4.26)$$

Because the proposal distribution $\pi(\mathbf{x}_k|\mathbf{x}_{k-1}^{(i)}, \mathbf{z}_k)$ is no longer conditioned on the past measurement history $\mathbf{z}_{1:k-1}$, Equation (4.26) is suitable for recursive implementation. As such, it is the form of the SIS weights used most often by practitioners. However, regardless of whether (4.24) or (4.26) is used for $\{w_k^{(i)}\}$, the SIS algorithm yields the following empirical estimate of the filtering density:

$$\hat{p}_{N_p}(\mathbf{x}_k|\mathbf{z}_{1:k}) \triangleq \sum_{i=1}^{N_p} \bar{w}^{(i)} \delta_{\mathbf{x}_k^{(i)}}(\mathbf{x}_k) \approx p(\mathbf{x}_k|\mathbf{z}_{1:k}), \quad (4.27)$$

where the approximation is in the sense indicated in (4.19), and $\{\bar{w}^{(i)}\}$ are the normalized importance weights. Although we will present a crucial improvement to the SIS algorithm in the next section, we summarize our development thus far by the pseudo-code description given in Figure 4.2. The code highlights the simplicity of implementation that has made the SIS algorithm such a popular means of performing (approximate) recursive Bayesian filtering.

4.3 Degeneracy of the SIS Algorithm

In the last section, we introduced the sequential extension of the importance sampling algorithm. Although conceptually we now have a viable statistical approach for approximating a recursive Bayesian filter, the SIS algorithm has a significant practical shortcoming. We present this shortcoming in the following lemma.

Algorithm 1 Sequential Importance Sampling

$$\{\mathbf{x}_k^{(i)}, w_k^{(i)}\}_{i=1}^{N_p} = \text{SIS}[\{\mathbf{x}_{k-1}^{(i)}, w_{k-1}^{(i)}\}_{i=1}^{N_p}, \mathbf{z}_k]$$

- FOR $i = 1 : N_p$
 - Draw $\mathbf{x}_k^{(i)} \sim \pi(\mathbf{x}_k | \mathbf{x}_{k-1}^{(i)}, \mathbf{z}_k)$.
 - Calculate $w_k^{(i)}$ according to (4.26).
- END FOR

Figure 4.2: Pseudo-code description of the sequential importance sampling algorithm.

Lemma 3 *Degeneracy of Importance Weights*

The unconditional variance of the true importance weights cannot decrease with time,

$$\text{var}_{\pi(\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k-1})}[w_{k-1}^*] \leq \text{var}_{\pi(\mathbf{x}_{0:k}, \mathbf{z}_{1:k})}[w_k^*], \quad (4.28)$$

where the subscripts on the variances are meant to highlight that the expectations are taken with respect to both the measurement sequence and the imputed state sequence. Furthermore, using the variance decomposition

$$\text{var}_{\pi(\mathbf{x}_{0:k}, \mathbf{z}_{1:k})}[w_k^*] = E_{p(\mathbf{z}_{1:k})}[\text{var}_{\pi(\mathbf{x}_{0:k} | \mathbf{z}_{1:k})}[w_k^* | \mathbf{z}_{1:k}]] \quad (4.29)$$

in (4.28) indicates that, for a given (nonrandom) sequence $\mathbf{z}_{1:k}$, the conditional variance of the true importance weights will tend to follow an increasing trend (although not strictly so).

Proof:

See Appendix A.

Because $w_k = p(\mathbf{z}_{1:k})w_k^*$ (compare Equations (4.10) and (4.14)), the conditional variance of w_k will follow the same increasing trend as well. Thus, while Lemma 3 does not rule out the possibility that, for a given realization $\mathbf{z}_{1:k}$, $\text{var}[w_k | \mathbf{z}_{1:k}]$ could actually *decrease* with k , in real applications this is never found [45, 53]. Instead, it is reported that after a few iterations of the SIS algorithm, most of the normalized importance weights will be nearly equal to zero. This is detrimental for two reasons. First, the quality of the random measure $\sum_{i=1}^{N_p} \bar{w}^{(i)} \delta_{\mathbf{x}_k^{(i)}}(\mathbf{x}_k)$, as well as any estimates based upon it, can be expected to deteriorate as more weights become equal to zero. Second, from a computational standpoint, it is inefficient to expend resources calculating support points and weights $\{\mathbf{x}_k^{(i)}, \bar{w}_k^{(i)}\}$ whose contributions to estimates produced by the filter are negligible. This is known in the literature as the problem of *degeneracy* [36]. In this section, we will discuss two approaches to reducing its occurrence.

4.3.1 Choice of the proposal distribution

When we introduced importance sampling in Section 4.1.3, the only constraint placed upon the proposal distribution $\pi(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})$ was that its support must contain the support of the true distribution $p(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})$. When we extended the algorithm to a sequential setting, we enforced the additional constraint that the proposal distribution should factor as $\pi(\mathbf{x}_{0:k}|\mathbf{z}_{1:k}) = \pi(\mathbf{x}_0) \prod_{j=1}^k \pi(\mathbf{x}_j|\mathbf{x}_{j-1}, \mathbf{z}_j)$ for each k . Other than these two requirements, we were free to use whatever distribution was convenient. Unfortunately, the SLLN convergence results for the SIS algorithm apply as $N_p \rightarrow \infty$. Thus, to obtain satisfactory performance for finite N_p , more care is required when choosing $\pi(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})$. In the context of alleviating the degeneracy of the SIS algorithm, a sensible choice of $\pi(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{z}_k)$ is the distribution that minimizes the conditional variance of the importance weights.³ This optimal proposal distribution is given in the following lemma.

Lemma 4 *Optimal Proposal Distribution*

The proposal distribution that minimizes $\text{var}_\pi[w_k^{(i)}|\mathbf{x}_{k-1}^{(i)}, \mathbf{z}_k]$ is

$$\pi_{opt}(\mathbf{x}_k|\mathbf{x}_{k-1}^{(i)}, \mathbf{z}_k) = p(\mathbf{x}_k|\mathbf{x}_{k-1}^{(i)}, \mathbf{z}_k). \quad (4.30)$$

Proof:

Beginning with (4.24), we have

$$w_k^{(i)} = w_{k-1}^{(i)} \cdot \frac{p(\mathbf{z}_k|\mathbf{x}_k^{(i)}) p(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)})}{\pi_{opt}(\mathbf{x}_k|\mathbf{x}_{k-1}^{(i)}, \mathbf{z}_k)} \quad (4.31)$$

$$= w_{k-1}^{(i)} \cdot \frac{p(\mathbf{x}_k^{(i)}, \mathbf{z}_k|\mathbf{x}_{k-1}^{(i)})}{p(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)}, \mathbf{z}_k)} \quad (4.32)$$

$$= w_{k-1}^{(i)} p(\mathbf{z}_k|\mathbf{x}_{k-1}^{(i)}), \quad (4.33)$$

where we used the conditional independence of \mathbf{z}_k given \mathbf{x}_k to go from (4.31) to (4.32). Thus, for the proposal distribution suggested in (4.30), the weight $w_k^{(i)}$ is conditionally independent of the actual draw of the current state $\mathbf{x}_k^{(i)}$, or $\text{var}_{\pi_{opt}}[w_k^{(i)}|\mathbf{x}_{k-1}^{(i)}, \mathbf{z}_k] = 0$, as stated.

There are two problems with the optimal proposal distribution. First, it requires the ability to sample from $p(\mathbf{x}_k|\mathbf{x}_{k-1}^{(i)}, \mathbf{z}_k)$, a distribution that may be nonstandard. Second, calculation of $w_k^{(i)}$ as specified in (4.33) requires that we evaluate the integral

$$p(\mathbf{z}_k|\mathbf{x}_{k-1}^{(i)}) = \int p(\mathbf{z}_k|\mathbf{x}_k) p(\mathbf{x}_k|\mathbf{x}_{k-1}^{(i)}) d\mathbf{x}_k, \quad (4.34)$$

a task that may be analytically intractable. Similar to our discussion on exact recursive Bayesian algorithms in Section 3.1, there are two cases when the use of π_{opt} is

³Because of the factorization required by (4.25), $\pi(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})$ is entirely specified by $\pi(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{z}_k)$ and $\pi(\mathbf{x}_0)$.

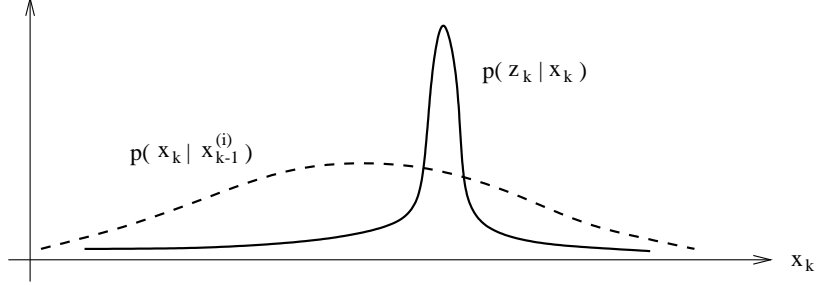


Figure 4.3: Example of how use of the prior $p(\mathbf{x}_k | \mathbf{x}_{k-1}^{(i)})$ as proposal distribution can lead to degeneracy of the importance weights.

possible [36]. The first is when \mathbf{x}_k is from a discrete finite state space. In that case, the integral in (4.34) becomes a sum and sampling from $p(\mathbf{x}_k | \mathbf{x}_{k-1}^{(i)}, \mathbf{z}_k)$ is possible. The second case occurs for dynamic models with additive Gaussian noise processes and linear measurement equations.⁴ With this model, Equation (4.34) can be solved analytically because both terms in the integrand are Gaussian. Furthermore, because the measurement process is linear, $p(\mathbf{x}_k | \mathbf{x}_{k-1}^{(i)}, \mathbf{z}_k)$ is also Gaussian and can be sampled. In general, such analytic evaluations are not possible for π_{opt} , and researchers in the field have spent a good deal of effort developing approximations to the optimal proposal distribution (see [45] for an overview). Some recent approaches include the auxiliary particle filter [54] and approximations based upon the unscented transformation [55, 56] and the Metropolis-Hastings algorithm [57, 58]. We will return to this topic in Section 4.3.4 when we discuss the auxiliary particle filter.

Before discussing the second method for mitigating the problem of degeneracy, we mention one more possibility for the proposal distribution. Although it may be far from optimal, a popular choice among practitioners is the so-called prior distribution

$$\pi(\mathbf{x}_k | \mathbf{x}_{k-1}^{(i)}, \mathbf{z}_k) = p(\mathbf{x}_k | \mathbf{x}_{k-1}^{(i)}). \quad (4.35)$$

Sampling from the prior is often straightforward. For example, in the case of an additive noise model, $\mathbf{x}_k = f_k(\mathbf{x}_{k-1}) + \mathbf{u}_k$, sampling from $p(\mathbf{x}_k | \mathbf{x}_{k-1}^{(i)})$ amounts to sampling from the noise distribution $p(\mathbf{u}_k)$. Furthermore, the weight update equation assumes an even simpler form,

$$w_k^{(i)} = w_{k-1}^{(i)} p(\mathbf{z}_k | \mathbf{x}_k^{(i)}). \quad (4.36)$$

Despite these benefits, use of $p(\mathbf{x}_k | \mathbf{x}_{k-1}^{(i)})$ as the proposal distribution can lead to poor performance. Figure 4.3 demonstrates why this might be so. In the figure, the likelihood $p(\mathbf{z}_k | \mathbf{x}_k)$ is much more peaked than $p(\mathbf{x}_k | \mathbf{x}_{k-1}^{(i)})$. Thus, if we were to use the prior as the proposal distribution for this case, Equation (4.36) indicates that many of the resulting samples $\mathbf{x}_k^{(i)}$ could have negligible weights. However, this is precisely

⁴This is the standard Kalman filter framework except the relationship between the previous state and the current state is allowed to be nonlinear: $\mathbf{x}_k = f_k(\mathbf{x}_{k-1}) + \mathbf{u}_k$.

the problem we were trying to alleviate by appropriate choice of $\pi(\mathbf{x}_k | \mathbf{x}_{k-1}^{(i)}, \mathbf{z}_k)$. In words, the prior as proposal distribution is suboptimal because it does not take the current measurement \mathbf{z}_k into account. In cases where the variance of the system noise is significantly greater than the variance of the measurement noise (as shown in Figure 4.3), the prior tends to be a poor choice for the proposal distribution.

4.3.2 Resampling

The second method for reducing the effects of degeneracy in the SIS algorithm is *resampling*. Recall that the SIS algorithm provides

$$\hat{p}_{N_p}(\mathbf{x}_k | \mathbf{z}_{1:k}) \triangleq \sum_{i=1}^{N_p} \bar{w}_k^{(i)} \delta_{\mathbf{x}_k^{(i)}}(\mathbf{x}_k)$$

as a discrete approximation to the true posterior $p(\mathbf{x}_k | \mathbf{z}_{1:k})$. Resampling can be thought of as drawing N_p independent samples from $\hat{p}_{N_p}(\mathbf{x}_k | \mathbf{z}_{1:k})$. In the following discussion, we will use the notation $\tilde{\mathbf{x}}_k^{(i)}$ to denote a state vector *before* resampling and $\mathbf{x}_k^{(i)}$ to denote a state vector *after* resampling. The resampling procedure can then be described as mapping

$$\left\{ \tilde{\mathbf{x}}_k^{(i)}, \bar{w}_k^{(i)} \right\}_{i=1}^{N_p} \longrightarrow \left\{ \mathbf{x}_k^{(i)}, \frac{1}{N_p} \right\}_{i=1}^{N_p}, \quad (4.37)$$

such that

$$\Pr(\mathbf{x}_k^{(i)} = \tilde{\mathbf{x}}_k^{(j)}) = \bar{w}_k^{(j)}. \quad (4.38)$$

Because resampling draws (in an approximate sense) from the true posterior, the resampled importance weights are uniform, as shown in (4.37). Thus, resampling prevents the SIS algorithm from degenerating by constructing a new random measure where all support points have weight $1/N_p$. Equation (4.38) indicates that resampling tends to multiply those states with significant weights while discarding those with negligible weights. In this sense, resampling can be considered as focusing the “attention” of the SIS algorithm on the most promising areas of the state space.

In order to know when to resample, we need a way of monitoring the degeneracy of the SIS algorithm. In [53], Kong et al. suggested the *effective sample size*, N_{eff} , as a measure of degeneracy. To arrive at their definition for N_{eff} , we first introduce the efficiency ratio

$$\eta_k(\pi, p) \triangleq \frac{\text{var}_{\pi}[\hat{I}_{N_p}(g) | \mathbf{z}_{1:k}]}{\text{var}_p[I_{N_p}(g) | \mathbf{z}_{1:k}]}, \quad (4.39)$$

where $\hat{I}_{N_p}(g)$ is the estimate of the integral $\int g(\mathbf{x}_k) p(\mathbf{x}_k | \mathbf{z}_{1:k}) d\mathbf{x}_k$ obtained by importance sampling (see (4.15)). Similarly, $I_{N_p}(g)$ is the Monte Carlo estimate of the same integral, except the samples are drawn from the true posterior (see (4.2)). As such, $\eta_k(\pi, p)$ is a measure of the inefficiency caused by not being able to sample from $p(\mathbf{x}_k | \mathbf{z}_{1:k})$. It is reasonable to expect the conditional variance $\text{var}_{\pi}[\hat{I}_{N_p}(g) | \mathbf{z}_{1:k}]$ to increase as the importance weights degenerate.

Ideally, we would like to define the effective sample size as the ratio $N_p / \eta_k(\pi, p)$,

but the efficiency ratio is typically unknown. Instead, we use the following result from [48, 53] to approximate $\eta_k(\pi, p)$.

Lemma 5 *Approximation for Efficiency Factor*

For functions $g(\mathbf{x}_k)$ that vary slowly with \mathbf{x}_k ,

$$\frac{\text{var}_\pi[\hat{I}_{N_p}(g)|\mathbf{z}_{1:k}]}{\text{var}_p[g|\mathbf{z}_{1:k}]} \approx \frac{\text{var}_\pi[w_k^*|\mathbf{z}_{1:k}] + 1}{N_p}. \quad (4.40)$$

Proof:

See Appendix B.

With this result, we have

$$\frac{N_p}{\eta_k(\pi, p)} = N_p \cdot \frac{\text{var}_p[\hat{I}_{N_p}(g)|\mathbf{z}_{1:k}]}{\text{var}_\pi[\hat{I}_{N_p}(g)|\mathbf{z}_{1:k}]} = \frac{\text{var}_p[g|\mathbf{z}_{1:k}]}{\text{var}_\pi[\hat{I}_{N_p}(g)|\mathbf{z}_{1:k}]} \approx \frac{N_p}{\text{var}_\pi[w_k^*|\mathbf{z}_{1:k}] + 1}.$$

Thus, as an approximation to $N_p/\eta_k(\pi, p)$, the effective sample size is defined as

$$N_{\text{eff}} \triangleq \frac{N_p}{\text{var}_\pi[w_k^*|\mathbf{z}_{1:k}] + 1}. \quad (4.41)$$

There are two practical difficulties with the use of N_{eff} . First, as discussed in Section 4.1.3, we generally cannot calculate the true importance weights $\{w_k^{*(i)}\}$ because $p(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})$ can only be evaluated up to a constant of proportionality. Second, even if $p(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})$ can be evaluated, a closed-form expression for $\text{var}_\pi[w_k^*|\mathbf{z}_{1:k}]$ is typically not available. Instead, recognizing that $(\text{var}_\pi[w_k^*|\mathbf{z}_{1:k}] + 1) = \text{E}_\pi[w_k^{*2}|\mathbf{z}_{1:k}]$, we use the approximation $N_p \bar{w}_k^{(i)} \approx w_k^{*(i)}$ (compare (4.9) and (4.15)) to estimate the effective sample size,

$$\hat{N}_{\text{eff}} \triangleq \frac{N_p}{\frac{1}{N_p} \sum_{i=1}^{N_p} (N_p \bar{w}_k^{(i)})^2} = \frac{1}{\sum_{i=1}^{N_p} (\bar{w}_k^{(i)})^2}, \quad (4.42)$$

where $\{\bar{w}_k^{(i)}\}$ are the normalized importance weights. As a heuristic check of the validity of (4.42), consider the two extreme distributions for $\{\bar{w}_k^{(i)}\}$. In the best-case scenario, the normalized weights will be uniform, $\bar{w}_k^{(i)} = 1/N_p$ for all i , yielding $\hat{N}_{\text{eff}} = N_p$. In the worst-case scenario, all but a single weight will be zero, yielding $\hat{N}_{\text{eff}} = 1$. Thus, we find that \hat{N}_{eff} agrees with our intuition as a measure of the effective sample size. With \hat{N}_{eff} , we now have a simple yet powerful way of monitoring the degeneracy of the SIS algorithm. After each measurement update, \hat{N}_{eff} is calculated and compared to a predetermined threshold, which we denote N_{thresh} . Resampling occurs if $\hat{N}_{\text{eff}} < N_{\text{thresh}}$.

There are many different ways to implement the actual resampling operation such that (4.38) is satisfied. Define $N_p^{(i)}$ as the discrete random variable corresponding to the number of times $\tilde{\mathbf{x}}_k^{(i)}$ is selected during resampling. (We suppress the subscript k on $N_p^{(i)}$ for convenience.) Thus, $\sum_{i=1}^{N_p} N_p^{(i)} = N_p$. We recognize that $\{N_p^{(i)}\}$ obeys a multinomial probability law. As such, resampling directly from the probability mass function $\{\bar{w}^{(i)}\}$ will result in $\text{E}[N_p^{(i)}|\bar{w}^{(i)}] = N_p \bar{w}^{(i)}$ and $\text{var}[N_p^{(i)}|\bar{w}^{(i)}] = N_p \bar{w}^{(i)}(1 -$

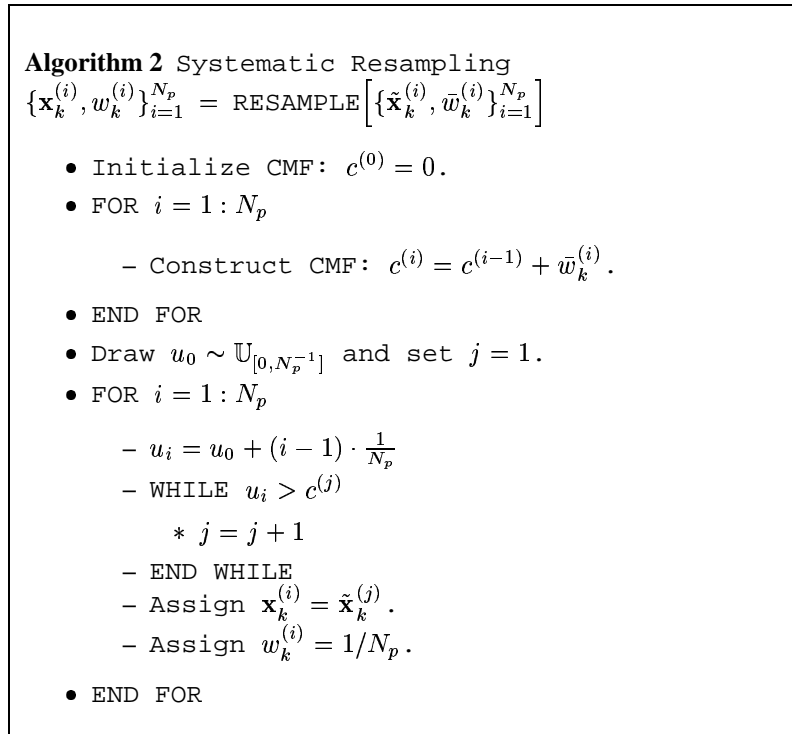


Figure 4.4: Pseudo-code description of the systematic resampling algorithm.

$\bar{w}^{(i)}$) for each i . However, as mentioned in [59], it is preferable to use a resampling scheme that reduces the variances $\text{var}[N_p^{(i)}|\bar{w}^{(i)}]$. This should, in turn, reduce the chance that resampling actually *increases* the extent of degeneracy by selecting too many states with negligible weights.

In our particle filter, we implement a scheme known as *systematic resampling* [60]. The algorithm is presented in Figure 4.4. Note that $\{c^{(i)}\}_{i=0}^{N_p}$ is the cumulative mass function for the discrete density corresponding to the normalized weights,

$$c^{(i)} = \sum_{j=1}^i \bar{w}^{(j)}, \quad \text{for } i = 1, 2, \dots, N_p,$$

and $c^{(0)} = 0$. The main loop in the algorithm consists of comparing a value u_i generated on the interval $[0, 1]$ to the cumulative mass function and selecting the index $\min_j \{j : c^{(j)} \geq u_i\}$. It is easiest to understand systematic resampling by considering a simple example. In Figure 4.5, an illustration is provided for $N_p = 3$. The locations of the dashed lines indicate that the states $\{\tilde{\mathbf{x}}_k^{(1)}, \tilde{\mathbf{x}}_k^{(3)}, \tilde{\mathbf{x}}_k^{(3)}\}$ have been selected by the resampling operation. Thus, the particle $\tilde{\mathbf{x}}_k^{(3)}$ has been multiplied at the expense of discarding $\tilde{\mathbf{x}}_k^{(2)}$, which had a lower normalized weight. It is important to notice that the only random element in the systematic resampling algorithm is the draw for u_0 ; the other selection points $\{u_i\}_{i=1}^{N_p}$ are deterministic given u_0 . As such, we have the following proposition.

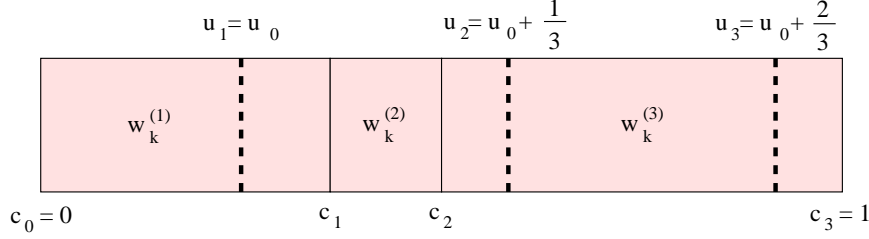


Figure 4.5: Illustration of systematic resampling for $N_p = 3$. The locations of the dashed lines indicate which states have been selected, in this case $\{\tilde{\mathbf{x}}_k^{(1)}, \tilde{\mathbf{x}}_k^{(3)}, \tilde{\mathbf{x}}_k^{(3)}\}$.

Proposition 3 *Reduced Variation of Systematic Resampling*

The systematic resampling algorithm yields a set of selection counts $\{N_p^{(i)}\}_{i=1}^{N_p}$ that satisfy $E[N_p^{(i)} | \bar{w}^{(i)}] = N_p \bar{w}^{(i)}$ and $\text{var}[N_p^{(i)} | \bar{w}^{(i)}] = N_p \Delta^{(i)} (1 - N_p \Delta^{(i)})$ for all i , where

$$N_p \Delta^{(i)} = N_p \bar{w}^{(i)} - \left\lfloor \frac{\bar{w}^{(i)}}{N_p} \right\rfloor. \quad (4.43)$$

The conditional variance in (4.43) is guaranteed to be strictly less than the conditional variance of the multinomial probability law.

Although resampling alleviates the effects of degeneracy, it does introduce two new problems [36, 45]. First, from a practical point of view, resampling limits the ability to parallelize the SIS algorithm because all weights $w_k^{(i)}$ must be summed during normalization. Second, from a theoretical perspective, the samples $\{\mathbf{x}_k^{(i)}\}$ will not be statistically independent after resampling. As such, we lose the simple convergence results discussed in Section 4.1.3. Nonetheless, under weak assumptions, it is still possible to guarantee almost sure convergence of the empirical distributions generated by the SIS algorithm toward the true ones, although the proofs become more involved [47, 61]. For now, we highlight the problem of *sample impoverishment*. Sample impoverishment occurs when a state $\mathbf{x}_k^{(i)}$ (or equivalently, a sequence $\mathbf{x}_{0:k}^{(i)}$) with high importance weight is selected many times during resampling, leading to a loss of diversity among the support points. Even though the next iteration of importance sampling “re-diversifies” the support set $\{\mathbf{x}_{k+1}^{(i)}\}$, sample impoverishment can still have a detrimental effect if we are interested in fixed-lag or fixed-point estimates. Although we have been considering the specific case of filtered estimates thus far, it is straightforward to show that, for $L > 0$,

$$\sum_{i=1}^{N_p} \bar{w}_k^{(i)} g(\mathbf{x}_{k-L}^{(i)}) \quad (4.44)$$

can be used as an approximation for the fixed-lag expectation $E_p[g(\mathbf{x}_{k-L}) | \mathbf{z}_{1:k}]$, as long as we are willing to maintain a partial state history

$$\mathbf{x}_{k-L:k}^{(i)} = \left\{ \mathbf{x}_{k-L}^{(i)}, \mathbf{x}_{k-L+1}^{(i)}, \dots, \mathbf{x}_k^{(i)} \right\}$$

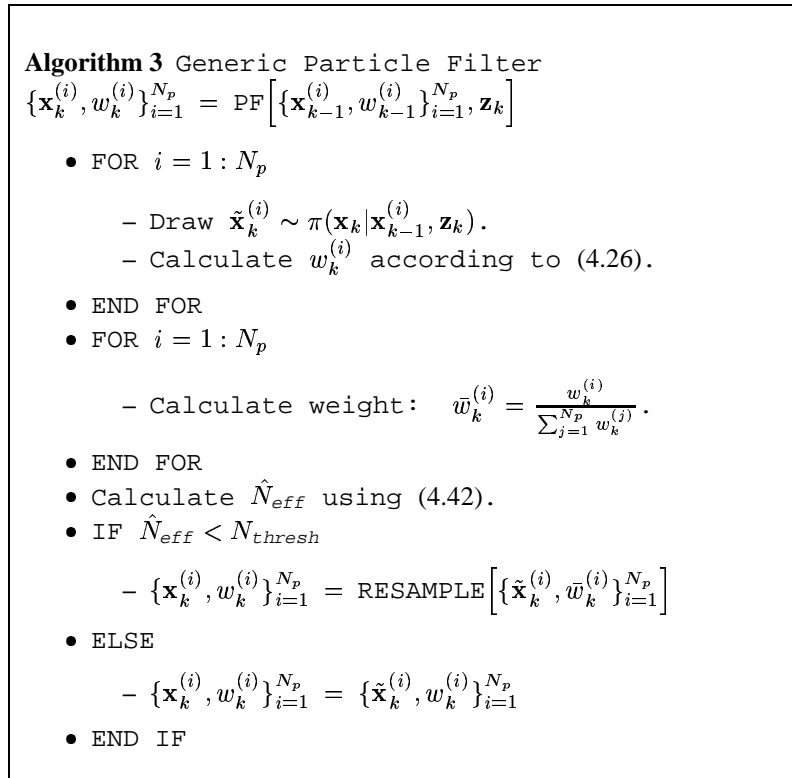


Figure 4.6: Pseudo-code description of a generic particle filtering algorithm.

for each i . In this case, if repeated resampling results in the trajectories $\mathbf{x}_{k-L:k}^{(i)}$ that all share a few common ancestors at time $k - L$, then we expect the quality of the estimate in (4.44) to be poor. However, in spite of these two difficulties, the need to prevent degeneracy of the SIS algorithm is more important, and resampling has become a component of virtually all SIS implementations.

4.3.3 Generic particle filter

With \hat{N}_{eff} as a monitor of degeneracy and systematic resampling as a corrective measure, we can now define an extension to the SIS algorithm. Figure 4.6 contains a pseudo-code description of an algorithm that incorporates both importance sampling and resampling. This algorithm has become increasingly popular over the last decade and is known variously as particle filtering [62], the Bayesian bootstrap [11], the condensation algorithm [13], and interacting particle approximations [63]. In this report, we will adopt the term *particle filter* for the algorithm in Figure 4.6, as its usage has become prevalent in the literature. Furthermore, we will refer to the support set $\{\mathbf{x}_k^{(i)}\}_{i=1}^{N_p}$ as “particles.” In Chapter 6, we will show how the particle filter’s flexibility enables us to perform recursive Bayesian filtering for the challenging task of RCS-based joint tracking/classification.

4.3.4 Auxiliary particle filter

According to the pseudo-code description in Figure 4.6, resampling will occur if the particle weights become too skewed. Although resampling results in a uniformly weighted sample, it also leads to a loss of diversity (i.e., sample impoverishment). As such, it is important to choose a proposal distribution that provides particles with relatively uniform weights. To accomplish this, it is necessary to incorporate the current measurement \mathbf{z}_k in the proposal distribution. In our case, the lack of a closed-form expression for RCS makes this difficult. In this section, we will describe an extension to the generic particle filtering algorithm that allows us to do so. The *auxiliary particle filter* (APF) was originally introduced in [54] as an alternative to using the prior $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ as the proposal distribution. Our presentation follows that of [36]. After providing its key equations, we will demonstrate how the APF can be used in our application.

The APF operates by obtaining a sample from the joint density $p(\mathbf{x}_k, i|\mathbf{z}_{1:k})$, where i is the *auxiliary variable* that represents the index of the particle at scan $k-1$ from which \mathbf{x}_k is predicted (i.e., $\mathbf{x}_{k-1}^{(i)} \rightarrow \mathbf{x}_k$). Using Bayes' rule, we can express this joint density as

$$p(\mathbf{x}_k, i|\mathbf{z}_{1:k}) \propto p(\mathbf{z}_k|\mathbf{x}_k, i, \mathbf{z}_{1:k-1})p(\mathbf{x}_k|i, \mathbf{z}_{1:k-1})p(i|\mathbf{z}_{1:k-1}) \quad (4.45)$$

$$= p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{x}_{k-1}^{(i)})w_{k-1}^{(i)}. \quad (4.46)$$

Next, we select the importance function to satisfy the proportionality

$$\pi(\mathbf{x}_k, i|\mathbf{z}_{1:k}) \propto p(\mathbf{z}_k|\lambda_k^{(i)})p(\mathbf{x}_k|\mathbf{x}_{k-1}^{(i)})w_{k-1}^{(i)}, \quad (4.47)$$

where $\lambda_k^{(i)}$ is some characterization of \mathbf{x}_k given $\mathbf{x}_{k-1}^{(i)}$. For example, $\lambda_k^{(i)}$ could be the conditional mean, $\lambda_k^{(i)} = \mathbb{E}[\mathbf{x}_k|\mathbf{x}_{k-1}^{(i)}]$. With these definitions, a sample $(\mathbf{x}_k^{(j)}, i^{(j)})$ drawn from the proposal $\pi(\mathbf{x}_k, i|\mathbf{z}_{1:k})$ will have an importance weight proportional to

$$w_k^{(j)} \propto \frac{p(\mathbf{x}_k^{(j)}, i^{(j)}|\mathbf{z}_{1:k})}{\pi(\mathbf{x}_k^{(j)}, i^{(j)}|\mathbf{z}_{1:k})} \propto \frac{p(\mathbf{z}_k|\mathbf{x}_k^{(j)})}{p(\mathbf{z}_k|\lambda_k^{(i^{(j)})})}, \quad (4.48)$$

where the second proportionality is obtained by taking the ratio of (4.46) to (4.47). The weighted sample set $\{\mathbf{x}_k^{(j)}, i^{(j)}, w_k^{(j)}\}_{j=1}^{N_p}$ is then distributed (approximately) as $p(\mathbf{x}_k, i|\mathbf{z}_{1:k})$. By simply discarding the auxiliary variables $\{i^{(j)}\}$, we obtain the desired sample from $p(\mathbf{x}_k|\mathbf{z}_{1:k})$.

At first sight, it might not be apparent how this alleviates the task of incorporating the current measurement \mathbf{z}_k into the proposal distribution. To clarify this, we factor the proposal as

$$\pi(\mathbf{x}_k, i|\mathbf{z}_{1:k}) = \pi(i|\mathbf{z}_{1:k})\pi(\mathbf{x}_k|i, \mathbf{z}_{1:k}) \quad (4.49)$$

$$\propto \left(p(\mathbf{z}_k|\lambda_k^{(i)})w_{k-1}^{(i)} \right) p(\mathbf{x}_k|\mathbf{x}_{k-1}^{(i)}), \quad (4.50)$$

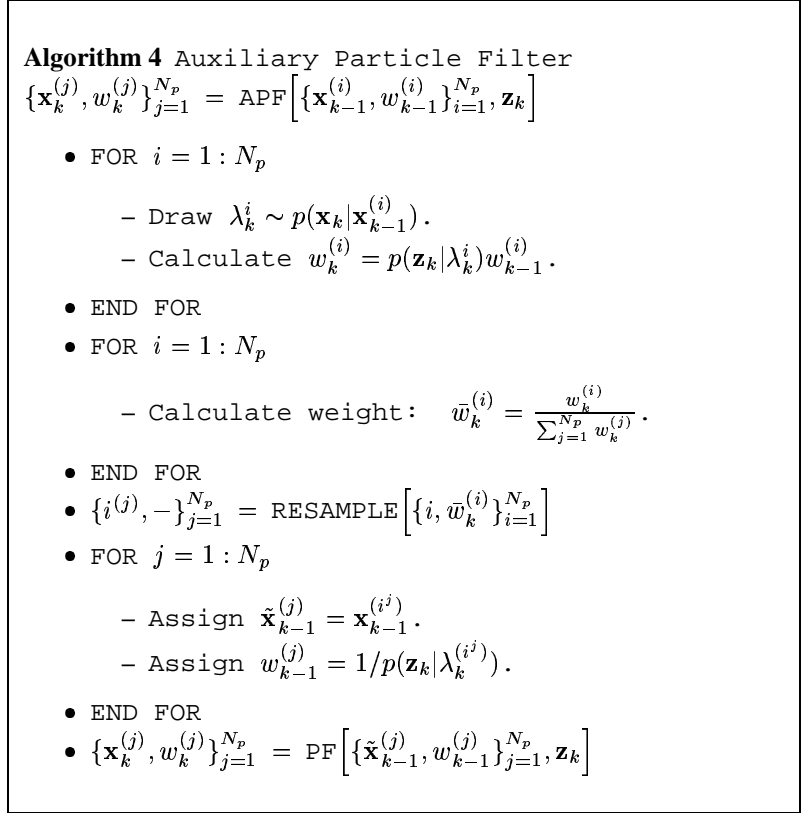


Figure 4.7: Pseudo-code description of the auxiliary particle filter with $\lambda_k^{(i)} \sim p(\mathbf{x}_k | \mathbf{x}_{k-1}^{(i)})$.

where we have regrouped the terms from the definition of $\pi(\mathbf{x}_k, i | \mathbf{z}_{1:k})$ in (4.47) to match the factorization in (4.49). Thus, a sample $(\mathbf{x}_k^{(j)}, i^{(j)})$ is generated in two steps.

1. Draw a sample of the previous index: $i^{(j)} \sim p(\mathbf{z}_k | \lambda_k^{(i)}) w_{k-1}^{(i)}$.
2. Given $i^{(j)}$, draw a sample of the current state: $\mathbf{x}_k^{(j)} \sim p(\mathbf{x}_k | \mathbf{x}_{k-1}^{(i^j)})$.

Thus, we now see how the APF incorporates the current measurement into its proposal distribution. Namely, the likelihood $p(\mathbf{z}_k | \lambda_k^{(i)})$ is used to select *previous* samples $\mathbf{x}_{k-1}^{(i)}$ that are likely to lead to current samples that are well-matched to \mathbf{z}_k . What remains is for us to specify $\lambda_k^{(i)}$. A popular choice is to define $\lambda_k^{(i)}$ as a sample from the prior, $\lambda_k^{(i)} \sim p(\mathbf{x}_k | \mathbf{x}_{k-1}^{(i)})$. With this choice of $\lambda_k^{(i)}$, we obtain the pseudo-code description of the APF algorithm provided in Figure 4.7.

There are three important things to note concerning the APF implementation in Figure 4.7. First, the generic particle filter routine is called at the end of the APF routine. Thus, when $\lambda_k^{(i)}$ is chosen to be a sample from $p(\mathbf{x}_k | \mathbf{x}_{k-1}^{(i)})$, each APF iteration is actually equivalent to *two* iterations of the generic particle filter. This is because the first set of state samples is discarded after the auxiliary variables $\{i^{(j)}\}$ have been chosen. Basically, the APF performs a “dry run” with the current data to see which

of the previous states are most promising. Then, when resampling to obtain $\{i^{(j)}\}$, previous states that are promising are reselected many times. Those that do not match \mathbf{z}_k (when predicted to the current time) are discarded. This way, multiple states $\mathbf{x}_k^{(j)}$ will be generated from each promising previous state. This leads us to our second point. The APF algorithm essentially performs resampling *before* state prediction and weight update, as opposed to the generic particle filter, which resamples *after* these operations have occurred. As such, sample impoverishment should be less of a problem for the auxiliary particle filter. Finally, note that the APF with $\lambda_k^{(i)} \sim p(\mathbf{x}_k | \mathbf{x}_{k-1}^{(i)})$ only requires the ability to sample from the prior and evaluate the likelihood function up to a constant of proportionality. Therefore, it is perfectly suited for use with a complex measurement like RCS.

In summary, the auxiliary particle filter offers the ability to incorporate the current measurement into the proposal distribution. Because the implementation in Figure 4.7 requires only that we be able to sample from the prior, RCS can be incorporated into the data vector. In Chapter 7, we will investigate the trade-off between filter performance and computational cost for the APF from this section and a particle filter that uses the prior as proposal distribution.

CHAPTER 5

KALMAN FILTER-BASED TRACKING

In this work, we are primarily interested in formulating a joint approach for FM-band passive radar tracking and classification. In Chapter 2, we showed that radar cross section has considerable potential as a discriminatory feature in the VHF band. Unfortunately, results from the same chapter also highlighted the complex relationship that exists between RCS and the angular aspect of the target. In Chapter 3, we specified that our solution should be both recursive and (approximately) Bayesian optimal. However, we discovered that the complex nature of RCS eliminated most popular Kalman filter-based algorithms from consideration. Finally, in Chapter 4, we found that sampling-based algorithms, such as the particle filter, offered the flexibility needed to incorporate RCS within the measurement vector.

In the next chapter, we will define both the state model f_k and the measurement model h_k for our particle filter. Both are key results because they demonstrate not only that tracking and classification *can* be performed jointly, but that they *should* be performed jointly, whenever possible. To substantiate our claims, extensive experimental results will be provided in Chapter 7. However, in order for these results to be meaningful, we need a benchmark against which performance comparisons can be made. That is the subject of the current chapter.

In what follows, we will derive an EKF-based filter for FM-band passive radar. As discussed in Section 3.2.1, the lack of a closed-form relationship for RCS prevents it from being used in an extended Kalman filter. Because this leaves delay and Doppler shift as measurements, our benchmark system will have tracking capabilities only. In order to make the system as robust as possible against maneuvering targets, we follow popular practice and implement it using the Interacting Multiple Model (IMM) algorithm from Section 3.2.1. Thus, we will refer to our benchmark filter as the IMM-EKF. In the following sections, we define the state models $\{f_k^{(\gamma)}\}$ for the IMM algorithm and the measurement model h_k . In addition, because the filter must operate in the presence of clutter and missed detections, we also discuss data association.

5.1 State Models for the IMM-EKF

As presented in Section 3.2.1, in many tracking applications, the same target maneuvers in different fashions at different times. For example, an airplane that is flying at a constant speed and fixed altitude may suddenly begin to climb or turn sharply. In cases such as these, it is unlikely that a single state model f_k will accurately represent the target's behavior at all times. Instead, a jump Markov system (JMS) often provides a better model of the underlying dynamics. The JMS dynamic system was defined in Equations (3.39) and (3.40). In this section, we specify the state model for the JMS.

For our application, we will assume a jump Markov state model of the form

$$\mathbf{x}_k = F_k^{(\gamma_k)} \mathbf{x}_{k-1} + \mathbf{u}_k^{(\gamma_k)}, \quad (5.1)$$

where $\{\gamma_k\}$ is the Markov chain representing the maneuver sequence. The terms in $\{\gamma_k\}$ take values in $\{1, \dots, N\}$, where N is the number of separate Kalman filters used in the implementation. Given the previous state γ_{k-1} , the current state is drawn according to the probabilities in the Markov transition matrix P_{IMM} , where $P_{IMM}(i, j) = \Pr(\gamma_k = j | \gamma_{k-1} = i)$ for any k . Note that (5.1) is a linear state model, whereas (3.39) represents the more general nonlinear case. Furthermore, we will assume that each of the N noise sequences are zero-mean white Gaussian, $\mathbf{u}_k^{(\gamma)} \sim \mathcal{N}(\mathbf{0}, Q_k^{(\gamma)})$. In this case, the following three pieces of information are required to specify our state model:

1. the number of models (or separate extended Kalman filters) N ,
2. the Markov transition matrix P_{IMM} ,
3. the parameters $\{F_k^{(\gamma)}, Q_k^{(\gamma)}\}_{\gamma=1}^N$ for the models.

For our IMM-EKF, we choose $N = 3$. As expected, each model is associated with a different type of motion. The first is a constant velocity (CV) model, the second is a constant acceleration (CA) model, and the third is a coordinated turn (CT) model. This is a popular choice of models, which has proven to be robust against maneuvering targets [40,64]. We defer specifying values for the probabilities in P_{IMM} until Chapter 7. This leaves us with the task of specifying $\{F_k^{(\gamma)}, Q_k^{(\gamma)}\}$ for each γ . Before doing so, we must comment on the general nature of these matrices. Because the targets we wish to track are free to climb and descend, the extended Kalman filters must be implemented in three dimensions. However, it is often assumed that motion along a given coordinate is uncoupled from motion along the other coordinates. If we assume that the components in \mathbf{x}_k are grouped according to their coordinate axes (i.e., x , y , or z), the matrices $F_k^{(\gamma)}$ and $Q_k^{(\gamma)}$ then assume a block diagonal form,

$$F_k^{(\gamma)} = \begin{bmatrix} \bar{F}_k^{(\gamma)} & 0 & 0 \\ 0 & \bar{F}_k^{(\gamma)} & 0 \\ 0 & 0 & \bar{F}_k^{(\gamma)} \end{bmatrix}, \quad Q_k^{(\gamma)} = \begin{bmatrix} \bar{Q}_k^{(\gamma)} & 0 & 0 \\ 0 & \bar{Q}_k^{(\gamma)} & 0 \\ 0 & 0 & \bar{Q}_k^{(\gamma)} \end{bmatrix}. \quad (5.2)$$

In this case, it suffices to specify the matrices $\bar{F}_k^{(\gamma)}$ and $\bar{Q}_k^{(\gamma)}$ for a single coordinate.

5.1.1 Constant velocity model

The constant velocity (CV) model of our IMM-EKF is intended to represent the dynamics of a nonmaneuvering aircraft. Target acceleration is modeled as zero-mean white Gaussian noise. The components of the state vector are

$$\mathbf{x}_k^{(CV)} = [p_{x,k} \ v_{x,k} \ p_{y,k} \ v_{y,k} \ p_{z,k} \ v_{z,k}]', \quad (5.3)$$

where $p_{x,k}$ and $v_{x,k}$ denote position and velocity, respectively, along the x -axis at scan k . The components for the y and z axes are defined in a similar manner. The state matrix and noise covariance are [64]

$$\bar{F}_k^{(CV)} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}, \quad \bar{Q}_k^{(CV)} = \begin{bmatrix} \Delta t^3/3 & \Delta t^2/2 \\ \Delta t^2/2 & \Delta t \end{bmatrix} q^{(CV)}, \quad (5.4)$$

where Δt is the time between consecutive scans and $q^{(CV)}$ is a tuning parameter that controls the strength of the noise for the CV model. The value for $q^{(CV)}$ is often determined empirically, on an application-specific basis. We will discuss this point further in Chapter 7.

5.1.2 Constant acceleration model

The constant acceleration (CA) model of our IMM-EKF is intended to represent the substantial, but transient, accelerations that are present at the beginning and end of maneuvers (e.g., the transition from constant-velocity flight to a coordinated turn). Target acceleration is modeled as Brownian motion. The components of the state vector \mathbf{x}_k , in this case, are

$$\mathbf{x}_k^{(CA)} = [p_{x,k} \ v_{x,k} \ a_{x,k} \ p_{y,k} \ v_{y,k} \ a_{y,k} \ p_{z,k} \ v_{z,k} \ a_{z,k}]', \quad (5.5)$$

where $a_{x,k}$ denotes acceleration along the x -axis at scan k , and the other components are as defined in (5.3). The state matrix and noise covariance are [64]

$$\bar{F}_k^{(CA)} = \begin{bmatrix} 1 & \Delta t & \Delta t^2/2 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix}, \quad \bar{Q}_k^{(CA)} = \begin{bmatrix} \Delta t^5/20 & \Delta t^4/8 & \Delta t^3/6 \\ \Delta t^4/8 & \Delta t^3/3 & \Delta t^2/2 \\ \Delta t^3/6 & \Delta t^2/2 & \Delta t \end{bmatrix} q^{(CA)}, \quad (5.6)$$

where $q^{(CA)}$ is a tuning parameter that controls the strength of the noise for the CA model. Similar to $q^{(CV)}$, the value for $q^{(CA)}$ is also best determined empirically.

5.1.3 Coordinated turn model

The CV and CA models, which we have presented, both assume that the motion along each coordinate is uncoupled from that of the other coordinates. This leads to block diagonal matrices of the form shown in (5.2). While this is a convenient assumption,

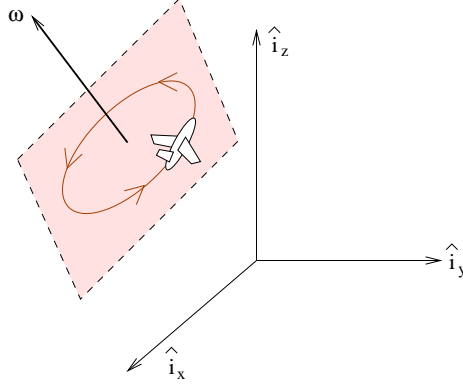


Figure 5.1: Example of IMM coordinated turn maneuver.

actual target maneuvers produce motion that is highly correlated across the coordinate directions [64]. The coordinated turn (CT) model attempts to remedy this shortcoming.

To begin, we note that the name “coordinated turn” is actually a bit of a misnomer, although its use is prevalent in the EKF literature. During a real coordinated turn, what is actually being “coordinated” are the various control surfaces of the airplane in order to maintain a certain angular orientation with respect to the velocity vector. Because delay and Doppler measurements provide limited information about target orientation, EKF-based filters have no real way of monitoring whether or not the pilot is actually coordinating a turn. In the next chapter, we will demonstrate how RCS can be used to define an aerodynamically correct CT model. For extended Kalman filters, what is actually meant by a coordinated turn model is a *constant-speed circular turn* model. An example of this sort of maneuver is given in Figure 5.1. In the figure, $(\hat{i}_x, \hat{i}_y, \hat{i}_z)$ are unit vectors directed along the coordinate axes that the filter is using, and ω is the angular velocity vector. The *turn rate* of the maneuver is simply the magnitude of the angular velocity, $\omega \triangleq \|\omega\|$. This example is meant to illustrate that, while the maneuver does occur in a plane, it is not necessary for that plane to be horizontal (i.e., parallel to the $\hat{i}_x\hat{i}_y$ -plane). Therefore, if the maneuver plane is vertical, the EKF-based CT model can actually be used to represent climbs and descents as well.

We now develop an expression for $\bar{F}_k^{(CT)}$. While the definition of this state matrix can be found throughout the tracking literature, details concerning its derivation are seldom provided. For this reason, we will summarize how $\bar{F}_k^{(CT)}$ is determined. To begin, in a coordinate system whose xy -plane coincides with the maneuver plane, velocity and acceleration along a circular path can be expressed as

$$\tilde{\mathbf{v}}_k = v \begin{bmatrix} -\sin(\omega k \Delta t) \\ \cos(\omega k \Delta t) \\ 0 \end{bmatrix}, \quad \tilde{\mathbf{a}}_k = -v\omega \begin{bmatrix} \cos(\omega k \Delta t) \\ \sin(\omega k \Delta t) \\ 0 \end{bmatrix}, \quad (5.7)$$

where v is the target’s constant speed, $v = \|\tilde{\mathbf{v}}_k\|$, and the tildes are used to indicate that $\tilde{\mathbf{v}}_k$ and $\tilde{\mathbf{a}}_k$ are not expressed in the $(\hat{i}_x, \hat{i}_y, \hat{i}_z)$ coordinate system. Define \mathcal{O} as the

orthogonal transformation that aligns the maneuver coordinates with the $(\hat{i}_x, \hat{i}_y, \hat{i}_z)$ system. Then, if \mathbf{v}_k and \mathbf{a}_k are the velocity and acceleration vectors in filter coordinates, we have

$$\begin{aligned}
\mathbf{v}_{k+1} &= \mathcal{O} \tilde{\mathbf{v}}_{k+1} \\
&= \mathcal{O} \cdot v \begin{bmatrix} -\sin(\omega(k+1)\Delta t) \\ \cos(\omega(k+1)\Delta t) \\ 0 \end{bmatrix} \\
&= \mathcal{O} \cdot v \begin{bmatrix} -\sin(\omega k \Delta t) \cos(\omega \Delta t) - \cos(\omega k \Delta t) \sin(\omega \Delta t) \\ \cos(\omega k \Delta t) \cos(\omega \Delta t) - \sin(\omega k \Delta t) \sin(\omega \Delta t) \\ 0 \end{bmatrix} \\
&= \mathcal{O} \cdot \left(\cos(\omega \Delta t) \tilde{\mathbf{v}}_k + \frac{\sin(\omega \Delta t)}{\omega} \tilde{\mathbf{a}}_k \right) \\
&= \cos(\omega \Delta t) \mathbf{v}_k + \frac{\sin(\omega \Delta t)}{\omega} \mathbf{a}_k. \tag{5.8}
\end{aligned}$$

The same approach can be used to find expressions for \mathbf{p}_{k+1} and \mathbf{a}_{k+1} in terms of \mathbf{p}_k , \mathbf{v}_k , and \mathbf{a}_k (where \mathbf{p}_k is the position vector expressed in filter coordinates). Combining these expressions yields the desired result,

$$\bar{F}_k^{(CT)}(\omega) = \begin{bmatrix} 1 & \sin(\omega \Delta t)/\omega & (1 - \cos(\omega \Delta t))/\omega^2 \\ 0 & \cos(\omega \Delta t) & \sin(\omega \Delta t)/\omega \\ 0 & -\omega \sin(\omega \Delta t) & \cos(\omega \Delta t) \end{bmatrix}. \tag{5.9}$$

Note that (5.8) corresponds to the second row in (5.9). Furthermore, while the overall state matrix $F_k^{(CT)}$ is still block diagonal, the submatrices $\bar{F}_k^{(CT)}(\omega)$ are now coupled through ω . This is precisely what we wished to accomplish.

In order to calculate $\bar{F}_k^{(CT)}(\omega)$, we must first estimate ω . For a planar circular turn, the turn-rate is simply $\omega = \|\mathbf{a}_k\|/v$. In general, a target is unlikely to execute a perfectly circular turn. As such, it is better to estimate ω each scan by first estimating the angular velocity vector,

$$\hat{\omega}_k = \frac{\mathbf{v}_k \times \mathbf{a}_k}{\mathbf{v}_k \cdot \mathbf{v}_k} \Rightarrow \hat{\omega}_k = \frac{\|\mathbf{v}_k \times \mathbf{a}_k\|}{v^2}. \tag{5.10}$$

To complete the specification of our CT model, we note that the components of $\mathbf{x}_k^{(CT)}$ are identical to those of $\mathbf{x}_k^{(CA)}$ given in (5.5). In addition, as a third-order model, $\bar{Q}_k^{(CT)}$ has the same form as $\bar{Q}_k^{(CA)}$ from (5.6). The only difference is the scaling terms, $q^{(CT)}$ and $q^{(CA)}$. In general, $q^{(CT)}$ will be much smaller than $q^{(CA)}$ because the acceleration causing the coordinated turn has already been incorporated in $\bar{F}_k^{(CT)}(\omega)$. Finally, because $\lim_{\omega \rightarrow 0} \bar{F}_k^{(CT)}(\omega) = \bar{F}_k^{(CA)}$, the CT model can also function as a constant acceleration model for a target that is not turning.

5.2 Measurement Model for the IMM-EKF

In this section, we present the measurement model for our IMM-EKF. The measurement equation provided by the jump Markov framework is

$$\mathbf{z}_k = h_k^{(\gamma)}(\mathbf{x}_k) + \mathbf{w}_k. \quad (5.11)$$

Therefore, we must specify $h_k^{(\gamma)}$ for each model and the statistics of the measurement noise sequence $\{\mathbf{w}_k\}$. Before proceeding, a comment must be made about the use of multiple transmitters in our application. As discussed in Chapter 2, the passive radar scenario is inherently bistatic (or multistatic, if multiple transmitters are used). For our simulations, we will assume that the scan times for each transmitter are interleaved (i.e., that the receiver processes one FM radio signal at a time). Because FM radio stations can be modeled as “always on,” we are free to process the scattered signals from these illuminators at any time and in any sequence. Furthermore, if multiple radio signals are received simultaneously (using a multichannel receiver), back-to-back measurement updates are allowed within the recursive Bayesian framework. All this is to say that it suffices to consider the measurement update for a single transmitter at a time. In this case, we are left with the following single-target measurement vector,

$$\mathbf{z}_k = [\tau_k \ d_k]^T, \quad (5.12)$$

where τ_k and d_k are the delay and Doppler shift measurements, respectively, received at scan k .¹ Because neither of these quantities (to a good approximation) is a function of target acceleration, the fact that $\mathbf{x}_k^{(CA)}$ and $\mathbf{x}_k^{(CT)}$ include \mathbf{a}_k is inconsequential. Therefore, the measurement function $h_k^{(\gamma)}$ will have the same form for each γ .

We begin by specifying $h_k^{(\gamma)}$. In this section, we consider tracking a single target in a clutter-free environment. The issue of measurements of uncertain origin and data association will be discussed in the next section. If we denote the position and velocity vectors of the target at scan k as \mathbf{p}_k and \mathbf{v}_k , respectively, then $h_k^{(\gamma)}$ for each of the IMM models is defined implicitly by

$$\tau_k = \frac{\|\mathbf{p}_k - \mathbf{p}_{t_k}\| + \|\mathbf{p}_k - \mathbf{p}_r\|}{c}, \quad (5.13)$$

$$d_k = \frac{\mathbf{v}_k \cdot \hat{\mathbf{n}}_{t_k}(\mathbf{p}_k)}{\lambda_{t_k}}, \quad (5.14)$$

where \mathbf{p}_{t_k} is the location of the transmitter whose signal is being used during scan k , λ_{t_k} is its wavelength, \mathbf{p}_r is the location of the receiver, and c is the speed of light. Note that t_k maps the scan index into a transmitter index. Thus, if N_{tx} is the total number of transmitters used by our system, $t_k \in \{1, \dots, N_{tx}\}$ for each k . In (5.14), $\hat{\mathbf{n}}_{t_k}(\mathbf{p}_k)$ is the inward normal at \mathbf{p}_k of the ellipsoid with foci $(\mathbf{p}_{t_k}, \mathbf{p}_r)$ and major axis length $\tau_k c$.

¹We assume that the beamwidth of the receiving antenna is too wide to supply useful angle of arrival data.

This normal can be evaluated as

$$\hat{n}_{t_k}(\mathbf{p}_k) = \frac{\mathbf{p}_{t_k} - \mathbf{p}_k}{\|\mathbf{p}_{t_k} - \mathbf{p}_k\|} + \frac{\mathbf{p}_r - \mathbf{p}_k}{\|\mathbf{p}_r - \mathbf{p}_k\|}. \quad (5.15)$$

The equations for Doppler shift, (5.14)–(5.15), take this somewhat complicated form due to the bistatic nature of our application. Because Doppler shift is proportionate to range rate, the velocity vector \mathbf{v}_k must be projected onto the normal of the contour of constant range. In the bistatic scenario, this contour is an ellipsoid. As a check of (5.15), consider a monostatic radar. In this case, $\mathbf{p}_{t_k} = \mathbf{p}_r$, yielding $\hat{n}_{t_k}(\mathbf{p}_k) = -2\hat{i}_{\mathbf{p}_k - \mathbf{p}_r}$, where $\hat{i}_{\mathbf{p}_k - \mathbf{p}_r}$ is the unit normal directed (radially outward) from the receiver towards the target. Because $\mathbf{v}_k \cdot \hat{i}_{\mathbf{p}_k - \mathbf{p}_r}$ is the range rate, Equation (5.14) then yields $d_k = -2\dot{R}_k/\lambda_{t_k}$, the familiar relationship for Doppler shift.

From (5.13)–(5.15), we see that the relationship between \mathbf{x}_k and \mathbf{z}_k for our application is a nonlinear one. This is why it is necessary to use extended Kalman filters for our benchmark system. As such, we will need to specify the Jacobian matrix

$$\tilde{H}_k^{(\gamma)} = \left. \frac{\partial h_k^{(\gamma)}(\mathbf{x})}{\partial \mathbf{x}} \right|_{\mathbf{x}=\hat{\mathbf{x}}_{k|k-1}^{(\gamma)}}. \quad (5.16)$$

Using the definitions of delay and Doppler shift from (5.13)–(5.15), we see that $\tilde{H}_k^{(\gamma)}$ has the following form,

$$\tilde{H}_k^{(\gamma)} = \left[\begin{array}{ccccccccc} \frac{\partial \tau_k}{\partial p_x} & 0 & 0 & \frac{\partial \tau_k}{\partial p_y} & 0 & 0 & \frac{\partial \tau_k}{\partial p_z} & 0 & 0 \\ \frac{\partial d_k}{\partial p_x} & \frac{\partial d_k}{\partial v_x} & 0 & \frac{\partial d_k}{\partial p_y} & \frac{\partial d_k}{\partial v_y} & 0 & \frac{\partial d_k}{\partial p_z} & \frac{\partial d_k}{\partial v_z} & 0 \end{array} \right] \bigg|_{\mathbf{x}=\hat{\mathbf{x}}_{k|k-1}^{(\gamma)}}, \quad (5.17)$$

where the nine-dimensional state vector of the CA and CT models has been assumed (see Equation (5.5)). For the CV model's six-dimensional state vector, $\tilde{H}_k^{(CV)}$ would be the same as (5.17), but with every third column eliminated. With some algebra, the following equations for the partial derivatives can be obtained,

$$\frac{\partial \tau_k}{\partial \mathbf{p}} = -\frac{\hat{n}_{t_k}(\hat{\mathbf{p}}_k)}{c}, \quad (5.18)$$

$$\frac{\partial d_k}{\partial \mathbf{v}} = \frac{\hat{n}_{t_k}(\hat{\mathbf{p}}_k)}{\lambda_{t_k}}, \quad (5.19)$$

$$\begin{aligned} \frac{\partial d_k}{\partial \mathbf{p}} = \frac{1}{\lambda_{t_k}} & \left(\frac{(\hat{\mathbf{p}}_k - \mathbf{p}_{t_k})(\hat{\mathbf{p}}_k - \mathbf{p}_{t_k})'}{\|\hat{\mathbf{p}}_k - \mathbf{p}_{t_k}\|^3} \right. \\ & \left. + \frac{(\hat{\mathbf{p}}_k - \mathbf{p}_r)(\hat{\mathbf{p}}_k - \mathbf{p}_r)'}{\|\hat{\mathbf{p}}_k - \mathbf{p}_r\|^3} - b_k I_3 \right) \cdot \hat{\mathbf{v}}_k, \end{aligned} \quad (5.20)$$

where I_3 is the 3×3 identity matrix, and the vectors $\hat{\mathbf{p}}_k$ and $\hat{\mathbf{v}}_k$ are used to denote the position and velocity estimates from $\hat{\mathbf{x}}_{k|k-1}^{(\gamma)}$. The coefficient b_k is defined as

$$b_k = \frac{\|\hat{\mathbf{p}}_k - \mathbf{p}_{t_k}\| + \|\hat{\mathbf{p}}_k - \mathbf{p}_r\|}{\|\hat{\mathbf{p}}_k - \mathbf{p}_{t_k}\| \|\hat{\mathbf{p}}_k - \mathbf{p}_r\|}. \quad (5.21)$$

This completes our specification of $h_k^{(\gamma)}$ for the IMM models.

A description of the measurement noise sequence is much more straightforward. We assume that $\{\mathbf{w}_k\}$ is a zero-mean white Gaussian process, $\mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, R_k)$. We also assume that the component noises for a given scan are independent, which yields

$$R_k = \begin{bmatrix} \sigma_{r_k}^2 & 0 \\ 0 & \sigma_{d_k}^2 \end{bmatrix}. \quad (5.22)$$

For the experimental results presented in Chapter 7, $\sigma_{r_k}^2$ and $\sigma_{d_k}^2$ will not vary with k .

5.3 Data Association for the IMM-EKF

In the previous section, we defined the IMM-EKF measurement model assuming that only a single target was being tracked in a clutter-free environment with zero probability of miss. In this simple case, \mathbf{z}_k contains exactly one pair of delay-Doppler measurements per scan. Furthermore, this single pair of measurements is guaranteed to have originated at the target. A more realistic scenario would include multiple targets, clutter returns (i.e., false alarms), and a nonzero probability that one or more of the targets would go undetected in any given scan. In this case, \mathbf{z}_k could contain more or less than a single pair of delay-Doppler features.

Data association is the process of dealing with the uncertain origin of each measurement in this case. It assigns measurements to target tracks so that the Kalman update can occur. Methods of data association can be grouped into two broad categories: those that perform hard assignments and those that perform soft assignments [64]. *Hard* assignment techniques attempt to determine the single “best” association of measurements to targets for use in the Kalman update stage. This optimal assignment is determined according to some suitably chosen criterion (e.g., maximum likelihood). *Soft* assignment techniques use all measurements to update all tracks, but the influence of each measurement varies with the probability that it was produced by the target under consideration. For our IMM-EKF system, we will use *joint probabilistic data association* (JPDA), a soft assignment algorithm that we will introduce in the following sections.

Although we will consider multitarget tracking at the end of this chapter, it is helpful to begin our discussion with the simpler case of tracking a single target in the presence of clutter and missed detections. Variants of the following results can be found in many sources [32, 64–67].

5.3.1 Probabilistic data association

In this section, we introduce the mathematical framework needed to perform probabilistic data association (PDA) by considering the case of tracking a single target in the presence of clutter and missed detections. In order to simplify notation, we will suppress the superscript notation used earlier to indicate which of the IMM models was

being considered. However, because each model corresponds to a self-contained EKF, the following results can be taken to apply to each filter within our IMM-EKF.

We begin with several definitions. Define V_k as the volume of the measurement space searched during scan k for target detections. (Practical constraints prevent the entire measurement space from being considered.) This subset of the measurement space is usually centered upon $h_k(\hat{\mathbf{x}}_{k|k-1})$ and referred to as the *gate*. We assume that the number of false alarms in each scan is Poisson-distributed with mean βV_k , where β is the density of false alarms per unit volume. We follow common practice and assume that, conditioned upon the number of false alarms, their locations are distributed uniformly across the gate volume. For each scan, the probability that the measurement for a target being tracked is contained within the gate is denoted P_G . The probability that the measurement is detected, conditioned on it being within the gate, is denoted P_D . In actuality, the probability of detection is a function of the round-trip range from transmitter to target to receiver (see Equation (2.2)), but it is common to simply assume that P_D is constant. If we denote the number of detections in scan k as m_k , the EKF measurement vector is then

$$\mathbf{z}_k = \left[\left(\tau_k^{(1)} \ d_k^{(1)} \right) \dots \left(\tau_k^{(m_k)} \ d_k^{(m_k)} \right) \right]'. \quad (5.23)$$

We define the vector of components from a single detection as

$$\mathbf{z}_k^{(j)} = \left[\tau_k^{(j)} \ d_k^{(j)} \right]', \quad j \in \{1, \dots, m_k\}, \quad (5.24)$$

and, where it does not lead to confusion, we will refer to $\mathbf{z}_k^{(j)}$ as a single measurement (even though it actually contains a delay-Doppler pair).

We will use $\Gamma_k^{(j)}$ to denote a possible association of measurements from \mathbf{z}_k to targets. In the general (multitarget) scenario, each association hypothesis would contain three pieces of information: (1) the number of false alarms, (2) the number of targets detected, and (3) a measurement index for each target (hypothesized) as having been detected. The third item identifies which measurement goes with which target track. In the single target scenario, this description simplifies because the number of detections is either one or zero. Thus, there are $m_k + 1$ possible association hypotheses per scan. If the target is detected, the correct measurement can be any one of the m_k in \mathbf{z}_k . If the target is not detected, all measurements are false. When considering the single target scenario, the superscript of $\Gamma_k^{(j)}$ will have the following meaning. For $j = 1, \dots, m_k$, $\Gamma_k^{(j)}$ will denote the hypothesis that $\mathbf{z}_k^{(j)}$ is the “correct” measurement, and the other $m_k - 1$ are false. $\Gamma_k^{(0)}$ will denote the hypothesis that the target went undetected, and thus all m_k measurements are false.² The goal of this section is to find expressions for the conditional probabilities $\Pr(\Gamma_k^{(j)} | \mathbf{z}_{1:k}), j = 0, 1, \dots, m_k$.

²When the superscript of the association hypothesis is a generic index with no implied meaning, we will use l instead of j (e.g., $\Gamma_k^{(l)}$).

Using Bayes' rule, we have

$$\begin{aligned} \Pr(\Gamma_k^{(j)} | \mathbf{z}_{1:k}) &= \Pr(\Gamma_k^{(j)} | \mathbf{z}_k, \mathbf{z}_{1:k-1}) \\ &\propto p(\mathbf{z}_k | \Gamma_k^{(j)}, \mathbf{z}_{1:k-1}) \Pr(\Gamma_k^{(j)} | \mathbf{z}_{1:k-1}). \end{aligned} \quad (5.25)$$

Conditioned upon the association event $\Gamma_k^{(j)}$, the first term in (5.25) is simply the product of the measurement likelihoods for each component $\mathbf{z}_k^{(j)}$. To simplify notation, define $\mathbf{y}_k^{(j)}$ as the innovation for the j^{th} measurement,

$$\mathbf{y}_k^{(j)} \triangleq \mathbf{z}_k^{(j)} - h_k(\hat{\mathbf{x}}_{k|k-1}). \quad (5.26)$$

Then, because the measurement noise is additive Gaussian and the false alarms are uniformly distributed across the gate,

$$p(\mathbf{z}_k | \Gamma_k^{(j)}, \mathbf{z}_{1:k-1}) = \begin{cases} V_k^{-(m_k-1)} P_G^{-1} \mathcal{N}(\mathbf{y}_k^{(j)}; \mathbf{0}, S_k) & j = 1, \dots, m_k \\ V_k^{-m_k} & j = 0, \end{cases} \quad (5.27)$$

where the term P_G^{-1} accounts for the fact that the Gaussian density is truncated by the gate.³ The second term in (5.25), the association probability, can be expressed as

$$\Pr(\Gamma_k^{(j)} | \mathbf{z}_{1:k-1}) = \Pr(\Gamma_k^{(j)}) \quad (5.28)$$

$$= \begin{cases} \frac{1}{m_k} P_D P_G \mu_F(m_k - 1) & j = 1, \dots, m_k \\ (1 - P_D P_G) \mu_F(m_k) & j = 0, \end{cases} \quad (5.29)$$

where $P_D P_G$ is the probability that the correct measurement is in \mathbf{z}_k (i.e., that the measurement satisfies the gate and is detected), and μ_F is the Poisson mass function for the false alarm count. Note that $\Gamma_k^{(j)}$ is independent of $\mathbf{z}_{1:k-1}$ because the number of false alarms and detections are modeled as independent from scan to scan. (This assumes that both β and P_D are known; otherwise, (5.28) would not hold.)

We now arrive at the desired result by substituting (5.27) and (5.29) back into (5.25) and simplifying,

$$\Pr(\Gamma_k^{(j)} | \mathbf{z}_{1:k}) = \frac{1}{C} \times \begin{cases} \frac{V_k P_D \mu_F(m_k-1)}{m_k} \mathcal{N}(\mathbf{y}_k^{(j)}; \mathbf{0}, S_k) & j = 1, \dots, m_k \\ (1 - P_D P_G) \mu_F(m_k) & j = 0, \end{cases} \quad (5.30)$$

where we absorbed the term $V_k^{-m_k}$ into the normalization constant C . The expression in (5.30) can be simplified even further by replacing $\mu_F(m_k - 1)$ and $\mu_F(m_k)$ with their actual probabilities. Because μ_F is a Poisson distribution with mean βV_k , we have

$$\mu_F(m_k) = \frac{(\beta V_k)^{m_k} e^{-\beta V_k}}{m_k!}. \quad (5.31)$$

³Typically, the gate is chosen such that $P_G \approx 1$ (i.e., the chance that the true measurement is outside the gate is negligible).

Substituting (5.31) into (5.30) yields

$$\Pr(\Gamma_k^{(j)} | \mathbf{z}_{1:k}) = \frac{1}{\tilde{C}} \times \begin{cases} P_D \mathcal{N}(\mathbf{y}_k^{(j)}; \mathbf{0}, S_k) & j = 1, \dots, m_k \\ (1 - P_D P_G) \beta & j = 0, \end{cases} \quad (5.32)$$

where we have absorbed terms common to both cases into the normalization constant. Because the events $\Gamma_k^{(j)}$ for $j = 0, 1, \dots, m_k$ are mutually exclusive and exhaustive, \tilde{C} can be calculated as

$$\tilde{C} = (1 - P_D P_G) \beta + P_D \sum_{j=1}^{m_k} \mathcal{N}(\mathbf{y}_k^{(j)}; \mathbf{0}, S_k). \quad (5.33)$$

Equation (5.32) is the result that we wished to derive. In the next section, we will use it to implement the probabilistic data association filter.

5.3.2 PDA filter

In the previous section, we considered the case of tracking a single target in clutter with $P_D < 1$. The main result was an expression for $\Pr(\Gamma_k^{(j)} | \mathbf{z}_{1:k})$, the conditional probability that the j^{th} measurement is correct (or that all measurements are false if $j = 0$). In this section, we will specify how these probabilities can be used to perform recursive filtering in the presence of clutter and missed detections. We begin by considering the task of performing data association across multiple scans.

Given a sequence of scans $\mathbf{z}_{1:k}$, define $\Gamma_{1:k}^{(l)}$ as the multiscan association event

$$\Gamma_{1:k}^{(l)} \triangleq \left\{ \Gamma_1^{(j_1)}, \Gamma_1^{(j_2)}, \dots, \Gamma_k^{(j_k)} \right\}, \quad (5.34)$$

where each index j_i satisfies $j_i \in [0, m_i]$. Note that $\Gamma_{1:k}^{(l)}$ is simply the concatenation of k single-scan association events. The total number of associations through k scans is then

$$M_k \triangleq \prod_{i=1}^k (m_i + 1), \quad (5.35)$$

which grows exponentially with k . Because the association events are mutually exclusive and exhaustive, the posterior distribution of \mathbf{x}_k can be expressed as

$$p(\mathbf{x}_k | \mathbf{z}_{1:k}) = \sum_{l=1}^{M_k} p(\mathbf{x}_k | \Gamma_{1:k}^{(l)}, \mathbf{z}_{1:k}) \Pr(\Gamma_{1:k}^{(l)} | \mathbf{z}_{1:k}). \quad (5.36)$$

As discussed in Section 3.2.1, we see that the presence of clutter and missed detections is likely to result in a decidedly non-Gaussian posterior. The conditional mean then takes the form

$$\mathbb{E}[\mathbf{x}_k | \mathbf{z}_{1:k}] = \sum_{l=1}^{M_k} \mathbb{E}[\mathbf{x}_k | \Gamma_{1:k}^{(l)}, \mathbf{z}_{1:k}] \Pr(\Gamma_{1:k}^{(l)} | \mathbf{z}_{1:k}). \quad (5.37)$$

Thus, in the presence of clutter and missed detections, the conditional mean $E[\mathbf{x}_k | \mathbf{z}_{1:k}]$ is a weighted sum of the expectations $E[\mathbf{x}_k | \Gamma_{1:k}^{(l)}, \mathbf{z}_{1:k}]$ taken over *every* possible multiscan association hypothesis. If the system model, when conditioned on $\Gamma_{1:k}^{(l)}$, is linear Gaussian, Equation (5.37) implies that M_k Kalman filters are needed to compute $E[\mathbf{x}_k | \mathbf{z}_{1:k}]$! Clearly, this is unacceptable for any real-time application. Instead, we will have to consider suboptimal solutions.

There are two data association techniques that are widely used in modern radar systems. The first, *multiple hypothesis tracking* (MHT), approximates (5.37) directly by maintaining the filtered outputs from the top N association hypotheses [67]. Of course, with each new measurement vector \mathbf{z}_k , the set of N hypotheses expands to $(m_k + 1)N$ possibilities, which must then be reduced back to a manageable level. The MHT algorithm accomplishes this by combining the filtered outputs of hypotheses that are sufficiently similar and discarding those associations whose conditional probabilities become too small. The second popular approach, the *probabilistic data association filter* (PDAF), “re-Gaussianizes” its estimate of the posterior density after each scan [68]. We present its key equations next.

The PDA filter can be motivated as follows. Assume that the posterior given $\mathbf{z}_{1:k-1}$ is actually Gaussian,

$$p(\mathbf{x}_{k-1} | \mathbf{z}_{1:k-1}) = \mathcal{N}(\mathbf{x}_{k-1}; \hat{\mathbf{x}}_{k-1|k-1}, \hat{\Sigma}_{k-1|k-1}), \quad (5.38)$$

for some choice of $\hat{\mathbf{x}}_{k-1|k-1}$ and $\hat{\Sigma}_{k-1|k-1}$. Note that, unlike (5.36), there is no conditioning on association events in (5.38). Next, express the posterior given $\mathbf{z}_{1:k}$ as

$$p(\mathbf{x}_k | \mathbf{z}_{1:k}) = \sum_{j=0}^{m_k} p(\mathbf{x}_k | \Gamma_k^{(j)}, \mathbf{z}_{1:k}) \Pr(\Gamma_k^{(j)} | \mathbf{z}_{1:k}). \quad (5.39)$$

Then, if the clutter-free system was linear Gaussian, we would have

$$p(\mathbf{x}_k | \Gamma_k^{(j)}, \mathbf{z}_{1:k}) = \mathcal{N}(\mathbf{x}_k; \hat{\mathbf{x}}_{k|k}^{(j)}, \hat{\Sigma}_{k|k}^{(j)}), \quad (5.40)$$

where $\hat{\mathbf{x}}_{k|k}^{(j)}$ and $\hat{\Sigma}_{k|k}^{(j)}$ denote the output of a single Kalman iteration using $\mathbf{z}_k^{(j)}$ as measurement and $\mathcal{N}(\hat{\mathbf{x}}_{k-1|k-1}, \hat{\Sigma}_{k-1|k-1})$ as prior.⁴ Plugging (5.40) into (5.39), the probabilistic data association filter makes the approximation

$$p(\mathbf{x}_k | \mathbf{z}_{1:k}) = \sum_{j=0}^{m_k} \mathcal{N}(\mathbf{x}_k; \hat{\mathbf{x}}_{k|k}^{(j)}, \hat{\Sigma}_{k|k}^{(j)}) \Pr(\Gamma_k^{(j)} | \mathbf{z}_{1:k}) \quad (5.41)$$

$$\approx \mathcal{N}(\mathbf{x}_k; \hat{\mathbf{x}}_{k|k}, \hat{\Sigma}_{k|k}), \quad (5.42)$$

where $\hat{\mathbf{x}}_{k|k}$ and $\hat{\Sigma}_{k|k}$ are the mean and covariance of the Gaussian mixture distribution in (5.41). In words, at each scan, the PDA algorithm approximates the previous posterior $p(\mathbf{x}_{k-1} | \mathbf{z}_{1:k-1})$ as Gaussian so that a Kalman filter (or an EKF) can be used to compute $p(\mathbf{x}_k | \Gamma_k^{(j)}, \mathbf{z}_{1:k})$ for $j = 0, 1, \dots, m_k$. The resulting mixture distribution is

⁴Note that the superscript notation does not refer to IMM models in this section.

then “re-Gaussianized” for the next scan.

Because the individual densities $p(\mathbf{x}_k | \Gamma_k^{(j)}, \mathbf{z}_{1:k})$ are never actually required, it suffices to compute $\hat{\mathbf{x}}_{k|k}$ and $\hat{\Sigma}_{k|k}$ directly. (Note, these will not, in general, be equal to the true conditional mean and covariance.) The equation for the PDAF mean is (see [69] for details)

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + K_k \mathbf{y}_k \quad \text{where} \quad \mathbf{y}_k = \sum_{j=1}^{m_k} \mathbf{y}_k^{(j)} \Pr(\Gamma_k^{(j)} | \mathbf{z}_{1:k}) \quad (5.43)$$

and $\Pr(\Gamma_k^{(j)} | \mathbf{z}_{1:k})$ is given by (5.32). The equation for the PDAF covariance is

$$\hat{\Sigma}_{k|k} = \hat{\Sigma}_{k|k}^{(0)} + K_k \left(\sum_{j=1}^{m_k} \mathbf{y}_k^{(j)} \mathbf{y}_k^{(j)'} \Pr(\Gamma_k^{(j)} | \mathbf{z}_{1:k}) - \mathbf{y}_k \mathbf{y}_k' \right) K_k' \quad (5.44)$$

where $\hat{\Sigma}_{k|k}^{(0)}$ is the covariance if $m_k = 1$,

$$\hat{\Sigma}_{k|k}^{(0)} \triangleq (1 - P_D P_G) \hat{\Sigma}_{k|k-1} + P_D P_G \hat{\Sigma}_{k|k}^* \quad (5.45)$$

and $\hat{\Sigma}_{k|k}^*$ is the standard Kalman gain matrix. Inspection of Equations (5.43) and (5.44) reveal that the PDAF can be implemented using a single Kalman filter (or EKF), with (5.43)–(5.45) replacing the standard Kalman update. As such, the PDA filter is computationally much less expensive than the MHT algorithm.

Finally, we note that it is straightforward to implement the PDA algorithm within the IMM framework. Only two things need to be changed. First, each of the IMM filters will use (5.43)–(5.45) instead of the standard Kalman (or EKF) update equations. Second, each model likelihood $\Lambda_k^{(\gamma)}$ must now be averaged over the possible data associations. This average takes the simple form $\Lambda_k^{(\gamma)} = \sum_{j=0}^{m_k} \Pr(\Gamma_k^{(j,\gamma)} | \mathbf{z}_{1:k})$, where $\Gamma_k^{(j,\gamma)}$ is the event that the j^{th} measurement is taken as correct for the γ^{th} IMM filter. More details can be found in [64].

5.3.3 JPDA filter

In this section, we extend our discussion to the case of multitarget tracking in the presence of clutter and missed detections. Although the PDA filter from the previous section can still be used in this case, it tends to give poor results because its clutter model is no longer valid. Recall, the PDA algorithm assumes that, at most, one measurement from \mathbf{z}_k corresponds to an actual target. Because this true measurement is detected with probability $P_D P_G > 0.7$ (typically), it can be considered as relatively “persistent” from scan to scan. The false alarms, on the other hand, are assumed to be independent from scan to scan, with a uniform distribution across the gate volume. As such, false alarms are unlikely to form persistent sequences across multiple scans (unless the clutter density β is high). When the gates of two or more targets overlap, there will then be *multiple* persistent sources, and this degrades the performance of the PDA filter. *Joint probabilistic data association* (JPDA) overcomes this limitation by

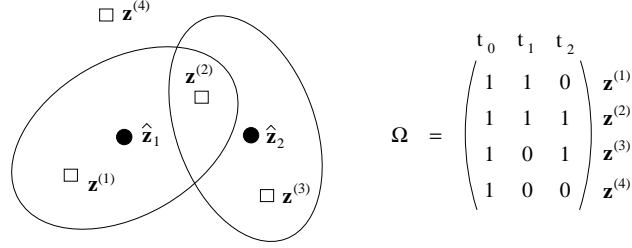


Figure 5.2: Illustration of measurement gates and the corresponding validation matrix.

computing the association probabilities jointly for all target tracks whose gates overlap [70].

The JPDA algorithm relies upon the notion of *feasible* associations. A feasible association of data to targets is one that satisfies two criteria: (1) no measurement can have more than one target as its source, and (2) in any given scan, no target can produce more than one measurement. The actual algorithm has four basic steps: formation of the validation matrix, enumeration of all feasible associations, calculation of association probabilities, and update of the posterior densities.

The first step, formation of the validation matrix, reduces computation in the subsequent steps. In Figure 5.2, a simple example involving two targets is shown. For notational simplicity, the scan subscript k is suppressed. We use $\hat{\mathbf{z}}_1$ and $\hat{\mathbf{z}}_2$ to denote the predicted measurements for targets t_1 and t_2 .⁵ $\{\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(4)}\}$ is the set of delay-Doppler measurements received. The extent of a target's validation gate is chosen so that any measurement falling outside is very unlikely to have been produced by the target. This gating eliminates many feasible associations that would have negligible probabilities. The gates in Figure 5.2 are represented by the two ellipses. The resulting validation matrix Ω is shown at the right. Each row of the matrix corresponds to a different measurement while each column corresponds to a different target. The column t_0 is included to represent the possibility that a measurement is a false alarm. Thus, $\Omega(j, t) = 1$ means that the j^{th} measurement may have originated from target t . The need for a joint approach to data association is highlighted whenever a measurement lies within multiple gates. In Figure 5.2, we see that $\mathbf{z}^{(2)}$ lies within both gates. Data assignment for the two targets must proceed jointly because association of $\mathbf{z}^{(2)}$ with t_1 precludes its association with t_2 , and vice versa.

The second step in the JPDA algorithm requires the enumeration of all feasible associations. Conceptually, a feasible association can be found by changing ones to zeros within Ω until each row contains exactly one nonzero entry and each column contains at most one nonzero entry (except for column t_0 , which may contain any number of ones). If the clutter density is high, it may become impractical to enumerate all feasible associations. In cases such as these, a technique such as Murty's algorithm can be used to return the top N associations [71]. Murty's algorithm offers the significant benefit that the search proceeds in order; namely, the best assignment is determined first, fol-

⁵In this section, the variable t is used for the target index instead of the transmitter index.

lowed by the second best, third best, and so on. Thus, it is possible to terminate the search after the first N feasible associations have been found.

The third step in the JPDA algorithm involves computation of $\Pr(\Gamma_k^{(l)} | \mathbf{z}_{1:k})$ for each feasible association. In this case, $\Gamma_k^{(l)}$ will associate measurements from \mathbf{z}_k with *multiple* targets. An expression for these multitarget association probabilities can be derived in the same way as the single-target probabilities from Section 5.3.1. Therefore, we will simply state the result. Further details can be found in [70]. Defining N_D^l and N_F^l as the number of detections and false alarms, respectively, in the association provided by $\Gamma_k^{(l)}$, it can be shown that

$$\Pr(\Gamma_k^{(l)} | \mathbf{z}_{1:k}) = \frac{(P_D)^{N_D^l} ((1 - P_D P_G) \beta)^{N_F^l}}{C} \prod_{t \in \Gamma_k^{(l)}} \mathcal{N}(\mathbf{y}_{k,t}^{(j_t)}; \mathbf{0}, S_{k,t}), \quad (5.46)$$

where C is a normalization constant and the double subscripts refer to the scan (k) and the target index (t). The product in (5.46) is over all targets that association $\Gamma_k^{(l)}$ specifies as having been detected. The innovation $\mathbf{y}_{k,t}^{(j_t)}$ is formed using $\mathbf{z}_k^{(j_t)}$, the measurement that $\Gamma_k^{(l)}$ associates with target t . Note the similarity between (5.46) and the single-target results in (5.32). Basically, for each target that $\Gamma_k^{(l)}$ specifies as having been detected, there is a factor of $P_D \mathcal{N}(\mathbf{y}_{k,t}^{(j_t)}; \mathbf{0}, S_{k,t})$ in (5.46). For each false alarm in $\Gamma_k^{(l)}$, there is a factor of $(1 - P_D P_G) \beta$.

The fourth and final step of the JPDA algorithm involves the modification of the Kalman update equations. The JPDA update equations are identical to those of the PDAF, (5.43)–(5.45). The only difference is that the joint association probabilities from the previous step are used instead of the single-target probabilities. It is these joint probabilities that allow the JPDA algorithm to handle targets with overlapping gates. Because our benchmark filter must be able to track multiple targets in the presence of clutter and missed detections, we will implement the JPDA algorithm to perform data association in our IMM-EKF tracker.

CHAPTER 6

PARTICLE FILTER-BASED TRACKING AND CLASSIFICATION

In Chapter 4, we presented the sequential importance sampling (SIS) algorithm upon which particle filters are based. We detailed two specific variants: a generic particle filter that used the prior as its proposal distribution (see Section 4.3.1) and an auxiliary particle filter whose characterization variable $\lambda_k^{(i)}$ was drawn from the prior (see Section 4.3.4). In Chapter 7, we will implement the flight model from this chapter using both of these algorithms. Because sampling from the prior $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ is equivalent to driving the state model $f_k(\mathbf{x}_{k-1}, \mathbf{u}_k)$ with a sample from the process noise, it suffices to specify our state and measurement models. We follow the organization of the previous chapter and divide our discussion into sections on the state model, the measurement model, and data association.

6.1 State Model for the Particle Filter

The sequential importance sampling framework provides an impressive amount of flexibility to the system designer. It is a straightforward matter for particle filters to model systems that are neither linear nor Gaussian. As such, we are free to use state models of substantial complexity. With this freedom, it makes sense to adopt a model that is as closely matched to the actual state dynamics as possible. Because we are primarily interested in tracking aircraft, this means finding an aerodynamically valid flight model. This is an important point. Unlike the extended Kalman filter, which adopts a Gaussian model because that is all the algorithm will allow, a particle filter is free to use the *correct* state model, whenever it is known.

For our purposes, an airplane can be modeled as a rigid body with six degrees of freedom [72]. Three degrees are needed to specify its location in space (or rather, the location of a reference point, such as its center of mass). The other three degrees are needed to specify the angular orientation of the aircraft with respect to a fixed reference frame. Its motion is then completely specified by its time-varying location and orien-

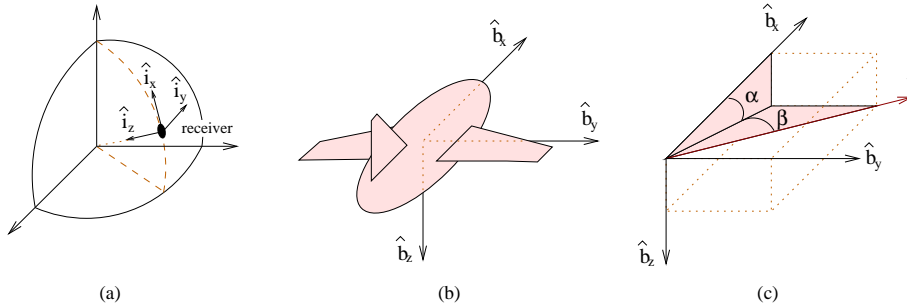


Figure 6.1: (a) Example of a receiver-centered, Earth-fixed (inertial) frame. (b) Example of a body-centered frame. (c) Definition of the angle of attack α and the sideslip angle β .

tation. Although we refrain from specifying the components of the state vector just yet, it is clear that we need a state model that accurately predicts both the translational and rotational motion of an airplane. The state vector is then implicitly defined as the set of variables needed to accomplish this task. Of course, because translational and rotational motions are caused by forces and torques, respectively, our modeling efforts must focus on these. Before we can do so, though, we first introduce the reference frames and aerodynamic angles that we will need in our discussion.

6.1.1 Reference frames and aerodynamic angles

One of the difficulties with rigid-body models is the need to work in different frames. In every dynamics problem, there must be an inertial reference frame [73]. This frame is fixed relative to the stars. In an inertial frame, the net acceleration of an object can be found using Newton's second law, $\mathbf{f} = m\mathbf{a}$, where m is the mass of the object and \mathbf{f} is the sum of all external forces acting upon it. Note that Newton's second law does *not* hold if the reference frame itself is rotating or accelerating. In most problems involving airplane dynamics, the rotation of the Earth (relative to an inertial frame) is negligible, and an Earth-fixed frame can be used in place of an inertial frame.

An *Earth-fixed* frame is simply a reference frame whose origin is fixed either inside or on the surface of the Earth. A natural choice for the origin in an Earth-fixed frame is the location of the radar receiver. Following common practice, we will use a receiver-centered Earth-fixed frame as our inertial frame. The unit vectors in this frame will be denoted as \hat{i}_x , \hat{i}_y , and \hat{i}_z . It is then standard practice to define the Earth-fixed axes such that \hat{i}_x points north, \hat{i}_y points east, and \hat{i}_z points down (towards the center of the Earth). An example of a receiver-centered Earth-fixed frame is shown in Figure 6.1(a).

A second type of reference frame that we will need is a body-centered (or body-fixed) frame. As the name implies, the origin of this frame is fixed to some point within the aircraft, typically its center of mass. The unit vectors in this frame will be denoted as \hat{b}_x , \hat{b}_y , and \hat{b}_z . It is common practice to orient the body axes such that \hat{b}_x points out the nose of the plane, \hat{b}_y points to the pilot's right, and \hat{b}_z points through the plane's

bottom. Furthermore, the rotational movement of the aircraft is usually specified in the body frame. Rotation about the \hat{b}_x , \hat{b}_y , and \hat{b}_z axes is typically referred to as *roll*, *pitch*, and *yaw*, respectively. An example of a body-centered frame is shown in Figure 6.1(b).

Finally, there is one more frame that we need to consider. This frame is referred to as the wind frame. Its origin is at the center of mass of the vehicle, but its x -axis is directed along the velocity vector \mathbf{v} , which generally does not coincide with \hat{b}_x . The z -axis of the wind frame is typically defined to lie in the $\hat{b}_x\hat{b}_z$ -plane. Because the wind frame is not a convenient reference frame in which to work, we will not define unit vectors for it. Instead, we define two angles that can be used to align the wind frame with the body frame. These angles are shown in Figure 6.1(c). The angle of attack is defined as

$$\alpha \triangleq \tan^{-1} (v_z^b / v_x^b), \quad (6.1)$$

where v_x^b denotes the x -component of the velocity vector expressed in the body frame. The sideslip angle is defined as

$$\beta \triangleq \sin^{-1} (v_y^b / v), \quad (6.2)$$

where v is the speed of the aircraft (i.e., $v = \|\mathbf{v}\|$). These angles, which are fundamentally important in determining the forces that act upon an aircraft, are typically referred to as the *aerodynamic angles*.¹

We have specified the three reference frames that we will need to derive our state model. We now discuss how vectors, such as velocity, can be transformed from one frame to another. To avoid confusion, we will use the superscripts b and w to identify vectors that are expressed in the body and wind frames, respectively. Vectors that are expressed in inertial coordinates will be written without a superscript. To begin, the orientation of any frame with respect to another is given by three angles. We are primarily interested in the sequence of right-handed rotations about \hat{i}_z , \hat{i}_y , and \hat{i}_x (in that order) that bring the inertial frame into alignment with the body-centered frame. This is a particular case of *Euler angles*. The necessary rotations about \hat{i}_z , \hat{i}_y , and \hat{i}_x will be referred to as the heading, elevation, and bank angles, respectively, of the aircraft. We follow common practice and denote heading as ψ , elevation as θ , and bank as ϕ . Given these three angles, we can then define the orthogonal transformation

$$\mathcal{O}^{bi} \triangleq \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \phi & \sin \phi \\ 0 & -\sin \phi & \cos \phi \end{bmatrix} \begin{bmatrix} \cos \theta & 0 & -\sin \theta \\ 0 & 1 & 0 \\ \sin \theta & 0 & \cos \theta \end{bmatrix} \begin{bmatrix} \cos \psi & \sin \psi & 0 \\ -\sin \psi & \cos \psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (6.3)$$

$$= \begin{bmatrix} c_\theta c_\psi & c_\theta s_\psi & -s_\theta \\ s_\phi s_\theta c_\psi - c_\phi s_\psi & s_\phi s_\theta s_\psi + c_\phi c_\psi & s_\phi c_\theta \\ c_\phi s_\theta c_\psi + s_\phi s_\psi & c_\phi s_\theta s_\psi - s_\phi c_\psi & c_\phi c_\theta \end{bmatrix}, \quad (6.4)$$

where we used the shorthand c for \cos and s for \sin to make (6.4) more readable. In

¹Because both conventions are widespread, we will use β to represent the clutter density and the sideslip angle. In all cases, the intended meaning should be clear from the context.

each case, the subscript accompanying c or s denotes the argument of the corresponding trigonometric function. The matrix \mathcal{O}^{bi} transforms vectors from the inertial frame into the body frame. Thus, the velocity vector in body-centered coordinates is $\mathbf{v}^b = \mathcal{O}^{bi}\mathbf{v}$. Furthermore, because \mathcal{O}^{bi} is orthogonal, we have $(\mathcal{O}^{bi})^{-1} = \mathcal{O}^{bi'}$. This means that \mathcal{O}^{bi} can also be used to transform body-centered vectors into inertial coordinates via $\mathbf{x} = \mathcal{O}^{bi'}\mathbf{x}^b$.

In addition to \mathcal{O}^{bi} , it is also helpful to define a transformation that maps wind coordinates into body-centered coordinates. From Figure 6.1(c), we see that $(-\beta, \alpha, 0)$ is the sequence of rotations that bring the wind frame into alignment with the body frame. Substituting $(-\beta, \alpha, 0)$ for (ψ, θ, ϕ) in (6.4) yields

$$\mathcal{O}^{bw} \triangleq \begin{bmatrix} \cos \alpha \cos \beta & -\cos \alpha \sin \beta & -\sin \alpha \\ \sin \beta & \cos \beta & 0 \\ \sin \alpha \cos \beta & -\sin \alpha \sin \beta & \cos \alpha \end{bmatrix}. \quad (6.5)$$

A vector in the wind frame can then be expressed in the body frame via $\mathbf{x}^b = \mathcal{O}^{bw}\mathbf{x}^w$.

6.1.2 Modeling the translational motion of an airplane

In the previous section, we developed the geometric framework needed to express the forces and torques that act upon an aircraft during flight. In this section, we provide an aerodynamically accurate description of the relevant forces. The issue of torque is taken up in the next section.

Broadly speaking, there are three types of forces that act upon an airplane:

1. weight due to gravity,
2. the propulsive force (or thrust) due to the engines,
3. the aerodynamic force due to forward motion through the atmosphere.

The magnitude of the weight vector is mg , where m is the mass of the target and g is the acceleration due to gravity. The direction of the weight vector is most naturally expressed in inertial coordinates where it points downward, $\mathbf{w} = mg\hat{i}_z$. The magnitude of the force due to thrust is controlled by the pilot. For a given aircraft, it will have both upper and lower bounds. The upper bound is determined by the maximum output of the engines. The lower bound is dictated by the minimum velocity at which adequate lift is generated by the wings to remain airborne. We denote the magnitude of the thrust vector as T and assume that it is directed along \hat{b}_x , the body-centered x -axis. The final item from the list, the aerodynamic force, does not lie along any of the reference axes that we have defined so far. We must consider each of its components individually.

The components of the aerodynamic force are, in general, proportional to $\rho v^2 S$, where ρ is the density of air at the aircraft's altitude, v is its speed, and S is a reference area, usually that of the wings. The aerodynamic force is most conveniently defined in

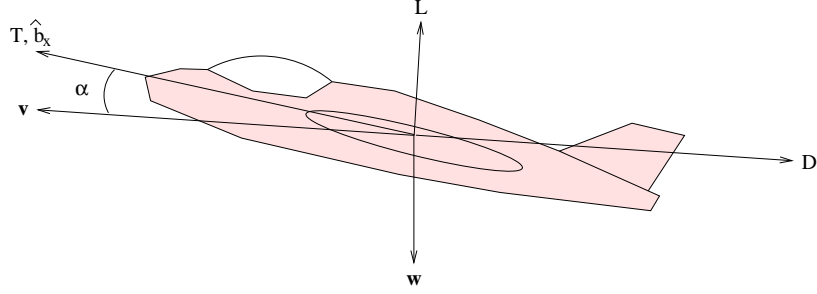


Figure 6.2: Illustration of forces relevant to flight. The cross-wind component is not shown. (It would be directed perpendicular to the page.)

the wind frame,

$$\mathbf{a}^w = - \begin{bmatrix} D \\ C \\ L \end{bmatrix} \triangleq -\frac{1}{2}\rho v^2 S \begin{bmatrix} C_D \\ C_C \\ C_L \end{bmatrix}, \quad (6.6)$$

where D , C , and L are the drag, cross-wind, and lift components, respectively, and C_D , C_C , and C_L are their nondimensional equivalents. Because \mathbf{v} points along the x -axis of the wind frame, we see that drag and lift are defined as expected. Namely, D is defined in opposition to the forward motion of the aircraft, while L is defined (approximately) perpendicular to the wings. Figure 6.2 illustrates the angular relationship between all of the relevant forces in our discussion. Of course, the definitions of these components, in and of themselves, are nothing significant. We are free to express the aerodynamic force vector in any frame of reference that we like. The advantage of this specific parameterization, though, is the simple forms that the coefficients C_D , C_C , and C_L assume. For subsonic and supersonic flight, the following approximations hold over a useful range of α [8, 73]:

$$C_L \approx C_{L_\alpha} \alpha, \quad (6.7)$$

$$C_D \approx C_{D_{\min}} + K_D C_L^2, \quad (6.8)$$

where the three terms C_{L_α} , $C_{D_{\min}}$, and K_D are principally functions of the aircraft's shape and speed. Because we are interested in maneuvers with limited variation in speed, we will ignore this aspect of their dependence. However, we will *not* ignore the dependence of C_{L_α} , $C_{D_{\min}}$, and K_D on the aircraft itself. In fact, variation of these coefficients between target classes is precisely why classification can improve tracking performance. If a target can be identified, the correct coefficients can be used within the motion model. As for the cross-wind component from (6.6), we take $C_C \approx 0$, which is a valid assumption as long as the sideslip angle β is relatively small.

We are now in a position to express the net force exerted upon an aircraft due to gravity, thrust, lift, and drag. It is most useful to do this in inertial coordinates. Using

the transformations \mathcal{O}^{bi} and \mathcal{O}^{bw} , we have

$$\begin{aligned}
\mathbf{f} &= \mathcal{O}^{bi'} (T\hat{\mathbf{b}}_x + \mathcal{O}^{bw} \mathbf{a}^w) + \mathbf{w} \\
&= \mathcal{O}^{bi'} \left(\begin{bmatrix} T \\ 0 \\ 0 \end{bmatrix} - D \begin{bmatrix} \cos \alpha \cos \beta \\ \sin \beta \\ \sin \alpha \cos \beta \end{bmatrix} - L \begin{bmatrix} -\sin \alpha \\ 0 \\ \cos \alpha \end{bmatrix} \right) + \mathbf{w} \\
&= \mathcal{O}^{bi'} \cdot \begin{bmatrix} 1 & -\cos \alpha \cos \beta & \sin \alpha \\ 0 & -\sin \beta & 0 \\ 0 & -\sin \alpha \cos \beta & -\cos \alpha \end{bmatrix} \begin{bmatrix} T \\ \frac{1}{2}\rho v^2 SC_D \\ \frac{1}{2}\rho v^2 SC_L \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix}, \quad (6.9)
\end{aligned}$$

where C_L and C_D are provided by (6.7) and (6.8). With this equation, we now have an excellent model for the net force exerted on an aircraft during flight. It is important to note that (6.9) is substantially more than just a rigid-body motion model; because of C_L and C_D , it is designed explicitly for flight. Thus, to calculate \mathbf{f} , two sets of parameters are needed. First are the class attributes. This set is defined as

$$\mathcal{A}_c \triangleq \{C_{L_\alpha}, C_{D_{\min}}, K_D, S, m\}, \quad (6.10)$$

where the subscript c identifies the class represented by the parameters. This set is completely determined, given the identity of the target. For our simulations in Chapter 7, we will assume that all five are time-invariant.² Second are the kinematic attributes $\{p_z, \mathbf{v}, \psi, \theta, \phi, T\}$. Note that p_z is only required because ρ , the density of air, is a function of altitude.³ Also, given the Euler angles (ψ, θ, ϕ) , the aerodynamic angles (α, β) can be derived from \mathbf{v} using (6.1) and (6.2).

At this point, it is now possible to summarize our entire argument for the necessity of a joint approach to tracking and classification in an RCS-based system. First, we consider the classifier. Within the VHF band, we have seen that the variation in RCS of a moving airplane can be a powerful means of target discrimination. In Section 2.2, we showed that RCS is a function of the incident and scattered wavevectors, \mathbf{k}_i and \mathbf{k}_s (see Equation (2.1)). To determine the directions of these vectors, the classifier needs an estimate of the target's position and angular orientation — precisely the output of the tracker. Second, consider the tracking system. To estimate the flight path of an aircraft, it is necessary to estimate the net force exerted on the airframe. As shown in (6.9), an accurate model for this force requires knowledge of the parameter set \mathcal{A}_c . As such, the tracker needs an estimate of the correct class c — precisely the output of the classifier.

²For example, we will ignore the fact that the mass m of the target decreases slightly with time as fuel is consumed.

³The density of air in kg/m^3 is given by $\rho = \text{Pressure}/(287.05 \cdot \text{Temperature})$, where the pressure is required in Pa and the temperature in degrees Kelvin. In the troposphere (sea level to 11 km), the following approximations are valid for dry air and p_z in meters: $\text{Pressure} \approx 101325(1 - .0065p_z/288.15)^{5.256}$ and $\text{Temperature} \approx 288.15 - .0065p_z$.

6.1.3 Modeling the rotational motion of an airplane

In the previous section, we developed an aerodynamically valid model for the force exerted on an aircraft during flight. In this section, we discuss the issue of torque. At the outset, it is very important that we recognize a key difference between the forces and torques that affect flight. For the most part, the forces on the aircraft have external origins. For example, \mathbf{w} results from the action of gravity upon the aircraft. Likewise, \mathbf{a}^w results from the action of the atmosphere upon the plane. The only force that is generated by the plane itself is the propulsive force $T\hat{b}_x$, which is controlled by the pilot via the throttle.⁴ To a good approximation, this is *not* the case with the torque exerted on an aircraft. In fact, airplanes are designed with the explicit intent of resisting rotations caused by external influences. Stability against undesired pitching and yawing is achieved by the horizontal and vertical surfaces of the tail, respectively. Stability against undesired rolls can be achieved by giving the wings of an aircraft a slight upward tilt. In a stable aircraft, the pilot can actually release the controls, and the vehicle will continue in straight and level flight [74].

Instead of having external causes, the torques that produce rolling, pitching, or yawing moments are under the direct control of the pilot. Rolling motion (rotation about \hat{b}_x) is controlled by the ailerons, the flaps at the ends of the wings. Pitching motion (rotation about \hat{b}_y) is controlled by the elevators, typically extensions on the horizontal stabilizer of the tail. Finally, yawing motion (rotation about \hat{b}_z) is controlled by the rudder. Therein lies the difference with modeling torque. Because its origin lies with the aircraft itself, modeling torque is really an exercise in modeling pilot intent. This is a distinct advantage because it means that there should always be a certain “intelligence” to the rotational motion of an airplane. The influence of pilot intent upon rotational motion can be quantified by defining a set of *rotation modes*. Each rotation mode corresponds to a different type of flight, such as a turn or a climb. With these modes, the rotational motion of an aircraft is then nothing more than the result of an unknown sequence of maneuvers $\{\gamma_1, \gamma_2, \dots, \gamma_k\}$ commanded by the pilot, where γ_k is the mode that the target operates in from $k\Delta t$ until $(k+1)\Delta t$.

Maneuvers can be classified as either longitudinal or lateral. A *longitudinal* maneuver is any that occurs with zero bank angle ($\phi = 0$). Longitudinal maneuvers are initiated by pitching moments. For example, to initiate a climb, the pilot will pitch the aircraft up to increase the angle of attack α and generate more lift (recall that $L \propto \alpha$). Maneuvers that involve nonzero bank angles are termed *lateral*. They are initiated through yawing and rolling moments. The useful assumption of *coordinated flight* is often made when dealing with lateral maneuvers. Coordinated flight is defined as flight in which the control surfaces (ailerons, elevators, and rudder) are manipulated such that the following two goals are attained. First, the velocity vector is kept in the $\hat{b}_x\hat{b}_z$ -plane (i.e., $\beta \approx 0$), so that air flows past the cabin and not into it. This minimizes drag and maximizes lift [74]. Second, the net force on objects within the cabin is directed along

⁴Of course, this is not to say that the pilot does not have some control over the lift and drag forces. For example, because L is proportional to both α and v^2 , the lift vector can be changed by altering either the angle of attack or the speed of the airplane.

Table 6.1: Euler Angles as a Function of Rotation Mode and Pilot Input

Mode	Pilot Input	Bank Angle ϕ	Elevation Angle θ	Heading Change $d\psi/dt$
CV	(none)	0	α_{ss}	0
PT	$\Delta\theta^*$	0	$\alpha_{ss} + \Delta\theta^*$	0
CT	$\omega^*, \Delta\theta^*$	$\tan^{-1}(\omega^*v/g)$	$\alpha_{ss} + \Delta\theta^*$	ω^*

the \hat{b}_z -axis. This means that coffee in a cup onboard the turning aircraft will not spill. It also minimizes the fatigue of being thrown from side to side during lateral maneuvering [8]. For our state model, we will assume that the rotational motion of the target can be described using two longitudinal modes and one (coordinated) lateral mode. The three modes are

- mode 1 (CV): constant-velocity fixed-altitude flight,
- mode 2 (PT): longitudinal climb or descent (i.e., pitch) with thrust increase,
- mode 3 (CT): constant-speed coordinated turn.

By concatenating these three modes, this small collection is capable of modeling a wide range of real trajectories. Note, we have used two labels (CV and CT) that already identify models within our IMM-EKF implementation. This is meant to signify that these particle filter rotation modes represent the same type of flight trajectory as the corresponding IMM filters. However, we must emphasize that the models themselves are *not* the same.

At this point, we could proceed by formulating an expression for the torque required to produce the rotational motion associated with each mode. However, this is unnecessary. Because our state model needs a description of the time evolution of the Euler angles (for \mathcal{O}^{bi} in (6.9) and RCS table look-up), the only use we would have for the net torque is to compute (ψ, θ, ϕ) . As such, it makes more sense to model the Euler angles directly for each rotation mode and bypass the issue of torque altogether.

Table 6.1 summarizes our model for the Euler angles during each rotation mode. Before considering each row, a few general comments are in order. First, we note that the constraint on heading angle is defined in terms of its *derivative*, not its value. CV flight, for example, requires that the heading angle be maintained at its initial value, whatever that value may be. Second, starred variables represent deterministic, but unknown, pilot inputs that drive the rotational motion of the system. At the beginning of a maneuver, these variables are supplied to the aircraft via the control surfaces. We model the response of the bank and elevation angles (ϕ and θ) to these inputs as an underdamped second-order system. Third, α_{ss} is the steady-state angle of attack needed to maintain level flight at some nominal speed. Its inclusion in the table highlights that, because $L \propto \alpha$, a positive angle of attack must be maintained at all times. For level flight, the velocity vector is parallel to the inertial $\hat{i}_x\hat{i}_y$ -plane. In this case, the elevation angle is equal to the angle of attack. We now proceed to justify the entries for each mode in Table 6.1.

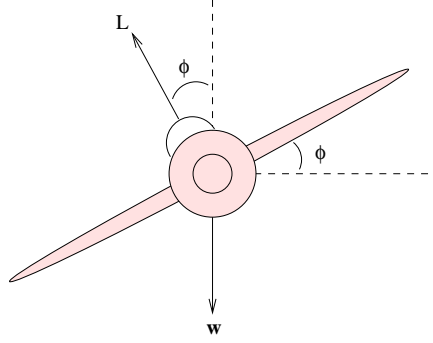


Figure 6.3: Force diagram during a constant-speed coordinated turn.

The CV model is a constant-velocity fixed-altitude mode. Because it models longitudinal flight, the bank angle must be zero, and there can be no change in heading. As discussed above, the elevation angle is equal to the steady-state angle of attack needed to generate enough lift to maintain level flight at the desired (constant) speed. The PT mode is intended to model longitudinal climbs and descents. In this mode, the pilot pitches the aircraft up or down by $\Delta\theta^*$, which changes the elevation angle by the same amount. The resulting change in the angle of attack causes the target to climb if $\Delta\theta^* > 0$ and descend if $\Delta\theta^* < 0$. In this mode, the pilot is also allowed to increase the thrust force in order to increase the aircraft's speed. (This is why there is a 'T' in the PT designation.) Finally, the CT mode is intended to model constant-speed fixed-altitude turns. As a lateral maneuver, the aircraft can roll, pitch, and yaw. However, because we have adopted a coordinated flight model for this mode, the resulting trajectory is primarily a function of ω^* , the desired turn-rate. The relation for the bank angle can be derived by considering Figure 6.3. For a constant-speed turn, the thrust and drag forces will cancel each other out (approximately), and we are left with the forces due to lift and gravity. A fixed-altitude turn requires that the net acceleration be in the $\hat{i}_x\hat{i}_y$ -plane. From Figure 6.3, this implies

$$L \cos \phi = mg \quad \Rightarrow \quad L = \frac{mg}{\cos \phi}. \quad (6.11)$$

Because the magnitude of the centripetal acceleration is ω^*v , we also have

$$L \sin \phi = m \omega^*v. \quad (6.12)$$

Substituting L from (6.11) into (6.12) yields

$$g \tan \phi = \omega^*v \quad \Rightarrow \quad \phi = \tan^{-1} \left(\frac{\omega^*v}{g} \right), \quad (6.13)$$

which is the equation for the bank angle in Table 6.1. Because a portion of the lift force is spent upon lateral movement in a turn, a pilot will typically pitch up the aircraft to increase the angle of attack and generate more lift. This practice is allowed in our CT

mode through the addition of the input $\Delta\theta^*$. Similar to the PT mode, $\Delta\theta^*$ affects the elevation angle, but in this case we require $\Delta\theta^* > 0$.

At this point, it is helpful to summarize our development thus far. We began by recognizing that the rotational motion of an airplane was generated almost entirely by pilot command. As such, it is appropriate to model the rotational motion using a set of flight modes for the most common maneuvers. We chose three modes: constant-velocity, pitch/thrust, and coordinated turn. Then, recognizing that calculation of \mathbf{f} only required knowledge of the Euler angles, we avoided the issue of torque altogether and instead modeled (ϕ, θ, ψ) directly, as shown in Table 6.1. Now, all that remains is for us to specify a model for the mode sequence itself. For this, we assume that $\{\gamma_k\}$ evolves according to a first-order Markov chain with transition matrix P . The elements of P are defined as $P(i, j) = \Pr(\gamma_k = j | \gamma_{k-1} = i)$. This completes our model for the rotational motion of an airplane.

6.1.4 The complete flight model

In this section, we combine our previous results in order to arrive at our complete flight model. So far in this chapter, we have largely neglected the sequential nature of the problem and suppressed the scan index k to simplify our equations. Now, we return to the primary problem of modeling the evolving state \mathbf{x}_k of our system. To do this, we first specify the components of \mathbf{x}_k . Based upon our models for translational and rotational motion, the state vector at scan k is defined as

$$\mathbf{x}_k = \{\mathbf{p}_k, \mathbf{v}_k, \psi_k, \theta_k, \phi_k, \zeta, \gamma_k, \omega_k^*, \Delta\theta_k^*, T_k^*\}, \quad (6.14)$$

where \mathbf{p}_k is the position vector, \mathbf{v}_k is the velocity vector, $(\psi_k, \theta_k, \phi_k)$ are the Euler angles, ζ is the class, γ_k is the current maneuver mode (*i.e.*, $\gamma_k \in \{\text{CV}, \text{PT}, \text{CT}\}$), and $(\omega_k^*, \Delta\theta_k^*, T_k^*)$ are the pilot inputs for γ_k . We note that the class ζ is not subscripted by k because it does not vary with time. Furthermore, some of $(\omega_k^*, \Delta\theta_k^*, T_k^*)$ are fixed, depending on the current mode (e.g., $\omega_k^* = 0$ if $\gamma_k = \text{CV}$; otherwise the maneuver would be a turn). Nonetheless, because \mathbf{x}_k is 14-dimensional, it would seem that sampling the state space is sure to be computationally daunting. To this concern, we point out that the effective state space is actually five-dimensional. To explain what we mean by “effective,” assume that the initial state \mathbf{x}_0 is known. Then, according to our state model, the sequence $\mathbf{x}_{0:k}$ is completely determined by the variables $\{\zeta, \gamma_{1:k}, \omega_{1:k}^*, \Delta\theta_{1:k}^*, T_{1:k}^*\}$. In words, if we are given the initial position, velocity, and orientation of the target, knowledge of the class, mode sequence, and corresponding pilot commands is sufficient to determine the state at all times. The other nine components are just deterministic functions of the first five (and each other). Therefore, if the initialization of the particle filter is sufficiently good, the majority of the subsequent inference should focus upon $\{\zeta, \gamma_k, \omega_k^*, \Delta\theta_k^*, T_k^*\}$. These five variables quantify the two key unknowns in any tracking problem: target class and pilot intent (*i.e.*, “What is it?” and “Where is it?”).

Algorithm 5 Aerodynamically Accurate State Model
 $\mathbf{x}_k = \text{STATE_MODEL}[\mathbf{x}_{k-1}]$

- Draw $\gamma_k \sim P(\gamma_{k-1}, :)$.
- IF $\gamma_k = \gamma_{k-1}$
 - Draw $\omega_k^* \sim \mathcal{N}(\omega_{k-1}^*, \sigma_\omega^2)$.
 - Draw $\Delta\theta_k^* \sim \mathcal{N}(\Delta\theta_{k-1}^*, \sigma_{\Delta\theta}^2)$.
 - Draw $T_k^* \sim \mathcal{N}(T_{k-1}^*, \sigma_T^2)$.
- ELSE
 - IF $\gamma_k = \text{CV}$
 - * Set $(\omega_k^*, \Delta\theta_k^*, T_k^*) = (0, \alpha_{ss}, T_{ss})$.
 - ELSEIF $\gamma_k = \text{PT}$
 - * Set $\omega_k^* = 0$.
 - * Draw $\Delta\theta_k^* \sim \mathbb{U}_{\Delta\theta}^{PT}(\zeta)$.
 - * Draw $T_k^* \sim \mathbb{U}_T^{PT}(\zeta)$.
 - ELSEIF $\gamma_k = \text{CT}$
 - * Draw $\omega_k^* \sim \mathbb{U}_\omega^{CT}(\zeta)$.
 - * Draw $\Delta\theta_k^* \sim \mathbb{U}_{\Delta\theta}^{CT}(\zeta)$.
 - * Set $T_k^* = T_{ss}$.
 - END IF
- END IF
- $\mathbf{x}_k = \text{INTEGRATE}[\mathbf{x}_{k-1}, \gamma_k, \omega_k^*, \Delta\theta_k^*, T_k^*]$

Figure 6.4: Pseudo-code description of our state model for flight.

Our complete state model can be expressed mathematically as

$$\mathbf{x}_k = f_k^{(\gamma_k)}(\mathbf{x}_{k-1}, \mathbf{u}_k). \quad (6.15)$$

Unfortunately, there is no simple expression for the functions $f_k^{(\gamma_k)}$. Instead, in Figure 6.4, we provide a pseudo-code description of how \mathbf{x}_k is obtained from \mathbf{x}_{k-1} . Each scan, the model first determines if a mode change has occurred. If not, a small amount of noise is added to the previous pilot inputs; if so, new pilot inputs are drawn. Note, we use $P(\gamma_{k-1}, :)$ to denote the conditional distribution over the current mode given γ_{k-1} . Steady-state values for the angle of attack and thrust are denoted by α_{ss} and T_{ss} , respectively. These are the values needed to maintain level flight at the aircraft's nominal speed. Finally, $\mathbb{U}_{\Delta\theta}^{PT}$, \mathbb{U}_T^{PT} , \mathbb{U}_ω^{CT} , and $\mathbb{U}_{\Delta\theta}^{CT}$ are uniform distributions from which the pilot inputs are drawn.⁵ Uniform distributions are used to provide a maximally non-committal model of pilot intent. Furthermore, using distributions with finite support

⁵In general, $\mathbb{U}_{\Delta\theta}^{PT}$ and $\mathbb{U}_{\Delta\theta}^{CT}$ are not the same. This is necessary because $\Delta\theta^* > 0$ during the CT mode, while $\Delta\theta^*$ can be negative during the PT mode to permit a longitudinal descent.

Algorithm 6 State Integration Routine
 $\mathbf{x}_k = \text{INTEGRATE}[\mathbf{x}_{k-1}, \gamma_k, \omega_k^*, \Delta\theta_k^*, T_k^*]$

- Store $\{\tilde{\mathbf{p}}, \tilde{\mathbf{v}}, \tilde{\psi}, \tilde{\theta}, \tilde{\phi}\} \leftarrow \{\mathbf{p}_{k-1}, \mathbf{v}_{k-1}, \psi_{k-1}, \theta_{k-1}, \phi_{k-1}\}$.
- FOR $n = 1 : (\Delta t / \Delta_i)$
 - $(\omega, \Delta\theta, T) = \text{SECOND_ORDER}[\omega_k^*, \Delta\theta_k^*, T_k^*]$
 - Determine \mathcal{O}^{bi} using $\tilde{\psi}$, $\tilde{\theta}$, and $\tilde{\phi}$ in (6.4).
 - Determine \mathbf{f} using $\tilde{\mathbf{p}}$, $\tilde{\mathbf{v}}$, and \mathcal{O}^{bi} in (6.9).
 - Calculate $\mathbf{v} = \tilde{\mathbf{v}} + (\Delta_i/m)\mathbf{f}$.
 - Calculate $\mathbf{p} = \tilde{\mathbf{p}} + \Delta_i\tilde{\mathbf{v}}$.
 - Calculate $\psi = \tilde{\psi} + \Delta_i\omega$.
 - Calculate $\theta = \alpha_{ss} + \Delta\theta$.
 - Determine ϕ using ω and $\tilde{v} = \|\tilde{\mathbf{v}}\|$ in (6.13).
 - Store $\{\tilde{\mathbf{p}}, \tilde{\mathbf{v}}, \tilde{\psi}, \tilde{\theta}, \tilde{\phi}\} \leftarrow \{\mathbf{p}, \mathbf{v}, \psi, \theta, \phi\}$.
- END FOR
- $\mathbf{x}_k = \{\mathbf{p}, \mathbf{v}, \psi, \theta, \phi, \zeta, \gamma_k, \omega_k^*, \Delta\theta_k^*, T_k^*\}$

Figure 6.5: Pseudo-code description of the state integration routine.

to model pilot input is justified because the motion of any airplane will be limited by its own maneuverability and propulsion. By allowing these distributions to depend on class ζ , the actual kinematic limitations of each airplane can be used, thereby providing another means of class discrimination.

The last line in the pseudo-code description of Figure 6.4 is a call to a numerical integration routine. This routine is responsible for the actual computation needed to advance the state from time index $k - 1$ to k . Given the previous state, it uses $(\omega_k^*, \Delta\theta_k^*, T_k^*)$, the current pilot inputs, to drive the translational and rotational flight models presented in Sections 6.1.2 and 6.1.3. A pseudo-code description of the routine is provided in Figure 6.5. A few comments should be made concerning its implementation. First, the required integrals are discretized using a time step of Δ_i . Because Δt is the sample period, $\Delta t / \Delta_i$ iterations are performed per function call. In our experiments, we will take $\Delta_i = 0.2$ seconds. Next, tildes are used to denote the dummy variables for the integration. Third, even though the Euler update equations have no functional dependence on the current mode γ_k , it can be shown that they still implement the desired model from Table 6.1. Finally, the function `SECOND_ORDER` models the response of the aircraft to pilot commands [40]. Inspection of the state model in Figure 6.4 reveals that significant changes in the pilot commands only occur during mode transitions (i.e., when $\gamma_k \neq \gamma_{k-1}$). For scans in which $\gamma_k = \gamma_{k-1}$, the inputs $(\omega_k^*, \Delta\theta_k^*, T_k^*)$ remain nearly unchanged.⁶ As such, the pilot input sequences

⁶The variances σ_ω^2 , $\sigma_{\Delta\theta}^2$, and σ_T^2 are all small. They are meant to model effects such as air turbulence.

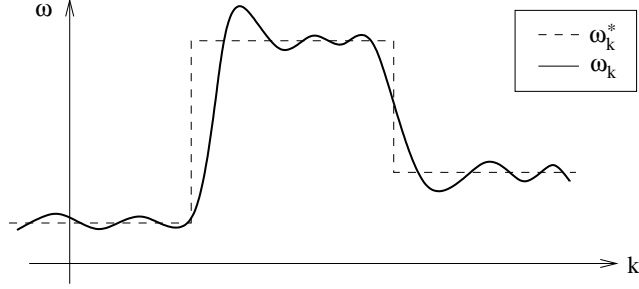


Figure 6.6: Example of airplane response to turn-rate commands from the pilot. $\{\omega_k^*\}$ is the command sequence; $\{\omega_k\}$ is the sequence of actual turn-rates achieved by the aircraft.

are (nearly) staircase functions. Because the aircraft cannot respond instantaneously to changes in these commands, `SECOND_ORDER` is used to model its response. An example is shown in Figure 6.6 for the turn-rate input ω_k^* .

This algorithmic descriptions in Figures 6.4 and 6.5 complete the description of $f_k^{(\gamma)}$. The statistics of the process noise \mathbf{u}_k are implicitly defined by the distributions used to alter the pilot inputs (e.g., $\mathbb{U}_{\Delta\theta}^{PT}$, \mathbb{U}_T^{PT} , \mathbb{U}_ω^{CT} , and $\mathbb{U}_{\Delta\theta}^{CT}$). Because our mode sequence $\{\gamma_k\}$ is Markov, the overall state model is jump Markov. In this sense, it bears a similarity with our IMM-EKF. However, while both filters use multiple maneuver modes, the IMM-EKF state equations are not based upon the actual translational and rotational motion of an airplane. The IMM-EKF cannot implement the state equations from this chapter because it has no way of estimating an airplane's angular orientation (especially the bank angle). RCS provides the particle filter with this capability. In the next chapter, we will demonstrate the significant improvement in tracking accuracy that can be achieved with our aerodynamically valid state model.

6.2 Measurement Model for the Particle Filter

In the previous section, we presented the state model for our particle filter. In this section, we specify our measurement model and the data likelihood function required to compute the particle weights. Similar to our discussion concerning the IMM-EKF in Chapter 5, we begin by considering the measurement model for a single target in a clutter-free environment with zero probability of miss. The issue of data association for measurements of uncertain origin is addressed in the next section.

As mentioned previously, one of the particle filter's key advantages for joint tracking/classification is its ability to incorporate RCS as a data feature. Assuming a single target in a clutter-free environment, we have

$$\mathbf{z}_k = [\tau_k, d_k, \sigma_k]', \quad (6.16)$$

where τ_k , d_k , and σ_k correspond to measurements of delay, Doppler shift, and RCS,

respectively.⁷ Because we model the noise affecting each component of (6.16) as independent, the data likelihood is just the product of the component likelihoods,

$$p(\mathbf{z}_k|\mathbf{x}_k) = p(\tau_k|\mathbf{x}_k) p(d_k|\mathbf{x}_k) p(\sigma_k|\mathbf{x}_k). \quad (6.17)$$

Because our IMM-EKF uses delay and Doppler, the first two terms in (6.17) were already defined in Section 5.2,

$$p(\tau_k|\mathbf{x}_k) = \mathcal{N}(\tau_k; h_k^{(\tau)}(\mathbf{x}_k), \sigma_{\tau_k}^2), \quad (6.18)$$

$$p(d_k|\mathbf{x}_k) = \mathcal{N}(d_k; h_k^{(d)}(\mathbf{x}_k), \sigma_{d_k}^2), \quad (6.19)$$

where $h_k^{(\tau)}$, the mapping from state to delay, is defined implicitly by (5.13) and $h_k^{(d)}$, the mapping from state to Doppler shift, is defined implicitly by (5.14)–(5.15). All that remains is for us to specify $p(\sigma_k|\mathbf{x}_k)$, the RCS likelihood function. Unfortunately, an additive Gaussian noise model is inappropriate for RCS. To understand why this is so, we must first consider the front-end processing that occurs in a radar receiver.

The operation of a typical radar system can be thought of as occurring in two stages: front-end processing and back-end processing. The front-end is responsible for extracting the desired data features (e.g., delay and Doppler shift) from the received signal. The back-end is responsible for detection and estimation, given the data from the front-end. The type of processing that occurs in the front-end is therefore determined by the desired feature set. Because most radars measure Doppler shift using a complex fast Fourier transform, we will assume that the front-end of our radar consists of an in-phase channel and a quadrature channel (i.e., two matched filters, 90 degrees out of phase) [64]. With this configuration, it is actually possible for our radar to extract the complex scattering coefficient from the reflected waveform. Because we will assume that only one polarization is measured, we suppress the vv or hh subscript and denote the scattering coefficient from scan k as s_k .

Because RCS is equal to $|s_k|^2$ (see Section 2.3), it would seem that we are throwing away useful information by not using the scattering coefficient itself. However, the relatively poor range resolution of VHF-band radars makes the phase of s_k unreliable. This leaves us with $|s_k|$, which is equivalent to using RCS. Because the thermal noise in each channel of our receiver can be modeled as additive Gaussian, we arrive at the following model,

$$\sigma_k = \left(\text{Re}(s_k) + n_k^{(I)} \right)^2 + \left(\text{Im}(s_k) + n_k^{(Q)} \right)^2, \quad (6.20)$$

where $\{n_k^{(I)}\}$ and $\{n_k^{(Q)}\}$ are the in-phase and quadrature noise processes, respectively. We assume that both are mutually independent and zero-mean Gaussian with variance $\sigma_{n_k}^2$. In this case, σ_k will have a noncentral chi-squared distribution with two degrees

⁷Because both conventions are widespread, we use σ to represent RCS and standard deviation. In all cases, the identity of the variable should be clear from either its subscript or its superscript.

of freedom,

$$p(\sigma_k | |s_k|) = \frac{1}{2\sigma_{n_k}^2} \exp\left(-\frac{\sigma_k + |s_k|^2}{2\sigma_{n_k}^2}\right) I_0\left(\frac{\sqrt{\sigma_k}|s_k|}{\sigma_{n_k}^2}\right), \quad (6.21)$$

where I_0 is the zero-order modified Bessel function of the first kind. As a noncentral chi-squared random variable, the first two moments of σ_k are

$$E[\sigma_k] = |s_k|^2 + 2\sigma_{n_k}^2, \quad (6.22)$$

$$\text{var}[\sigma_k] = 4\sigma_{n_k}^2 \left(|s_k|^2 + \sigma_{n_k}^2\right). \quad (6.23)$$

Thus, we see that σ_k is biased, and its variance grows as $\sigma_{n_k}^4$. In Chapter 7, we will investigate the effect that the scattering coefficient variance has on the particle filter's performance.

Before continuing, a comment should be made concerning (6.20). Strictly speaking, the receiver measures the magnitude of the in-phase and quadrature components of the received waveform; Equation (2.2) must be used to recover the magnitude of the I/Q components of s_k . Of course, (2.2) requires knowledge of the target's position. Because the typical error in position estimate is small when compared to the distances between the target and the transmitters and receiver, we make the simplifying assumption that the I/Q components of s_k are measured directly.

This completes the specification of the measurement model for our particle filter. In summary, the measurement process is of the form $\mathbf{z}_k = h_k(\mathbf{x}_k, \mathbf{w}_k)$, where the influence of \mathbf{w}_k is no longer additive because of (6.20). Our particle weight update equation is then

$$\bar{w}_k^{(i)} \propto \bar{w}_{k-1}^{(i)} \cdot \frac{p(\mathbf{z}_k | \mathbf{x}_k^{(i)}) p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)})}{\pi(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)}, \mathbf{z}_k)}, \quad i = 1, \dots, N_p, \quad (6.24)$$

where

$$p(\mathbf{z}_k | \mathbf{x}_k^{(i)}) = \mathcal{N}(\tau_k; h_k^{(\tau)}(\mathbf{x}_k^{(i)}), \sigma_{\tau_k}^2) \mathcal{N}(d_k; h_k^{(d)}(\mathbf{x}_k^{(i)}), \sigma_{d_k}^2) p(\sigma_k | \mathbf{x}_k^{(i)}), \quad (6.25)$$

and $p(\sigma_k | \mathbf{x}_k^{(i)})$ is given by (6.21) with $|s_k|$ determined by table look-up. More precisely, given \mathbf{p}_k and $(\psi_k, \theta_k, \phi_k)$ from $\mathbf{x}_k^{(i)}$, the incident and scattered directions for the FM radio signal can be determined. These angles are then be used to access the RCS table compiled for the target class corresponding to the label ζ from $\mathbf{x}_k^{(i)}$.

6.3 Data Association for the Particle Filter

So far, in the development of our particle filter, we have restricted our attention to tracking a single target in a clutter-free environment with zero probability of miss. In this section, we extend our discussion to include filtering in the presence of multiple targets, false alarms, and nonzero probability of miss. In this case, our data vector

becomes

$$\mathbf{z}_k = \left[\left(\tau_k^{(1)} d_k^{(1)} \sigma_k^{(1)} \right) \dots \left(\tau_k^{(m_k)} d_k^{(m_k)} \sigma_k^{(m_k)} \right) \right]', \quad (6.26)$$

where m_k is the number of returns in scan k . Similar to our presentation in Sections 5.3.1–5.3.3 for the EKF, the uncertain origin of each delay-Doppler-RCS triplet will now be addressed. We begin by considering the case of a single target in the presence of clutter and missed detections.

6.3.1 PDA particle filter

In discussing data association for our particle filter, we use the same notation that was introduced in Section 5.3.1. Therefore, β and P_D will represent the clutter density and probability of detection, respectively. Furthermore, we use the superscript notation $\mathbf{z}_k^{(j)}$ to denote a single delay-Doppler-RCS triplet, $(\tau_k^{(j)} d_k^{(j)} \sigma_k^{(j)})$. Where it does not lead to confusion, we will refer to $\mathbf{z}_k^{(j)}$ as a single “measurement.” In this section, we consider the case of tracking a single target when $\beta > 0$ and $P_D < 1$. Because these additional assumptions do not affect our state model, we only need to reformulate the update equation for the particle weights. Our approach is equivalent to the one in [65].

As presented in Section 5.3.1, there are $m_k + 1$ association hypotheses in the single-target case. Either one of the m_k measurements is correct, or all are false. As before, the hypothesis that $m_k - 1$ measurements are false and $\mathbf{z}_k^{(j)}$ is correct will be denoted as $\Gamma_k^{(j)}$ (for $j = 1, \dots, m_k$). The hypothesis that all m_k measurements are false will be denoted as $\Gamma_k^{(0)}$. Referring to (6.24), we see that the likelihood function $p(\mathbf{z}_k | \mathbf{x}_k)$ is required to update the particle weights. When there is uncertainty in the origin of the measurements in \mathbf{z}_k , we have

$$p(\mathbf{z}_k | \mathbf{x}_k) = \sum_{j=0}^{m_k} p(\mathbf{z}_k | \Gamma_k^{(j)}, \mathbf{x}_k) \Pr(\Gamma_k^{(j)} | \mathbf{x}_k), \quad (6.27)$$

because $\{\Gamma_k^{(j)}\}_{j=0}^{m_k}$ are mutually exclusive and the only feasible associations. Recall, each $\Gamma_k^{(j)}$ contains three items: (1) the number of false alarms, (2) the number of detections, and (3) a measurement index for each target hypothesized to have been detected. Because the probability of detection is assumed to be independent of the round-trip range, we have $\Pr(\Gamma_k^{(j)} | \mathbf{x}_k) = \Pr(\Gamma_k^{(j)})$. Substituting the results from (5.27) and (5.29), we have

$$p(\mathbf{z}_k | \mathbf{x}_k) = p(\mathbf{z}_k | \Gamma_k^{(0)}, \mathbf{x}_k) \Pr(\Gamma_k^{(0)}) + \sum_{j=1}^{m_k} p(\mathbf{z}_k | \Gamma_k^{(j)}, \mathbf{x}_k) \Pr(\Gamma_k^{(j)}) \quad (6.28)$$

$$\begin{aligned} &= V_k^{-m_k} (1 - P_D P_G) \mu_F(m_k) \\ &\quad + \frac{V_k^{-(m_k-1)} P_D \mu_F(m_k - 1)}{m_k} \sum_{j=1}^{m_k} p(\mathbf{z}_k^{(j)} | \mathbf{x}_k) \end{aligned} \quad (6.29)$$

$$= \frac{e^{-\beta V_k} \beta^{m_k-1}}{m_k!} \left((1 - P_D P_G) \beta + P_D \sum_{j=1}^{m_k} p(\mathbf{z}_k^{(j)} | \mathbf{x}_k) \right), \quad (6.30)$$

where V_k is the gate volume at scan k , P_G is the probability that the target measurement is contained within that volume, and μ_F is assumed to be Poisson distributed. Because the particle filter weights are normalized, the constant term in (6.30) can be neglected, and we are left with the following modified weight update equation,

$$\bar{w}_k^{(i)} \propto \bar{w}_{k-1}^{(i)} \cdot \frac{\left((1 - P_D P_G) \beta + P_D \sum_{j=1}^{m_k} p(\mathbf{z}_k^{(j)} | \mathbf{x}_k^{(i)}) \right) p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)})}{\pi(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)}, \mathbf{z}_k)}, \quad (6.31)$$

where $p(\mathbf{z}_k^{(j)} | \mathbf{x}_k^{(i)})$ is defined in (6.25). Equation (6.31) is the probabilistic data association rule for updating the particle weights. We see that the likelihood $p(\mathbf{z}_k | \mathbf{x}_k^{(i)})$ from (6.24) has been replaced by a sum over the component likelihoods. The constant term $(1 - P_D P_G) \beta$ can be thought of as a threshold that must be exceeded in order for a particle's likelihood to stand out from the rest. For example, if β is very large, the PDA update rule yields $p(\mathbf{z}_k | \mathbf{x}_k^{(i)}) \approx C(1 - P_D P_G) \beta$ for all i . In this case, those particles with large values of $p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)})$ will end up with the most significant weights, implying that the prior is more trustworthy than \mathbf{z}_k when the clutter density is high.

6.3.2 Joint data association for particle filters

Having discussed how to alter the particle weight equation when tracking a single target in the presence of clutter and missed detections, we now consider the multitarget scenario. Recall, the extension from single target data association to multitarget data association (PDAF to JPDA) was relatively straightforward for the extended Kalman filter. The association probabilities $\Pr(\Gamma_k^{(l)} | \mathbf{z}_{1:k})$ for targets whose gate volumes overlapped just needed to be computed jointly (see Section 5.3.3). Unfortunately, the extension to multiple targets is not as simple for particle filters.

Define N_t as the number of targets that we wish to track. Then, the state of the multitarget system is

$$\mathbf{x}_k = \left\{ \mathbf{x}_k^{(1)} \ \mathbf{x}_k^{(2)} \ \dots \ \mathbf{x}_k^{(N_t)} \right\}, \quad (6.32)$$

where the superscript notation $\mathbf{x}_k^{(t)}$ is used to denote those components of \mathbf{x}_k that correspond to target t .⁸ The multitarget state vector is just the concatenation of N_t single-target "substates." Note that (6.32) does not imply any sort of relationship between $\mathbf{x}_k^{(t_1)}$ and $\mathbf{x}_k^{(t_2)}$ for $t_1 \neq t_2$. In fact, each $\mathbf{x}_k^{(t)}$ is assumed to evolve independently of all others. With these definitions, the likelihood function can be expressed as

$$p(\mathbf{z}_k | \mathbf{x}_k) = \sum_{l=1}^{M_k} p(\mathbf{z}_k | \Gamma_k^{(l)}, \mathbf{x}_k) \Pr(\Gamma_k^{(l)} | \mathbf{x}_k), \quad (6.33)$$

where M_k is the total number of association hypotheses at scan k . If we assume that P_D is a constant and N_t is known, the association probability is independent of the state vector, $\Pr(\Gamma_k^{(l)} | \mathbf{x}_k) = \Pr(\Gamma_k^{(l)})$. This probability can then be found in much the

⁸Note, in this section only, the term $\mathbf{x}_k^{(t)}$ will be used to denote a single-target state, not the t^{th} sample from a proposal distribution.

same way as the single-target case. The conditional likelihood term from (6.33) is

$$p(\mathbf{z}_k | \Gamma_k^{(l)}, \mathbf{x}_k) = V_k^{-N_F^l} \prod_{t \in \Gamma_k^{(l)}} P_G^{-1} p(\mathbf{z}_k^{(j_t)} | \mathbf{x}_k^{(t)}), \quad (6.34)$$

where N_F^l is the number of false alarms under $\Gamma_k^{(l)}$, and the product is taken over the targets that were detected. The superscript in $\mathbf{z}_k^{(j_t)}$ identifies the measurement from \mathbf{z}_k that is associated with target t under hypothesis $\Gamma_k^{(l)}$. The component likelihood $p(\mathbf{z}_k^{(j_t)} | \mathbf{x}_k^{(t)})$ can be calculated using (6.25).

On the surface, it would seem that the extension to multiple targets is trivial for particle filters. Conceptually, this is true, but there is a serious practical difficulty. Consider the definition of the multitarget state vector in (6.32) again. As a concatenation of single-target substates, the resulting dimension of \mathbf{x}_k is N_t times larger than the single-target case. To take a numeric example, tracking $N_t = 10$ targets using the flight model from Section 6.1 would require a state with $10 \cdot 14 = 140$ components! To adequately represent the posterior density in such a high-dimensional space, the number of particles would need to be substantially larger than the single-target case. Furthermore, as N_t grows, the chance that at least one of the component likelihoods from (6.34) is negligible for each $\Gamma_k^{(l)}$ increases. This, in turn, can lead to degeneracy of the particle weights and the need for frequent resampling. In other words, many good single-state estimates $\mathbf{x}_k^{(t)}$ will be rejected during resampling because they are in states that also contain poor single-state estimates. One technique that has been proposed in the literature to reduce this effect is known as *independent partition particle filtering* [75]. In this algorithm, substates are swapped between particles so that the components $\mathbf{x}_k^{(t)}$ in a given state \mathbf{x}_k are either mostly good or mostly bad. The particle weights must then be modified to undo the bias introduced by this crossover operation.

A second option for multitarget tracking is to simply maintain a separate particle filter for each target. In this case, computation grows linearly with N_t , as opposed to the potentially exponential growth in the case discussed above. The disadvantage, of course, is that data association can no longer be performed jointly across all N_t targets. Instead, data association is performed separately for each filter using the PDA update (6.31) from the previous section. While this might sound like a substantial disadvantage, recall that joint association is only required when the gates of two or more targets overlap. For gates to overlap in our application, the targets would have to generate similar delay, Doppler, and RCS measurements. Across multiple scans, similar delay and Doppler measurements imply that the targets have roughly the same speed, heading, and location. Even if this were the case, their RCS measurements might still differ enough for their gates to remain disjoint.⁹

Therefore, while it is conceptually attractive to handle data association jointly in the multitarget scenario, practically speaking, it may be unnecessary. For this reason, we will use a separate particle filter for each target in our implementation. In Chapter 7,

⁹Identical airplanes flying in formation would generate the same delay, Doppler, and RCS. In this case, though, sensor resolution is more of a limiting factor than data association.

we will show that our PF-based system outperforms our IMM-EKF, which uses joint data association, even when multiple targets are present.

CHAPTER 7

EXPERIMENTAL RESULTS

In this chapter, we present experimental results comparing the performance of the IMM extended Kalman filter described in Chapter 5 and two variants of the particle filter from Chapter 6. In the following sections, we will compare the performance of these systems on a series of increasingly difficult tasks. First, we consider the simplest case of tracking a single target with known identity. Then, we extend the single-target case to allow both false alarms and missed detections. Next, we explore the classification capability of the particle filter when tracking a single target. Finally, we conclude with the most challenging scenario: joint tracking and classification of multiple targets in the presence of clutter and missed detections. However, before we begin, there are several preliminary matters that must be addressed, such as our method for generating simulated data and definitions for the metrics that we will use to gauge tracking and classification performance.

7.1 Preliminaries

Although many of the techniques presented throughout this report are applicable to a broad range of tracking and classification problems, we will focus on the specific application of commercial FM radio-based passive radar now. This decision impacts our simulations in two important ways. First, operation in the VHF band (30–300 MHz) places practical limitations on the type and quality of measurement data available. Whereas a typical active radar operating at X-band (8–12.5 GHz) would most likely use delay, angle of arrival (both azimuth and elevation), and possibly Doppler shift, our extended Kalman filter will operate with only delay and Doppler, while our particle filter uses delay, Doppler, and radar cross section. We neglect angle measurements at VHF frequencies because typical antenna dimensions result in beamwidths that are too large to be useful. For all simulations, we take $\sigma_r^2 = 0.0001 \text{ ms}^2$ and $\sigma_d^2 = 1 \text{ Hz}^2$. This corresponds to range and range rate uncertainties from a single measurement on the order of 1500 m and 1.5 m/s, respectively. These are reasonable values for a commercial FM broadcast signal.

In addition to measurement availability and quality, focusing on FM-band passive radar influences the design of our simulations in a second important way. Namely,

because transmitters are essentially provided for free in a passive radar, multistatic systems are quite feasible (and, in a sense, necessary to make up for the shortcomings in data availability and quality). Because a constant round-trip range contour in the bistatic scenario is an ellipsoid, a minimum of three delay measurements are needed to locate a target in three-dimensional space. In addition, we have seen in Equations (5.14) and (5.15) that Doppler shift is related to the projection of the target’s velocity vector onto the inward normal of the constant range contour at the location of the target. Thus, a minimum of three transmitters are also required for unique determination of a target’s three-dimensional velocity vector from measurements of Doppler shift. Of course, in the presence of noise, we could use more than three transmitters to improve tracking accuracy, but any such benefit would need to be weighed against the resulting increase in hardware cost required to field such a system. As such, we will use the minimum number of transmitters in our simulations.

For simplicity, we also assume a single receiver location, although both the IMM-EKF and PF could be fielded with receivers at multiple sites.¹ In a sense, by restricting our simulations to three transmitters and a single receiver, we are considering the most challenging, but least expensive, configuration of an FM-band passive radar system. For all experiments, we fix the sample period to $\Delta t = 0.4$ s. As discussed earlier, we also assume that only one FM radio signal is processed at a time. Therefore, the measurement acquisition cycles for the three transmitters are interleaved, with each station processed for 0.4 out of every 1.2 s.

In our experiments, we will consider the two trajectories depicted in Figure 7.1. The triangular patches in the upper panels of the figure indicate the time-varying angular orientation of the aircraft. Both last for 30 s and are realizations of the flight model introduced for the particle filter in Section 6.1. Under this model, at any given time, an aircraft operates in one of the following three rotation modes:

- mode 1 (CV): constant velocity fixed-altitude flight,
- mode 2 (PT): longitudinal climb or descent with thrust increase,
- mode 3 (CT): constant-speed coordinated turn.

In the top row of Figure 7.1, the projection of the trajectory onto the ground plane is color-coded to indicate the underlying mode sequence. The lower two panes in the figure indicate the location of the maneuver relative to the receiver and FM transmitters. The three transmitters are actual commercial FM radio facilities, all located in the state of Maryland. We use their actual latitudes, longitudes, and altitudes and identify each by its FCC call sign. Because we adopt a receiver-centered system as an approximation for a true inertial reference frame, the receiver is located at the origin.

As discussed in Section 6.1.4, our particle filter’s flight (or state) model is specified by its Markov transition matrix and the distributions $\mathbb{U}_{\Delta\theta}^{PT}$, \mathbb{U}_T^{PT} , \mathbb{U}_ω^{CT} , and $\mathbb{U}_{\Delta\theta}^{CT}$ from which the pilot command variables $\{\omega_k^*, \Delta\theta_k^*, T_k^*\}$ are drawn. The transition matrix

¹We use the term “receiver” to refer broadly to the entire collection of hardware needed for the reception of both direct path and reflected commercial FM signals at a single location.

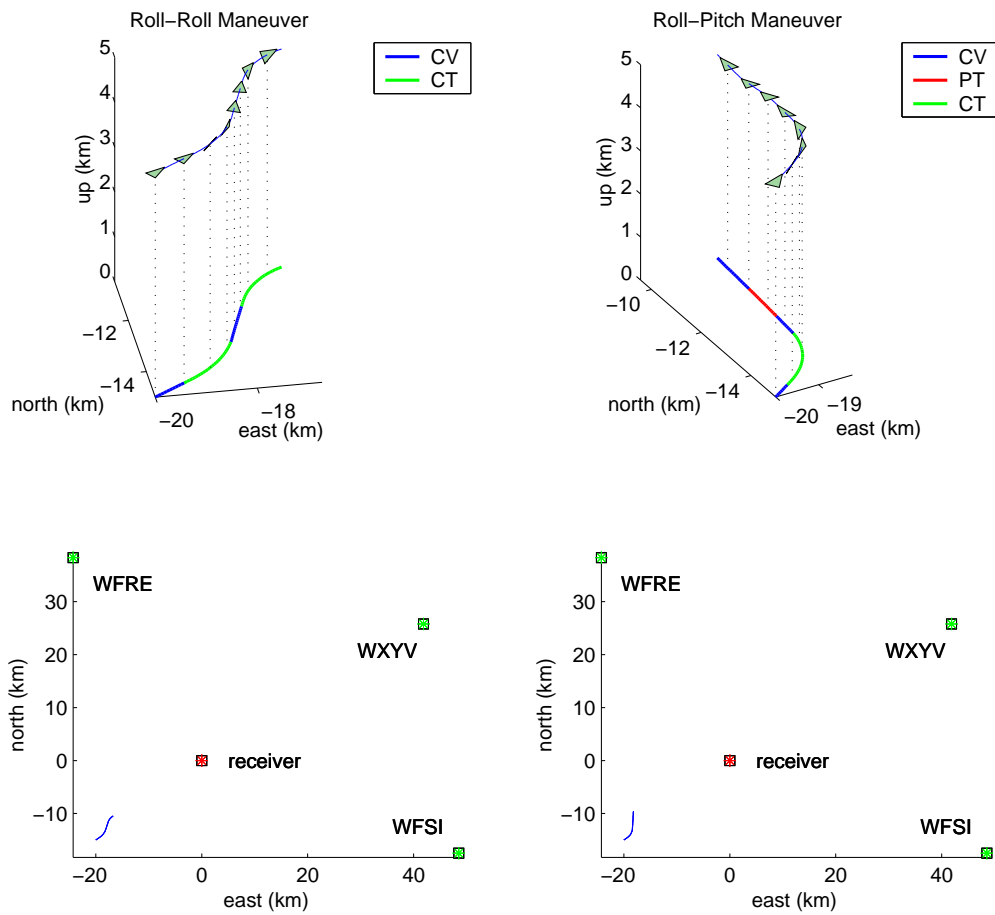


Figure 7.1: Top row: trajectories for the roll-roll maneuver and the roll-pitch maneuver. The projection of the path onto the ground plane is color-coded to indicate the time-varying rotation mode. The bottom row shows the locations of the target maneuvers relative to the receiver and the FM radio transmitters.

Table 7.1: Ranges of Values for Pilot Command Variables

Mode	Command Variable	True Range		PF Range	
		min	max	min	max
PT	$ \Delta\theta_k^* $ (deg)	7.45	14.90	5.59	22.35
	T_k^* (T_{ss})	2.5	5	1.25	5
CT	$ \omega_k^* $ (deg/sec)	3.45	6.88	2.58	10.31
	$\Delta\theta_k^*$ (deg)	2	2	1	3



F-4E Phantom

Length	19.20 m
Wingspan	11.71 m
Height	5.03 m
Wing Area	49.23 m ²
Max Speed	670 m/s
Weight (empty)	13397 kg

Figure 7.2: The F-4E Phantom fighter and its dimensions. The kinematic coefficients used in our single-target experiments were chosen to simulate the flight characteristics of the F-4E.

governs the frequency of mode changes while the turn-rate, pitch, and thrust commands produce the actual maneuvers. Both trajectories were generated using the transition matrix

$$P = \begin{bmatrix} 0.938 & 0.031 & 0.031 \\ 0.062 - \epsilon & 0.938 & \epsilon \\ 0.038 - \epsilon & \epsilon & 0.962 \end{bmatrix}, \quad (7.1)$$

where ϵ is a small probability, and the index-to-mode mapping is $1 \rightarrow CV$, $2 \rightarrow PT$, and $3 \rightarrow CT$. P yields expected dwell periods of 6, 6, and 10 s for the CV, PT, and CT modes, respectively. The third and fourth columns in Table 7.1 list the ranges over which the command variables were allowed to vary in order to produce the roll-roll and roll-pitch maneuvers shown in Figure 7.1. The kinematic coefficients used to generate these two trajectories were chosen to match those of an F-4E Phantom fighter plane, as shown in Figure 7.2. The F-4E Phantom was designed in the late 1960s and earned its reputation for excellent performance and good maneuverability during the Vietnam War. It has since been retired from combat use. In all experiments, the particle filter will use the true value of the transition matrix P . However, to make the task more challenging, the ranges of the particle filter's command variables will be extended relative to the true ranges used to generate the trajectories. These extended ranges are listed in the fifth and sixth columns of Table 7.1.

7.1.1 Performance metrics

In order to present a wide range of results as compactly as possible, we will need to quantify various aspects of system performance using simple figures of merit whenever

possible. We begin by defining the scan- k estimation error from the n^{th} Monte Carlo trial as

$$e_{k,n} \triangleq \sqrt{\left(\hat{\mathbf{p}}_{k,n} - \mathbf{p}_k^{(\text{true})}\right)' \left(\hat{\mathbf{p}}_{k,n} - \mathbf{p}_k^{(\text{true})}\right)}, \quad (7.2)$$

where $\hat{\mathbf{p}}_{k,n}$ is the position estimate produced by the tracker (EKF or PF), and $\mathbf{p}_k^{(\text{true})}$ denotes the ground truth for scan k . In our experiments, a target will be considered “lost” if $e_{N_k,n} > 500$ m, where N_k is the number of scans per trial. ($N_k = 76$ for a 30 s maneuver with $\Delta t = 0.4$.) If N_{MC} is the number of Monte Carlo trials and N_L is the total number of lost tracks during these trials, we then define the RMS error as

$$e_{RMS} = \sqrt{\frac{1}{(N_{MC} - N_L)N_k} \sum_{n=1}^{N_{MC}-N_L} \sum_{k=1}^{N_k} (e_{k,n})^2}. \quad (7.3)$$

By excluding lost tracks from (7.3), we can obtain the RMS error with fewer Monte Carlo trials because extremely bad runs will not skew the average. The tracking performance of each filter must then be gauged using both quantities: the RMS error e_{RMS} and the number of lost tracks N_L . The computational complexity of each filter will be evaluated using the average time per trial. Finally, classification performance (for the particle filter) will be quantified using confusion matrices. For all experiments, we take $N_{MC} = 100$. This number was found to offer a reasonable trade-off between the precision of our results and the computational resources needed to conduct the simulations.

7.1.2 Track maintenance

Although we want our simulations to be as realistic as possible, there is one key difference between our filters and the software component of an actual tracking system: neither of our systems handles track maintenance. *Track maintenance* typically involves the computation of a track score function to accomplish three tasks: (1) initiation of potential new tracks using recent measurements that were not associated with any existing tracks, (2) confirmation of initiated tracks whose scores have surpassed a prescribed threshold, and (3) deletion of confirmed tracks whose scores have fallen below a prescribed threshold [64]. Using this terminology, each confirmed track should correspond to an actual target.

We avoid the issue of track maintenance in our simulations by assuming that the number of targets N_t is known. This assumption does *not* eliminate the need for data association because, while both filters search for exactly N_t targets each scan, the correct measurement-to-track assignment remains unknown. Without a track maintenance routine, we need an explicit way of initializing the N_t tracks for each filter. In a real system, unassociated delay and Doppler measurements from consecutive scans would be clustered and fed to a nonlinear solver to provide initial values of position and velocity. Because both our IMM-EKF and PF could use the same nonlinear solver to generate initial position and velocity estimates, inclusion of this aspect of target track-

ing would not offer much insight for our comparative analysis. Instead, each model of the IMM-EKF is initialized with $\mathbf{x}_1^{(\text{true})}$, the true value of the target state from the first scan, and a diagonal covariance matrix with position and velocity standard deviations of 50 m and 0.5 m/s, respectively. The initial particle set for the PF is drawn from a Gaussian distribution with mean $\mathbf{x}_1^{(\text{true})}$ and the same values of standard deviation for position and velocity.² All tracks are initialized in the constant velocity mode of the corresponding filter. As such, both filters begin with the same initial distribution. The relatively high accuracy of the initialization is used to model the scenario where a non-maneuvering target has been tracked for a substantial period of time. This allows us to focus on the effects of target maneuvers in our simulations.

7.1.3 Generation of synthetic RCS data

As discussed in Chapter 6, the key difference between the particle filter and the extended Kalman filter is the ability to incorporate RCS as a data feature. This ability allows us to use an aerodynamically valid state model, which links the tasks of tracking and classification through a handful of kinematic coefficients (\mathcal{A}_c from Equation (6.10)). In Chapter 2, we demonstrated that there is no closed-form relationship between a target’s RCS and its orientation with respect to the transmitter and receiver. Furthermore, in the VHF band in which we are interested, popular high-frequency approximations are invalid. Instead, Maxwell’s equations must be solved using a method of moments solver such as FISC. In Chapter 2, we advocated using a program like FISC to build RCS tables for every target class. Each table would contain RCS values for various frequencies and incident/scattered directions. The extent of the incident/scattered angle space that would need to be sampled would depend on the maneuvers that each target might execute while being tracked. For example, the RCS table for a target that never rolled more than 20° would not need to include incident directions with large elevation angles because it would be impossible for FM transmitters on the ground to ever “see” the top of the plane.³

Unfortunately, this presents a problem for us. In our simulations, we are primarily interested in maneuvering targets. Thus, our RCS tables would need to include virtually all possible combinations of incident/scattered directions. Creating RCS tables such as these for a large number of targets is a computationally daunting task. Compounding this difficulty, the CAD model, which defines the target geometry for FISC, must be designed to precise standards. CAD models of this caliber are difficult to obtain, meaning the extent of our classification experiments would be limited by the number of CAD models we could obtain, rather than the capabilities of our PF algorithm. Finally, the military nature of some of the CAD models that we do possess might preclude publica-

²The initial angular orientation of the particle filter must also be specified. For all particles, the roll angle is initialized as zero. The yaw angle for each particle is set so that the velocity vector lies in the $\hat{b}_x \hat{b}_z$ -plane (i.e., zero sideslip angle). The pitch of each particle is drawn from a uniform distribution about the true value.

³This does not imply that the top of the aircraft is not involved in the scattering process. At wavelengths in the resonance region for a target, the entire surface contributes to the scattering mechanism. Rather, we mean that it is impossible for a wavefront launched from the ground to arrive at a nonbanking target from above.

tion of our results in the open literature. For all these reasons, we instead seek a means of generating RCS data synthetically.

In our simulations, RCS data will be simulated using a scattering center model. It is important that we be clear on this point: a scattering center model is *not* valid in the VHF band for fighter-sized aircraft because the scattering response is no longer dominated by a few distinct target features (e.g., edges and corners). However, this is not a problem for us. We merely need a model that generates “RCS-like” data; whether our synthetic data matches the RCS of any existing aircraft is not a concern for our simulations. To reiterate, a real implementation of our system, charged with tracking and classifying real aircraft, would require an accurate RCS table for each target in the class library. However, for proof-of-concept, we really just need data with properties similar to RCS, and scattering center models have been cited as being among the most useful and powerful tools in scattering theory. They provide a simple physical model that accounts for the lobe structure of RCS and provides a basis for simulations [76]. In addition, as a model, they become increasingly accurate at high frequencies. Of course, the question immediately arises as to whether our results will be meaningful without the use of RCS data from real aircraft. While the use of synthetic RCS impacts our ability to claim that our classifier would work in the field, the lack of actual radar data makes it difficult to know whether other effects (e.g., multipath distortion) would invalidate our results. Thus, for our simulations, we will focus on proof-of-concept using a synthetic model for RCS. This will allow us to create as many target classes as needed.

For our synthetic RCS model, we modify a popular parametric scattering center model [77]. Because commercial FM radio signals have narrow bandwidths, we can ignore the frequency dependence that is included in the amplitude coefficients of models based upon the geometrical theory of diffraction (GTD). Instead, we model all scattering centers as point scatterers and express the amplitude of the normalized scattered electric field (for a given polarization) as

$$E^s(\lambda, \hat{i}_i, \hat{i}_s) = \sum_{m=1}^{N_{sc}} \tilde{a}_m e^{j \frac{2\pi}{\lambda} (\hat{i}_i + \hat{i}_s) \cdot \mathbf{r}_m}, \quad (7.4)$$

where N_{sc} is the number of scattering centers used by the model, and $\{\mathbf{r}_m, \tilde{a}_m : m = 1, \dots, N_{sc}\}$ is the collection of their locations and complex amplitudes. The unit vectors \hat{i}_i and \hat{i}_s represent the incident and scattered (or observed) directions in the body frame of the target.⁴ According to this model, each scatterer is visible from all incident and scattered directions. This is not a good approximation in the VHF band, and therefore we redefine the scattering center amplitudes as

$$\tilde{a}_m = a_m e^{-\frac{(\hat{i}_m \cdot \hat{i}_i - 1)^2}{2\sigma_m^2}} e^{-\frac{(\hat{i}_m \cdot \hat{i}_s - 1)^2}{2\sigma_m^2}}. \quad (7.5)$$

⁴The vector \hat{i}_i points towards the transmitter (i.e., opposite the direction of the incoming wavefront). The vector \hat{i}_s points towards the receiver (i.e., along the direction of the scattered wavefront).

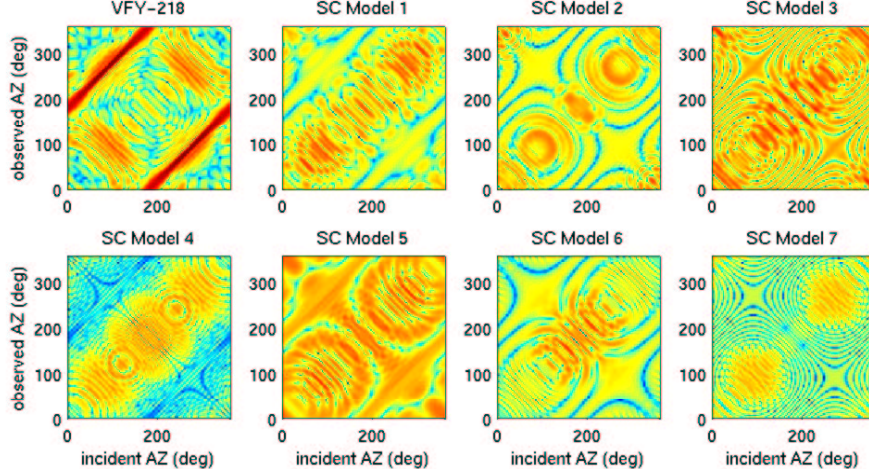


Figure 7.3: Bistatic RCS in dBsm for 0° incident elevation and 0° observed elevation. The upper left panel is the RCS generated by FISC for a VFY-218 fighter. Each of the other panels was created using the scattering center model from (7.4) and (7.5) and a different set of randomly generated parameters.

Because \hat{i}_m is a unit vector, we have $\max_{\hat{i}_i} \hat{i}_m \cdot \hat{i}_i = 1$. The same applies for \hat{i}_s . Thus, each \hat{i}_m defines the direction in which the corresponding scattering center is maximally visible. Accordingly, each σ_m^2 determines the angular extent about \hat{i}_m in which the m^{th} scattering center will contribute significantly to E^s . We refer to the two exponential terms in (7.5) as the incident and observed *visibility factors*. Because of them, individual scattering centers will blink in and out as the target's aspect with respect to the transmitter and receiver changes. The term a_m in (7.5) is the typical complex scattering center amplitude found in [77]. With this modification, E^s for a given target class is completely defined by N_{sc} and $\{\mathbf{r}_m, a_m, \hat{i}_m, \sigma_m^2 : m = 1, \dots, N_{sc}\}$. The synthetic RCS that we will use in our simulations is then simply $\sigma = |E^s|^2$, where E^s is computed using (7.4) and (7.5).

For our simulations, we will need seven different target classes. Other than N_{sc} , all parameters will be generated randomly for each class. The complex amplitudes $\{a_m\}$ will be drawn uniformly from a finite range in order to upper-bound the resulting RCS. Likewise, the locations of the scatterers $\{\mathbf{r}_m\}$ will be drawn from a uniform distribution over the volume occupied by a typical fighter-sized aircraft. However, to make our synthetic RCS as realistic as possible, we will force lateral symmetry among the collection of scattering centers. This implies that only $N_{sc}/2$ of the scatterers are independent; the others are reflections of the first half through the $\hat{b}_x \hat{b}_z$ -plane.⁵

Figure 7.3 shows the bistatic RCS in the zero-degree elevation plane for the seven classes we will need. The scattering center models each use between 12 and 18 scatterers. As a means of checking the quality of our approximation, the actual RCS of a VFY-218 fighter (as generated by FISC) is shown in the upper left panel. Although

⁵More specifically, the vectors \mathbf{r}_m and \hat{i}_m are reflected through the $\hat{b}_x \hat{b}_z$ -plane. The parameters a_m and σ_m^2 remain equal to their reflected equivalents.

some of the models are clearly more “RCS-like” than others, they all suffer from a couple shortcomings. First, the forward scatter phenomenon (the two bright red lines in the VFY-218 plot) is missing from the synthetic models. However, because neither of the maneuvers occurs between any of the transmitters and the receiver, this will not impact our simulations. Second, the synthetic RCS values for approximately nose-on geometries are sometimes too large. However, because we do not have a monostatic geometry, the target is never simultaneously illuminated and observed nose-on.

In summary, even though the assumptions of the scattering center model are inappropriate in the VHF band for fighter-sized aircraft, we find that the resulting RCS plots have similar enough fine structure for use in our simulations. In the following single-target experiments, scattering center (SC) model 1 will be used to generate the necessary RCS measurements.

7.2 Tracking Results for a Single Target

In this section, we begin by considering the simplest scenario: tracking a single target of known identity in a clutter-free environment with $P_D = 1$. These simulations will allow us to determine the fundamental tracking capability of each system. Unfortunately, it is not as simple as running a single set of trials for each tracker. Both systems have several free parameters that must be adjusted for optimal performance. In the following discussion we will explore the effects of these free parameters on system performance.

7.2.1 Tracking performance of the IMM extended Kalman filter

We first consider the tracking performance of our EKF tracker. Because our EKF is implemented using the Interacting Multiple Model (IMM) algorithm, there are three sets of parameters that must be specified. First, we must decide how many models will be used. As discussed in Section 5.1, we follow popular practice and implement our filter using a constant velocity (CV) model, a constant acceleration (CA) model, and coordinated turn (CT) model [40, 64]. Second, we must specify the Markov transition matrix for the IMM algorithm. Because $\lim_{\omega \rightarrow 0} F_k^{(CT)}(\omega) = F_k^{(CA)}$, the IMM coordinated turn model can also serve as a constant acceleration model. The IMM-CA model is instead used to model the brief, but substantial, accelerations present at the beginning and end of maneuvers. In this case, we adopt the transition matrix

$$P_{IMM} = \begin{bmatrix} 0.937 & 0.063 & 0 \\ 0.125 & 0.75 & 0.125 \\ 0 & 0.045 & 0.955 \end{bmatrix}, \quad (7.6)$$

where the index-to-model mapping is $1 \rightarrow CV$, $2 \rightarrow CA$, and $3 \rightarrow CT$. The dwell probability of the IMM-CV model is chosen to match P_{11} from (7.1). The dwell probability of the IMM-CA model is chosen to yield an expected dwell time of 1.2 s. Finally, the dwell probability of the IMM-CT model is selected to yield an expected dwell time

Table 7.2: EKF Track Accuracy versus IMM Process Noise Variances
 Roll-Roll Maneuver Roll-Pitch Maneuver

$q^{(CV)}$	$q^{(CA)}$	$q^{(CT)}$	e_{RMS} (m)	N_L	e_{RMS} (m)	N_L
3	60	3	136.80	6	146.93	5
		9	139.26	2	149.74	3
		15	148.21	8	154.90	8
3	80	3	132.09	3	140.13	5
		9	140.94	4	144.01	8
		15	140.71	5	154.60	3
3	100	3	138.68	3	138.65	7
		9	143.47	3	146.65	3
		15	137.89	2	140.37	3
3	120	3	136.96	3	153.61	5
		9	155.52	5	142.22	9
		15	135.33	3	146.95	2

of 8.5 s. The third and final set of parameters that must be specified are the process noise variances $\{q^{(CV)}, q^{(CA)}, q^{(CT)}\}$ for the three models. As mentioned in [64], the choice of these parameters can be considered a tuning process best determined empirically.

In order to find good values for $\{q^{(CV)}, q^{(CA)}, q^{(CT)}\}$, we will run 100 Monte Carlo simulations for each maneuver in Figure 7.1 and various choices of the three noise variances. The number of combinations that must be considered can be limited by noting that the standard deviation of each IMM model should be on the order of the maximum change in acceleration during a sample period in which the target operates in the mode specified by the model. Thus, because the IMM-CV model corresponds to nominally zero acceleration, $q^{(CV)}$ can be expected to be relatively small. On the other hand, because the IMM-CA model corresponds to maneuver transients, $q^{(CA)}$ will need to be substantially larger. Finally, because the centripetal acceleration of a coordinated turn is already incorporated in the structure of $F_k^{(CT)}(\omega)$, $q^{(CT)}$ does not need to be very large. However, because the IMM-CT model must also match sustained longitudinal accelerations such as the pitch/thrust portion of the roll-pitch maneuver, a good choice for $q^{(CT)}$ will be a trade-off.

Table 7.2 displays the results of 12 different process noise combinations. Overall, we find that the IMM-EKF performs better against the roll-roll maneuver than the roll-pitch maneuver. This is not surprising because the IMM-EKF includes a coordinated turn model. However, because the EKF cannot model the true dynamics of air flight, it does not perform as well against a climbing or descending target such as during the pitch/thrust portion of the roll-pitch maneuver. In terms of the IMM noise variances, the best choice of $q^{(CT)}$ seems to vary directly with $q^{(CA)}$. As $q^{(CA)}$ grows, it becomes an increasingly poor match to any portion of either maneuver, and $q^{(CT)}$ must increase proportionately in order to serve as both a coordinated turn and constant acceleration model. Because we want the filter models to operate as designed, we select

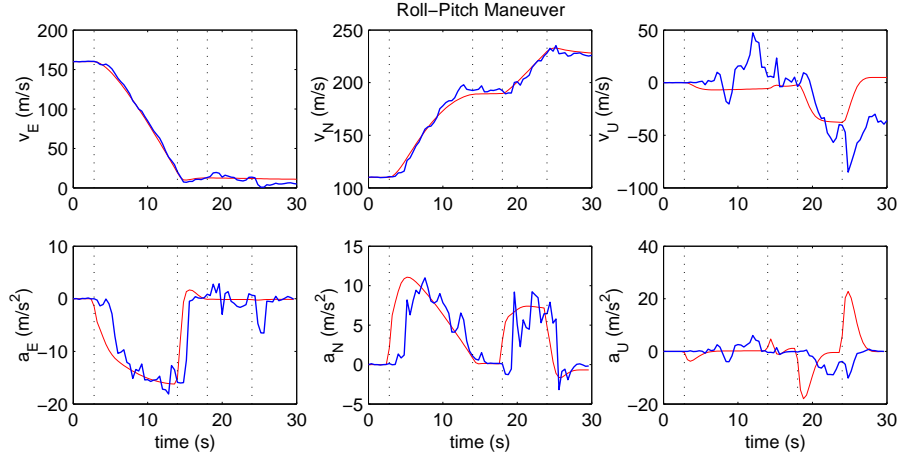


Figure 7.4: Example of tracking performance of the IMM-EKF during the roll-pitch maneuver. Red lines are true values; blue lines are IMM-EKF estimates. Vertical black dotted lines indicate changes in the target’s flight mode.

the combination $q^{(CV)} = 3$, $q^{(CA)} = 80$, and $q^{(CT)} = 3$ as providing the best overall tracking performance for both maneuvers. We also note that the number of lost tracks is relatively small for this choice. From here forward, all results for the IMM-EKF will be given with respect to this choice of noise variances. It should also be mentioned that the computational complexity of the IMM-EKF is largely independent of the values selected for $q^{(CV)}$, $q^{(CA)}$, and $q^{(CT)}$. The average run-time for the filter was approximately 4 s for all of the combinations listed in Table 7.2.

To understand the shortcomings of the EKF-IMM against the types of maneuvers we are considering, it is helpful to look at the results from a single Monte Carlo run. Figure 7.4 shows the filtered estimates of target velocity and acceleration from one of the 100 roll-pitch experiments conducted for the optimal combination of IMM noise variances selected above. The top row depicts the components of target velocity in a receiver-centered Earth-fixed system (from left to right, the component directions are east, north, and up). The same applies to the bottom row, which depicts target acceleration. The vertical dotted lines indicate transitions in the flight mode of the target. Referring back to Figure 7.1, we see that the roll-pitch maneuver consists of the following sequence of modes: CV, CT, CV, PT, CV.

Because FM-band passive radar provides high accuracy measurements of Doppler shift, we expect the IMM-EKF estimate of velocity to be very good. While this is clearly true of v_E and v_N in Figure 7.4, the filtered estimate of v_U , the vertical velocity of the target, is quite poor. To understand why this is so, we refer back to (5.14), which defines the relationship between the velocity vector and Doppler shift. We see that the observability of the velocity vector during a given scan depends critically on the orientation of the surface normal $\hat{n}_{t_k}(\mathbf{p}_k)$ defined in (5.15). For both trajectories, because the altitude of the target is relatively small compared to the distance to the receiver or transmitters, $\hat{n}_{t_k}(\mathbf{p}_k)$ will be nearly parallel to the ground plane for all k . This means

that changes in vertical velocity will be more difficult to detect through Doppler shift than corresponding changes in horizontal velocity (i.e., v_E or v_N). This is precisely what we see in Figure 7.4. The solution to this shortcoming is to track changes in target pitch, which *cause* changes in vertical velocity. Thus, although changes in v_U will be similarly difficult to detect through Doppler shift for a particle filter, changes in pitch *will* be detectable via RCS. Coupled with an aerodynamically valid flight model, we will find that our particle filter provides superior estimation of v_U .

The bottom row in Figure 7.4 verifies that the magnitudes of the IMM noise variances are adequate to track the true acceleration curves. However, we note a significant lag in the filter’s estimates of a_E and a_N at the onset of the target roll (the second segment of the trajectory). This is not surprising because the IMM-EKF only measures target velocity (through Doppler shift). Thus, with the onset of a new force acting upon the target, the resulting change in acceleration goes undetected until the target’s horizontal velocity changes considerably. In the next section, we will find that the change in RCS accompanying a target roll allows the particle filter to track changes in acceleration more quickly. We defer analysis of the IMM model probabilities until the next section, where they can be directly compared to the particle filter’s mode probabilities.

7.2.2 Tracking performance of the particle filters

In this section, we quantify the tracking performance of both a generic particle filter (PF) that uses the prior as proposal distribution and the auxiliary particle filter (APF) described in Section 4.3.4. As with the IMM-EKF, we will perform 100 Monte Carlo trials for each of the maneuvers in Figure 7.1. The process noise parameters for the particle filter state model were already presented in Table 7.1. However, because the particle filter can incorporate RCS, we now need to specify the variance of the Gaussian noise added to the in-phase and quadrature components of the scattering coefficient. For now, we take $\sigma_{n_k}^2 = 8.0 \text{ m}^2$ for all k and postpone discussion about the validity of this assumption until the next section. The only parameter left to specify then is N_p , the number of particles. Similar to the IMM noise variances, this can be considered a tuning parameter that is best determined empirically. Because computational complexity scales linearly with N_p , we wish to find the smallest value that yields good tracking performance.

Table 7.3 contains results quantifying the tracking performance of both the generic particle filter and the auxiliary particle filter for many different choices of N_p . Four comments are in order. First, as expected, both the RMS error and the number of lost tracks decrease as N_p increases. There are a few entries in the table where this is not the case, but these can be attributed to the variation inherent in Monte Carlo simulations. Second, we find that the auxiliary particle filter meets or exceeds the tracking accuracy of the generic particle filter for both maneuvers on all but a single value of N_p ($N_p = 200$ for the roll-roll maneuver). This is not surprising either because the generic particle filter uses the prior as proposal distribution, $\pi(\mathbf{x}_k | \mathbf{x}_{k-1}^{(i)}, \mathbf{z}_k) = p(\mathbf{x}_k | \mathbf{x}_{k-1}^{(i)})$. As such, it operates without any knowledge of the current measurement. The auxiliary

Table 7.3: PF Tracking Accuracy versus Particle Count
 Roll-Roll Maneuver Roll-Pitch Maneuver

	N_p	e_{RMS} (m)	N_L	e_{RMS} (m)	N_L
Generic Particle Filter	200	92.64	0	122.93	7
	300	92.03	0	114.53	4
	400	91.80	0	107.94	1
	600	86.78	0	92.31	0
	800	80.70	0	97.84	0
Auxiliary Particle Filter	200	94.91	0	112.06	5
	300	86.92	0	107.12	3
	400	80.25	0	91.77	3
	600	86.47	0	92.10	0
	800	76.61	0	89.93	0

particle filter, on the other hand, takes $\pi(\mathbf{x}_k, i | \mathbf{z}_{1:k}) \propto p(\mathbf{z}_k | \lambda_k^{(i)})p(\mathbf{x}_k | \mathbf{x}_{k-1}^{(i)})w_{k-1}^{(i)}$, and is thus able to “steer” its particles towards promising areas of the state space based upon \mathbf{z}_k (see Section 4.3.4). A third observation from Table 7.3 is that the roll-pitch maneuver is more difficult to track than the roll-roll maneuver. This can be understood by referring back to the command variable ranges in Table 7.1. We see that the PT and CT modes both have two command variables (i.e., two variables that largely determine the flight path until the next mode transition). However, closer inspection shows that the command variable controlling the CT pitch increase has a very small range. (Otherwise, it would be a climbing or descending turn.) In essence, a particle filter primarily needs to accurately determine ω_k^* , the turn-rate, at the onset of a coordinated turn; the precise value of the CT pitch increase has little effect on the resulting trajectory. On the other hand, the pitch change for the PT mode can be considerable, and therefore, both of the command variables for the PT mode must be accurately estimated. In simple terms, there is more variability in the pitch/thrust mode than the coordinated turn. This is compounded by the fact that the change in target aspect accompanying the CV \rightarrow PT transition is not nearly as substantial as that of the CV \rightarrow CT transition. Thus, RCS is typically a better indicator of the onset of a coordinated turn than a climb or descent. The fourth and final observation to be made from Table 7.3 concerns its comparison with the IMM-EKF error statistics in Table 7.2. We find that both particle filters yield at least a 33% reduction in RMS error on both maneuvers. In addition, for all but $N_p = 200$ on the roll-pitch maneuver, they also result in fewer lost tracks.

Upon first thought, it might not seem surprising that both particle filters outperform the IMM-EKF; after all, they use the same motion model as the simulated trajectories. Two comments should be made, though. First, as depicted in Table 7.1, the extent of the state space in which the particle filters operate is considerably larger than the extent of the state space in which the roll-roll and roll-pitch maneuvers exist. Second, and more importantly, we recall that the motion model was designed, first and foremost, to produce aerodynamically valid trajectories. Thus, it is not that the data has been fashioned to work well with a particle filter. Rather, the flexibility of the sequential

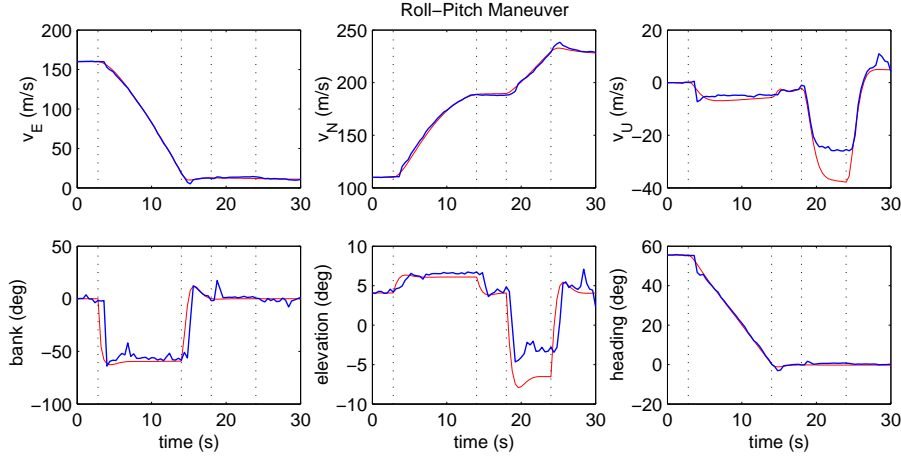


Figure 7.5: Example of the tracking performance of the generic particle filter with $N_p = 600$. Red lines are true values; blue lines are particle filter estimates. Vertical black dotted lines indicate changes in the target’s flight mode.

importance sampling framework has allowed us to implement a valid state model for flight.

Figure 7.5 depicts several estimates produced by the generic particle filter with $N_p = 600$ for one of its roll-pitch Monte Carlo trials. The top row is target velocity in receiver-centered east-north-up coordinates. The bottom row is the Euler angles needed to bring the inertial frame into coincidence with the body-centered frame. Comparing the top rows of Figures 7.4 and 7.5, we find that the particle filter is equally adept as the IMM-EKF at tracking v_E and v_N . However, the particle filter is noticeably better at tracking v_U , the target’s vertical velocity. Recall, this component of the velocity vector is typically much more difficult to measure through Doppler shift than horizontal velocity. Inclusion of RCS in the measurement vector allows the particle filter to detect the change in pitch, which *causes* the change in v_U . In the roll-pitch maneuver, a sharp decrease in pitch at 18 s results in a decrease in the angle of attack, which reduces the magnitude of the lift vector. The natural result: the target begins to descend. We note that the estimation error in v_U during the target’s descent (from $t = 18$ – 24 s) is caused by the 4 – 5° error in the filter’s pitch estimate during the same period. Depending on many factors, such as target geometry and orientation with respect to the transmitters and receiver, a discrepancy of such small magnitude may not be detectable through RCS.

We have identified one of the reasons for the particle filter’s superior performance: more accurate estimation of vertical velocity. Now, we highlight a second reason for the improvement in tracking accuracy. In Figure 7.6, we display examples of both the IMM model probabilities and the particle filter mode probabilities for the roll-roll and roll-pitch maneuvers. It is important to remember that models are labeled based upon the flight mode they are intended to represent. The underlying models are not the same, though. For example, even though the IMM-EKF and PF both have CV

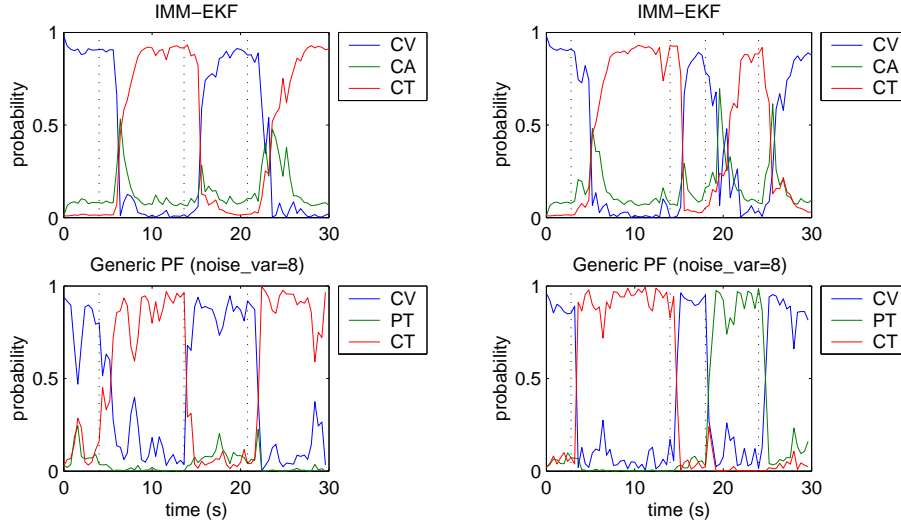


Figure 7.6: Example of mode probabilities for the IMM-EKF and the generic particle filter ($N_p = 600$). Left column: roll-roll maneuver. Right column: roll-pitch maneuver. Vertical black dotted lines indicate changes in the target’s flight mode.

Table 7.4: Transition Times and Mode Sequences for Both Trajectories

Roll-Roll	Time (s)	0	4.0	13.6	20.8	-
	Mode	CV	CT	CV	CT	-
Roll-Pitch	Time (s)	0	2.8	14.0	18.0	24.0
	Mode	CV	CT	CV	PT	CV

models for tracking constant velocity flight, these models differ in significant ways. The particle filter’s CV model requires that the bank angle be zero. The CV model for the IMM-EKF, on the other hand, places no requirements on bank angle because the EKF has no practical way of measuring target orientation. The same remark applies to the coordinated turn models of the IMM-EKF and PF.

In Figure 7.6, the vertical black dotted lines indicate times at which mode changes occur. To assist our analysis, the correct mode sequence and transition times are listed in Table 7.4. Comparing the top row of Figure 7.6 with the entries in Table 7.4, we find that the IMM models are operating as designed for both trajectories. First, all constant velocity segments are correctly detected. Second, the IMM-CA model (whose variance we tuned in Section 7.2.1) only has significant probability during mode transitions, as designed. Finally, the IMM-CT model is the most probable during both coordinated turns and climbs/descents (e.g., the segment beginning at $t = 18$ s of the roll-pitch maneuver). This is to be expected because the IMM-CT model reduces to a constant acceleration model as ω approaches zero.

Comparing the bottom row of Figure 7.6 with the entries in Table 7.4, we find that the particle filter is also performing as designed. Notably, because the particle filter incorporates a pitch/thrust (PT) mode for climbs and descents, the PT segment of the

Table 7.5: Average Run-Time for the Roll-Roll Maneuver versus Particle Count

N_p	Generic PF Time (s)	Auxiliary PF Time (s)
200	129.8	264.9
300	201.0	386.8
400	267.0	515.2
600	400.9	777.1
800	535.1	1036.4

roll-pitch maneuver is correctly identified. While Figure 7.6 shows that both filters are operating properly, it clearly identifies a second reason for the particle filter’s superior performance: time lag before a mode transition is detected. Comparing the delays between transitions in Table 7.4 (the black dotted lines in the plots) and changes in each filter’s mode probabilities, we find that the particle filter is substantially faster than the IMM-EKF at detecting the onsets of new flight modes. Two specific examples of this are the CT \rightarrow CV transition at 13.6 s of the roll-roll maneuver and the CV \rightarrow CT transition at 2.8 s of the roll-pitch maneuver. While the particle filter should detect transitions involving the PT mode more quickly (because the IMM-EKF does not have a legitimate pitch/thrust model), the two examples we have highlighted involve flight modes for which the IMM-EKF has been optimized.

The reason for this discrepancy in performance between the IMM-EKF and the PF is easy to understand. A change in target orientation alters the net force on the target and eventually results in a change in velocity. Because the IMM-EKF cannot incorporate RCS measurements, it must detect changes in flight mode through changes in target velocity. As such, the IMM-EKF is “one integration” removed from the source of the mode transition. The particle filter, on the other hand, is able to use RCS to detect the changes in target aspect that are present at the very beginning of a new flight mode. This translates into faster mode detection, which provides superior estimation accuracy.

So far in this section, we have evaluated the tracking performance of both a generic particle filter and an auxiliary particle filter. We have found that both types outperform the IMM-EKF. The two main reasons for the superior performance are better estimation of vertical velocity and faster mode detection following transitions. Both improvements can be attributed to the particle filter’s ability to incorporate RCS as a data feature, which permits the use of a aerodynamically valid state model. Of course, engineering is full of trade-offs, and the price we pay for the flexibility of the sequential importance sampling framework is a substantial increase in computation. Table 7.5 lists the average times per trial as a function of N_p on the roll-roll maneuver. As expected, the run-times of both PF algorithms scale linearly with N_p . We note that the auxiliary particle filter requires, on average, just under twice as much time as the generic particle filter. If we revisit the pseudo-code description of the APF algorithm in Figure 4.7, we find that each iteration of the APF is actually equivalent to *two* complete iterations of the generic particle filter. However, incorporation of the current measurement in

the proposal distribution yields particle weights that are more uniform. Because resampling then occurs less frequently (i.e., after the second measurement update), this explains why the APF is slightly less than twice as slow as the generic PF. Finally, we recall that the IMM-EKF had an average run-time of approximately 4 s per trial. The generic particle filter with $N_p = 600$ (whose results are displayed in Figure 7.5) requires 400 s per run — two orders of magnitude greater than the IMM-EKF!

Two comments need to be made on this point. First, we must remember that the particle filter is actually doing two things simultaneously: tracking *and* classifying. Even though we have only discussed tracking performance for a single known target, the filter would operate exactly the same if the target class was unknown.⁶ Second, because of the substantial difference in computation, it might seem that our performance comparisons would be more fair if we included additional models in our IMM-EKF and thereby narrowed the run-time gap. Unfortunately, this is unlikely to lead to better performance for the IMM-EKF. Simply put, the EKF cannot use the correct motion model because it has no way of measuring target aspect. Including additional mismatched models is unlikely to correct for this fundamental deficiency.

Choice of scattering coefficient noise variance

In this section, we briefly address our choice for the variance of the additive Gaussian noise that affects the measurement of the in-phase and quadrature components of the scattering coefficient. All of the experimental results presented so far have taken $\sigma_n^2 = 8 \text{ m}^2$. Using Equations (6.22) and (6.23), the noncentral chi-squared distribution for RCS then has an expected value of $|s_k|^2 + 16$ and a variance of $32(|s_k|^2 + 8)$. For large values of RCS, the measurement error can clearly be substantial. The question then becomes, “Is this a realistic value for σ_n^2 ?” Unfortunately, there is no easy answer for this question. Because current passive radars do not attempt to perform joint tracking and classification, they have no need to extract RCS as a data feature. As such, the current literature has little to offer on this matter. Instead, we will investigate the influence of σ_n^2 on the tracking performance of the generic particle filter. If it happens that the filter is not especially sensitive to the value of σ_n^2 , then the exact value that is used in our experiments becomes less important.

In Table 7.6, tracking results for the generic particle filter (with $N_p = 600$) are presented for the roll-roll and roll-pitch trajectories using several different values of σ_n^2 . As usual, 100 Monte Carlo trials were run for each entry in the table. For the roll-roll maneuver, we find that the tracking performance does not seem to vary greatly with the value of σ_n^2 . This occurs because, as the RCS measurements become increasingly noisy, the particle filter can still fall back upon delay and Doppler data for tracking. The entries for the roll-pitch maneuver in Table 7.6 exhibit a strange trend, though. As σ_n^2 increases, tracking performance initially *improves*! After $\sigma_n^2 = 8$, performance then

⁶In terms of code, the known-identity scenario is created by initializing all particles with the same (correct) target class. If the target class was unknown, the particles would be initialized with equal numbers devoted to each potential class. From an algorithmic perspective, there is no difference between the two cases.

Table 7.6: Generic PF Tracking Accuracy versus σ_n^2
 Roll-Roll Maneuver Roll-Pitch Maneuver

σ_n^2	e_{RMS} (m)	N_L	e_{RMS} (m)	N_L
2	85.85	0	110.26	7
4	89.87	0	102.50	0
8	86.78	0	92.31	0
16	79.39	0	94.21	0
32	86.02	0	99.29	1

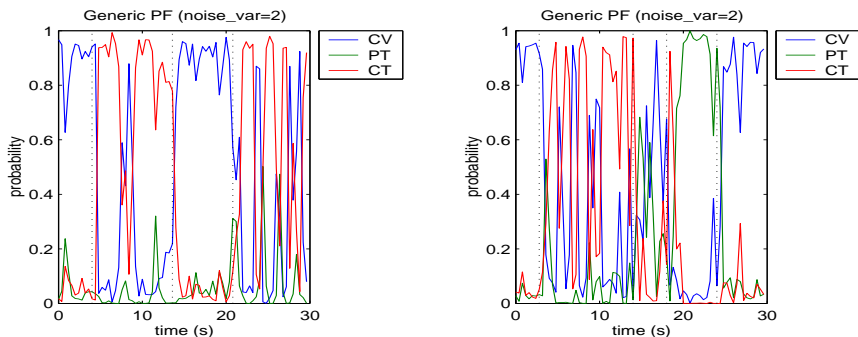


Figure 7.7: Example of the mode probabilities for the generic particle filter with $N_p = 600$ and $\sigma_n^2 = 2$. Left plot: roll-roll maneuver. Right plot: roll-pitch maneuver.

begins to degrade, as expected. To understand why less noise could actually degrade track accuracy, we plot the mode probabilities from one of the $\sigma_n^2 = 2$ simulations in Figure 7.7.

Comparing these two plots with those from the bottom row of Figure 7.6, we immediately notice a difference. In the second CT segment of the roll-roll maneuver (20.8–30 s) and the only CT segment of the roll-pitch maneuver (2.8–14.0 s), the particle filter with $\sigma_n^2 = 2$ incorrectly “chatters” between the CT and CV modes. This occurs because the low value of σ_n^2 favors trajectories that, first and foremost, match the RCS measurements, even if they are unrealistic from a kinematic perspective. For example, during a turn, it could be that a noisy RCS measurement happens to match CV particles (whose bank angles are zero) better than CT particles (whose bank angles are correct). This causes the CV flight mode to increase in probability. A few samples later, when it becomes clear through RCS or Doppler that the target is still turning, the CT mode will return to prominence. This can occur because N_p , the PF sample size, is finite. As such, the limited number of CT mode particles may provide inadequate coverage of the range of possible bank angles. Because we want the particle filter to operate as intended (i.e., without chattering between modes), we will avoid small values for σ_n^2 . Instead, we select $\sigma_n^2 = 8$ as a reasonable value that minimizes occurrence of the phenomenon seen in Figure 7.7 while still providing RCS measurements with enough fidelity to be useful for tracking and classification.

Table 7.7: Generic PF Tracking Accuracy versus Scattering Center Model

SC Model	Roll-Roll Maneuver		Roll-Pitch Maneuver	
	e_{RMS} (m)	N_L	e_{RMS} (m)	N_L
1	86.78	0	92.31	0
2	81.06	0	93.92	0
3	82.25	1	100.84	2
4	85.13	0	106.23	0
5	84.99	1	102.34	0
6	87.11	1	101.15	0
7	83.53	0	102.02	0

Choice of scattering center model

We conclude our experiments for a single target of known identity by investigating the effect of our choice of scattering center model on tracking performance. Figure 7.3 shows the bistatic RCS for seven different scattering center models with randomly generated parameters. In this section, we wish to make sure that the superior tracking performance displayed by the particle filter is not caused by a fortuitous choice of scattering center model parameters. Table 7.7 lists the results of 100 trials for each combination of trajectory and scattering center model, run using the generic particle filter with $N_p = 600$ and $\sigma_n^2 = 8$. For a given maneuver (roll-roll or roll-pitch), the target follows the same flight path and uses the same kinematic coefficients. The only difference is the parameters that are used to synthesize the RCS measurements. Comparing the entries in the table, we find that, while SC 1 (the model that we have used in previous sections) performs quite well on the roll-pitch maneuver, its performance on the roll-roll maneuver is actually below average. More importantly, comparing the results in Tables 7.2 and 7.7, we note that the generic particle filter outperforms the IMM-EKF, regardless of the choice of scattering center model. Thus, the improvement in performance achieved by the particle filter is indeed related more to its aerodynamically correct state model than the specific nature of the target's radar cross section.

7.3 Tracking Results for a Single Target in Clutter

In the previous section, we began the empirical analysis of our particle filtering algorithm using the simplest possible scenario: tracking a single target of known identity. Because we did not allow false alarms or missed detections, data association was not an issue. In this section, we consider the more challenging task of tracking a single target of known identity in the presence of clutter (i.e., false alarms) and missed detections. By continuing to assume that the identity of the target is known, any performance degradation (compared to the results from the previous section) can then be attributed to clutter and $P_D < 1$. As before, we will evaluate the performance of our IMM-EKF, generic PF, and auxiliary PF by Monte Carlo simulation.

Before we can present the experimental results, there is a technical difficulty that

must be addressed. As discussed in Sections 5.3 and 6.3, we follow popular practice and assume that the number of false alarms in scan k is Poisson-distributed with mean βV_k , where β is the clutter density and V_k is the scan volume. In the previous section, filter performance was quantified using two figures of merit: the RMS error e_{RMS} and the lost track count N_L . These metrics could be compared between filters because the simulations occurred under identical circumstances (namely, the same two flight paths). In this section, a fair comparison is much more difficult to achieve. To see why this is so, we note that the IMM-EKF operates in a two-dimensional measurement space (delay and Doppler). On the other hand, because the particle filter incorporates RCS as a data feature, it operates in a three-dimensional measurement space. As such, it is not possible to simply use the same value of clutter density to insure a fair comparison. Instead, we need a different metric to quantify the extent of clutter for each trial.

For our IMM-EKF, the measurement $\mathbf{z}_k^{(j)}$ satisfies the gate condition for a given target track if

$$\mathbf{y}_k^{(j)T} S_k^{-1} \mathbf{y}_k^{(j)} < 2 \ln \left(\frac{P_D}{\beta(1 - P_D P_G) 2\pi \sqrt{|S_k|}} \right), \quad (7.7)$$

where $\mathbf{y}_k^{(j)}$ is the innovation, $\mathbf{y}_k^{(j)} = \mathbf{z}_k^{(j)} - h_k(\hat{\mathbf{x}}_{k|k-1})$, S_k is its covariance matrix, and β is the clutter density. Equation (7.7) is derived by requiring that $\Pr(\Gamma_k^{(j)} | \mathbf{z}_{1:k}) > \Pr(\Gamma_k^{(0)} | \mathbf{z}_{1:k})$ for $j \neq 0$, using the association probabilities from (5.32). In words, a measurement that is more likely to have been produced by the target than a false alarm is said to satisfy the gate condition. Any measurement that does not satisfy at least one target gate is discarded prior to data association.⁷ For simulations such as ours in which the number of targets is known *a priori*, ungated false alarms have no effect on track accuracy. As such, the average number of gated measurements can be used in place of β to quantify the extent of clutter. We define this average as

$$\bar{N}_g^{EKF} \triangleq \frac{\text{total \# of gated measurements}}{(N_k - 1)N_{MC}}, \quad (7.8)$$

where N_k is the number of scans, and N_{MC} is the number of Monte Carlo trials. The number of scans is $(N_k - 1)$ instead of N_k because there is no measurement update for the first scan. (It is used to initialize the filter.)

While \bar{N}_g^{EKF} would seem to provide a means of ensuring that the simulations for each filter were conducted under similar circumstances, the need to estimate the covariance S_k for (7.7) is problematic for the particle filter. First, we recall that one of the main advantages of the particle filtering approach is the ability to model highly non-Gaussian densities. Using a single statistic, such as the covariance matrix S_k , to summarize the extent of a potentially multimodal distribution negates much of this benefit. Second, if degeneracy of the sample weights results in frequent resampling, the particle set may become impoverished (i.e., may contain few unique values). In this case, the estimate of S_k will be inaccurate, which could lead to further degradation if

⁷Actually, in a real implementation, these ungated measurements would be used to initiate new tracks.

the resulting gate becomes too small to include the true measurement. For these reasons, particle filters often avoid gating altogether [65]. However, in order to have some basis on which to compare the performance of the IMM-EKF and the particle filter, we will perform a crude type of gating based upon (6.30). In that equation, the likelihood function in the presence of clutter and missed detections was shown to satisfy

$$p(\mathbf{z}_k | \mathbf{x}_k^{(i)}) \propto (1 - P_D P_G) \beta + P_D \sum_{j=1}^{m_k} p(\mathbf{z}_k^{(j)} | \mathbf{x}_k^{(i)}), \quad (7.9)$$

where m_k is the number of measurements received during scan k . Thus, we see that any measurement $\mathbf{z}_k^{(j)}$ whose likelihood is substantially less than $\beta(1 - P_D)/P_D$ will make a negligible contribution to $w_k^{(i)}$. With this as motivation, a measurement $\mathbf{z}_k^{(j)}$ will be said to satisfy the gate condition for particle $\mathbf{x}_k^{(i)}$ if

$$p(\mathbf{z}_k^{(j)} | \mathbf{x}_k^{(i)}) > g \frac{\beta(1 - P_D P_G)}{P_D}, \quad (7.10)$$

where $g = 0.02$ in our experiments. Only those measurements that satisfy (7.10) will be used in (7.9) to update the particle weight $w_k^{(i)}$. This gate definition has two attractive properties for particle filters. First, it does not rely upon an estimate of the state covariance. Second, the gate threshold varies directly with β . Thus, as the number of false alarms increases, so will the gate threshold. With (7.10) as the gate condition for our particle filter, we can now define \bar{N}_g^{PF} . Because the gate is applied individually for each particle, \bar{N}_g^{PF} must be averaged over the entire particle set, as well as each scan and Monte Carlo trial,

$$\bar{N}_g^{PF} \triangleq \frac{\text{total \# of gated measurements}}{N_p(N_k - 1)N_{MC}}, \quad (7.11)$$

where measurements can be counted multiple times if they satisfy (7.10) for more than one particle.

For the rest of this section, tracking performance will be quantified using e_{RMS} , N_L , and either \bar{N}_g^{EKF} or \bar{N}_g^{PF} for various choices of P_D and β . As a reminder of the difference in dimension of the underlying measurement space, we will use β_2 to denote the (two-dimensional) clutter density for the IMM-EKF and β_3 to denote the (three-dimensional) density for the particle filter.⁸ The average gate counts will allow us to compare the extent of clutter between systems. However, because the gate conditions for \bar{N}_g^{EKF} and \bar{N}_g^{PF} are different, this is not a strict equivalence.

7.3.1 Results for the IMM extended Kalman filter

Our IMM-EKF simulations were performed using the optimal set of IMM noise variances, as determined in Section 7.2.1. We considered two values for the probability of detection: $P_D = 0.9$ and $P_D = 0.7$. Table 7.8 lists the tracking results for several different clutter densities using the roll-roll maneuver. Note that for $\beta_2 = 0$ and

⁸ β_2 is unitless; β_3 has units m^{-2} .

Table 7.8: IMM-EKF Tracking Accuracy versus β_2 and P_D

β_2	$P_D = 0.9$			$P_D = 0.7$		
	e_{RMS} (m)	N_L	\bar{N}_g^{EKF}	e_{RMS} (m)	N_L	\bar{N}_g^{EKF}
0	157.79	9	0.91	210.62	34	0.73
25	167.02	6	0.93	189.40	38	0.73
50	167.63	17	0.95	223.54	44	0.74
100	175.99	24	1.00	191.93	56	0.77
200	162.54	30	1.05	212.01	76	0.81

$P_D \neq 1$, some of the scans will be empty (i.e., will not contain any measurements). In these cases, the EKF state and covariance matrices are predicted to the current time, but there is no measurement update. The results in Table 7.8 should be compared to the single-target results from Table 7.2. In those simulations, the optimal choice of IMM noise variances yielded $e_{RMS} = 132.09$ m and $N_L = 3$ for the roll-roll maneuver. When compared to the results in Table 7.8, the negative impact of clutter and missed detections is apparent. Because the average number of false measurements per gate is approximately equal to $\bar{N}_g^{EKF} - P_D$, we find that performance can be degraded by just a few false associations. As an example, consider the $(\beta_2, P_D) = (200, 0.9)$ simulations. In this case, false alarms contained in $1.05 - 0.9 = 15\%$ of the gates and a 10% chance of missed detection were enough to cause 30 out of 100 tracks to be lost.⁹

Overall, there are two conclusions that can be drawn from Table 7.8 for our application. First, missed detections are more damaging to track accuracy than false alarms. This can be seen by comparing the first row of the table ($\beta_2 = 0$) to the others. We find that e_{RMS} degrades immediately for $P_D \neq 1$. Increasing β_2 , on the other hand, mostly affects N_L , the number of lost tracks. For those cases where e_{RMS} actually *decreases* as β_2 grows, we refer back to the definition of e_{RMS} in (7.3). We see that e_{RMS} is averaged over $N_{MC} - N_L$ trials. Thus, as the number of lost tracks grows with β_2 , so will the Monte Carlo variation in e_{RMS} . Second, a modest change in P_D can have severe consequences. While the IMM-EKF still functions for $P_D = 0.9$ and small values of β_2 , it is almost unusable for $P_D = 0.7$, regardless of the choice of β_2 . To understand why this is so, consider the track loss example shown in Figure 7.8. In the plot, the red line represents the true roll-roll trajectory; the blue line is the IMM-EKF estimate. In this example, missed detections at the crucial onset of the first coordinated turn causes the filter to miss the maneuver entirely. Instead, the filter locks onto a series of false alarms that match its constant velocity mode. This demonstrates that the target trajectory can have a substantial impact on filter performance in the presence of clutter and missed detections. Thus, the results in Table 7.8 are not necessarily indicative of the performance that could be achieved against a nonmaneuvering target.

⁹This is just an approximation because the true measurement, when detected, may not always satisfy the gate condition. There can also be more than one false alarm per gate.

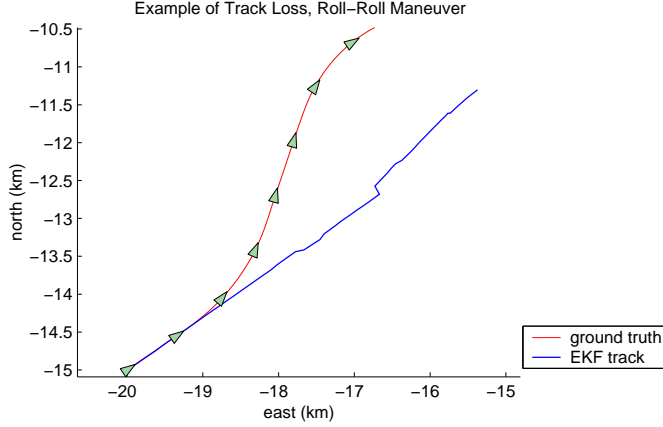


Figure 7.8: Example of track loss for IMM-EKF due to clutter and missed detections. For the trial shown, $\beta_2 = 100$ and $P_D = 0.7$.

Table 7.9: PF Tracking Accuracy versus β_3 and P_D

	β_3	$P_D = 0.9$			$P_D = 0.7$		
		e_{RMS} (m)	N_L	\bar{N}_q^{PF}	e_{RMS} (m)	N_L	\bar{N}_q^{PF}
Generic Particle Filter $N_p = 600$	0	80.76	3	0.87	96.48	14	0.66
	2	88.58	2	1.05	106.09	3	0.78
	4	84.44	3	1.27	106.09	3	0.93
	8	92.09	5	1.65	117.97	14	1.25
	16	101.92	14	2.29	153.12	21	1.92
Auxiliary Particle Filter $N_p = 400$	0	83.12	1	0.90	97.62	11	0.72
	2	94.72	0	1.16	93.78	4	0.92
	4	86.79	3	1.39	95.00	13	1.11
	8	93.07	6	1.81	116.21	16	1.47
	16	133.64	19	2.55	149.02	27	2.19

7.3.2 Results for the particle filters

In this section, we present results for a generic particle filter and an auxiliary particle filter in the presence of clutter and missed detections. As in the previous section, simulations will be performed using the roll-roll maneuver with $P_D = 0.9$ and $P_D = 0.7$. Because the proposal distribution of the auxiliary particle filter incorporates the current measurement, fewer particles are needed to achieve a desired accuracy. As such, we will take $N_p = 600$ for the generic particle filter and $N_p = 400$ for the auxiliary particle filter. Table 7.9 includes the tracking accuracies for both types of particle filters over a range of clutter densities. The impact of clutter and missed detections can be determined by referring back to Table 7.3 from Section 7.2.2. For those simulations, the generic particle filter with $N_p = 600$ achieved $e_{RMS} = 86.78$ m and $N_L = 0$ while the auxiliary particle filter with $N_p = 400$ achieved $e_{RMS} = 80.25$ m and $N_L = 0$.

First, we compare the clutter performance of both particle filters to the results achieved by the IMM-EKF. As discussed previously, our intent was to choose values of

β_2 and β_3 such that $\bar{N}_g^{EKF} \approx \bar{N}_g^{PF}$. Comparing Tables 7.8 and 7.9, we find that the average gate counts for the particle filters tend to exceed those of the IMM-EKF. However, because the PF and IMM-EKF use different gating conditions, this does not necessarily mean that the particle filters are faced with the more difficult task. As discussed previously, the average gate counts should be interpreted qualitatively, and to this extent, we find that the results are worthy of comparison. Considering e_{RMS} and N_L from Tables 7.8 and 7.9, it is clear that the particle filter is more robust than the IMM-EKF to false alarms and missed detections. For $P_D = 0.9$, it takes \bar{N}_g^{PF} in excess of 2.0 before filter performance begins to degrade substantially. This means that, on average, each particle is updated using both the true measurement and a false alarm. We also find that $P_D = 0.7$, which rendered the IMM-EKF inoperative, results in substantially fewer lost tracks for the particle filter. To understand why the particle filter outperforms the IMM-EKF in the presence of clutter and missed detections, consider how each algorithm maintains its estimate of the posterior density. For the extended Kalman filter, the posterior density is completely specified by (an estimate of) the conditional mean and covariance. Thus, a single false measurement that shifts the conditional mean away from its true value during the Kalman update can change the posterior significantly. The posterior density of the particle filter, on the other hand, is “encoded” in the locations and weights of hundreds of particles. Thus, while any single false measurement might disturb a portion of the particles, it is unlikely that the entire distribution would be substantially distorted. Instead, it would require *many* high-likelihood false alarms to cause a significant change in the entire particle set, something that is statistically less likely.

Now, we compare the performance of the generic particle filter with that of the auxiliary particle filter. Broadly speaking, there is little difference between the performance of the filters for the values of β_3 and P_D that we have considered. However, one observation can be made. It appears that the generic particle filter performs slightly better than the auxiliary particle filter for the simulations with $P_D = 0.9$. On the other hand, the auxiliary particle filter seems to do slightly better when $P_D = 0.7$. Recall that the proposal distribution of the auxiliary particle filter is designed to shift particles into areas of the state space that will result in high data likelihoods. When false measurements are present, this technique can backfire because particles are then shifted into areas of the state space that match the false alarms, but not necessarily the true posterior. On the other hand, this same shortcoming allows the auxiliary particle filter to recover more quickly after a missed detection by shifting particles during state prediction back towards the true measurement, once it reappears. As discussed in the previous section, missed detections are more troublesome than false alarms when tracking maneuvering targets. As such, if P_D is low, the APF’s ability to recover quickly from a missed detection makes it the better choice. However, if missed detections are unlikely because P_D is high, the generic particle filter seems to be more robust to false measurements.

7.4 Classification Results for a Single Target

In this section, we investigate the classification performance of the particle filter. Because the IMM-EKF cannot incorporate RCS as a measurement, it is not suitable as a classifier for FM-band passive radar applications and will not be discussed in this section. Instead, we will compare results for the generic and auxiliary particle filters. For the simulations in this section, we take $\beta_3 = 0$ and $P_D = 1$. This will give an indication of the best that an RCS-based classifier could be expected to achieve. We remove these restrictions when we consider multitarget tracking in the next section.

Before presenting our results, we must first discuss the metric that will be used to gauge classification performance. Given the measurement sequence $\mathbf{z}_{1:k}$, the maximum *a posteriori* (MAP) estimate of target class at scan k is

$$\hat{c}_k \triangleq \arg \max_{j=1,2,\dots,N_c} \Pr(c(\mathbf{x}_k) = j | \mathbf{z}_{1:k}), \quad (7.12)$$

where N_c is the number of classes and c is a decision rule that extracts the discrete class label from the mixed state \mathbf{x}_k . While \hat{c}_k is the class that would be declared by the filter at scan k , further insight can be obtained by considering the entire collection of probabilities $\{\Pr(c(\mathbf{x}_k) = j | \mathbf{z}_{1:k}) : j = 1, \dots, N_c\}$. This collection allows us to see which classes are most often confused. These probabilities can be approximated using the particle weights,

$$\Pr(c(\mathbf{x}_k) = j | \mathbf{z}_{1:k}) \approx \sum_{i=1}^{N_p} \bar{w}_k^{(i)} 1_{j,c(\mathbf{x}_k^{(i)})}, \quad (7.13)$$

where $1_{a,b} = 1$ if $a = b$ and zero otherwise. In order to gauge the performance throughout the maneuver, we will average each of these probabilities across many scans. More specifically, we compute the average class probability for class j as

$$\frac{1}{(N_{MC} - N_L)(N_k - k_0 + 1)} \sum_{n=1}^{N_{MC} - N_L} \sum_{k=k_0}^{N_k} \sum_{i=1}^{N_p} \bar{w}_k^{(i)} 1_{j,c(\mathbf{x}_k^{(i)})}, \quad (7.14)$$

where we have suppressed the particle weights' dependence on the simulation index n . The index k_0 denotes the scan in which averaging begins. By choosing $k_0 = N_k$, we have the average *a posteriori* probability given the entire measurement sequence $\mathbf{z}_{1:N_k}$. However, it is advantageous to choose a smaller value of k_0 in order to provide a broader view of the classifier's performance. This way, the average class probabilities are not overly sensitive to the measurement noise in the last few scans. In our experiments, we take $k_0 = 11$, which results in averaging from $t = 4$ to $t = 30$ s. Because all classes are equally likely at $t = 0$, setting k_0 any smaller yields results that are artificially influenced by the filter's initialization.

A target class is completely specified in our classifier by an RCS table and a set of aerodynamic coefficients. However, as discussed in Section 7.1.3, we do not have enough CAD models to use actual RCS tables in our experiments. Instead, we will use

the scattering center model presented in Section 7.1.3 to generate synthetic RCS data. By varying the parameters of the scattering center model, we can easily create different target classes. The seven classes that we will use are shown in Figure 7.3.

We have a similar problem when it comes to the aerodynamic coefficients required by our state model. Although we used the actual coefficients of an F-4E Phantom to generate the roll-roll and roll-pitch trajectories, we do not have aerodynamic coefficients for any other real aircraft. While we could use the F-4E coefficients for all seven classes, this would negate some of the benefit of performing tracking and classification jointly because variation in the aerodynamic coefficients provides a secondary means of class discrimination. More specifically, if a target’s velocity and orientation can be estimated through Doppler and RCS, the aerodynamic coefficients then specify the ensuing flight path (until the next maneuver). If the target does not follow this path, either the estimates of velocity and orientation are wrong, or the target does not belong to the hypothesized class. For these reasons, the aerodynamic coefficients for classes 2 through 7 will be randomly selected to approximate the characteristics of a fighter-sized plane. Class 1 will continue to use the actual F-4E coefficients. Ground truth trajectories for classes 2 through 7 can then be generated by driving the state model for each class with the original sequence of pilot commands from the roll-roll maneuver.¹⁰

7.4.1 Simulations using three classes

In this section, we compare the performance of the two particle filters using classes 1–3 from Figure 7.3. As mentioned previously, we consider the roll-roll maneuver with $\beta_3 = 0$ and $P_D = 1$. The average class probabilities are computed using (7.14) with $k_0 = 11$ and $N_{MC} = 100$ (as always). The resulting probabilities are listed in Table 7.10 for the generic particle filter and Table 7.11 for the auxiliary particle filter. We find that the classification performance is excellent for both filters. Each table also lists the RMS error as a way of quantifying the impact that class uncertainty has on track accuracy. Compared to the results for a single target of known identity (see Table 7.3), we find that the generic particle filter maintains its accuracy while the auxiliary particle filter degrades slightly. This is not surprising because the particle sets are now split between the three classes (although usually not evenly). In a sense, some of the resources that went to estimating the position and velocity of the target before are now being spent trying to determine the correct class. For the generic particle filter with $N_p = 600$, there are enough particles to spare. For the auxiliary particle filter, which used fewer particles in this experiment ($N_p = 400$), track accuracy is degraded. The solution is simple, of course; we can always increase N_p if needed.

¹⁰Recall, the pilot command set is $\{\gamma_k, \omega_k^*, \Delta\theta_k^*, T_k^*\}$. The mode sequence and accompanying inputs serve as deterministic but unknown inputs to the flight model. Thus, while all seven classes will execute the same roll-roll maneuver, the resulting trajectories will differ somewhat because of the variation in the aerodynamic coefficients.

Table 7.10: Confusion Matrix for Generic PF ($N_p = 600$)

		Average Class Probability			e_{RMS} (m)
		1	2	3	
Correct Class	1	.987	.004	.009	82.27
	2	.001	.999	.000	86.32
	3	.002	.004	.994	88.71

Table 7.11: Confusion Matrix for Auxiliary PF ($N_p = 400$)

		Average Class Probability			e_{RMS} (m)
		1	2	3	
Correct Class	1	.991	.007	.002	89.82
	2	.000	.999	.000	95.81
	3	.002	.002	.996	94.82

7.4.2 Simulations using seven classes

In this section, we consider the more challenging task presented by a 7-class problem. These experiments will use scattering center models 1–7 from Figure 7.3 to generate the synthetic RCS measurements for each class. As before, we take $\beta_3 = 0$ and $P_D = 1$. The average class probabilities will also be computed in the same manner as in the previous section. Because there are over twice as many classes as before, we will increase N_p by 200 particles for each filter. The average class probabilities for the generic and auxiliary particle filters are presented in Tables 7.12 and 7.13, respectively. Overall, we find that both particle filters perform quite well on this task. This offers hope that a real implementation designed around actual RCS tables might perform similarly well. Ironically, scattering center model 1, which we used to generate the synthetic RCS data for all the single-class experiments, happens to be the most difficult to identify. It also yields the largest RMS error of all the classes!

If we compare the diagonal entries of the two confusion matrices, we find that the auxiliary particle filter achieves a slight advantage in classification performance.

Table 7.12: Confusion Matrix for Generic PF ($N_p = 800$)

		Average Class Probability							e_{RMS} (m)
		1	2	3	4	5	6	7	
Correct Class	1	.685	.015	.000	.081	.108	.052	.059	96.37
	2	.000	.998	.001	.000	.000	.000	.001	78.55
	3	.000	.000	.989	.000	.010	.000	.000	84.03
	4	.080	.000	.000	.773	.017	.020	.109	84.83
	5	.051	.002	.000	.025	.882	.001	.039	83.43
	6	.118	.000	.000	.069	.040	.762	.010	89.60
	7	.054	.053	.000	.040	.023	.000	.830	91.82

Table 7.13: Confusion Matrix for Auxiliary PF ($N_p = 600$)

		Average Class Probability							e_{RMS} (m)
		1	2	3	4	5	6	7	
Correct Class	1	.736	.011	.000	.070	.057	.071	.055	97.57
	2	.000	.989	.001	.000	.000	.000	.010	88.28
	3	.011	.003	.955	.000	.019	.000	.011	87.50
	4	.073	.000	.000	.787	.014	.055	.071	83.87
	5	.041	.003	.001	.007	.890	.010	.048	83.91
	6	.139	.000	.000	.043	.003	.813	.002	82.29
	7	.021	.035	.000	.055	.052	.000	.836	88.71

Notably, for the most difficult class to recognize (SC model 1), the average class probability is over 7% greater for the auxiliary particle filter than the generic PF (0.736 versus 0.685). The APF probability for class 6 also achieves a similar gain versus that of the generic PF. Unfortunately, the other diagonal terms are roughly the same or even slightly smaller for the APF. Because we included the auxiliary particle filter in this thesis in hopes that it would provide better classification than the generic PF, this comparison is a bit disappointing. Furthermore, recall that the auxiliary particle filter requires just under twice as much time as the generic PF (for the same value of N_p). Thus, the auxiliary PF used in these experiments will be almost 50% slower than the generic PF. Depending on the computational resources available, the slight improvement in classification offered by the APF may not be attractive enough to offset the increase in run-time.

7.5 Multitarget Tracking and Classification Results

In this final section, we consider the full tracking scenario. Each filter will attempt to track three maneuvering targets in the presence of clutter and missed detections. In order to make the data association more challenging, all three target trajectories will cross, thereby placing the targets in close proximity. Finally, the class identities of the targets will be unknown to the particle filters in order to test their ability to jointly track and classify in the presence of clutter and missed detections. (Note that the classification experiments in the previous section were performed in a clutter-free environment with zero probability of miss. Both restrictions are lifted in this section.)

The three trajectories that we will consider are shown in the left panel of Figure 7.9. The green arrows are spaced 4 s apart and point in the direction of travel. Thus, we see that from time = 12–16 s, all three targets are in close proximity. In the following discussion, we will denote the i^{th} target as t_i . For the purpose of classification, each target must be assigned a class identity. To simplify matters, t_1 will be assigned to class 1, t_2 will be assigned to class 2, and t_3 will be assigned to class 3.

Recall from Section 7.4, a class identity conveys two types of information: (1) kinematic coefficients \mathcal{A}_c for our flight model and (2) a scattering center model for RCS data generation. As discussed in the same section, the kinematic coefficients of class 1

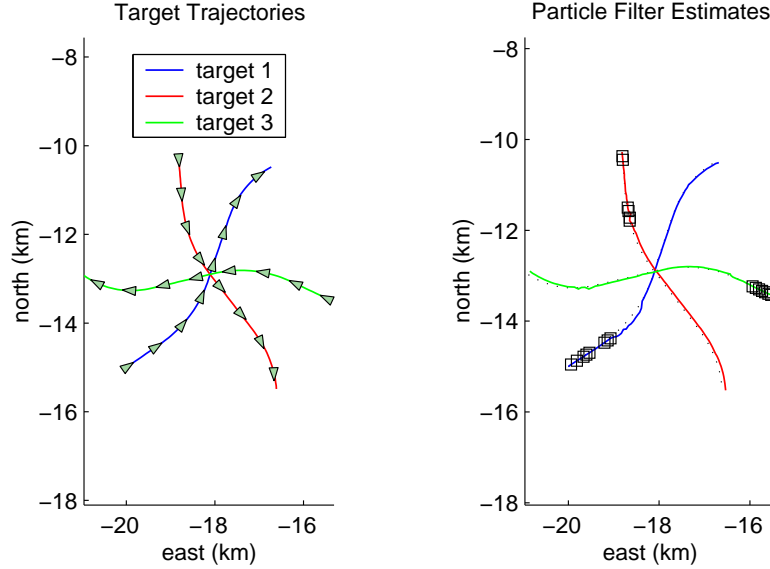


Figure 7.9: Left panel: ground truth trajectories for each target. Arrows are spaced 4 s apart and indicate the direction of travel. Right panel: APF track estimates. Black squares indicate scans in which the MAP class for the given track was incorrect.

Table 7.14: IMM-EKF Multitarget Tracking Accuracy

Target	e_{RMS} (m)	N_L	\bar{N}_q^{EKF}
1	158.76	9	0.93
2	173.03	28	1.03
3	149.10	29	1.02

are those of an F-4E. The kinematic coefficients for the other classes are generated randomly to simulate the characteristics of fighter-sized planes. The blue trajectory flown by t_1 is identical to the roll-roll maneuver, which we have been using throughout this chapter. The red trajectory flown by t_2 is generated by running our state model (see Figures 6.4–6.5) using the pilot commands from the roll-roll maneuver and the class-2 kinematic coefficients. The resulting flight path is then rotated 120° clockwise. The green trajectory flown by t_3 is generated in same way, except the class-3 kinematic coefficients are used and the resulting trajectory is then rotated 120° counterclockwise.

We present the results of the IMM-EKF simulations first. We ran 100 Monte Carlo trials using the optimal choice of $\{q^{(CV)}, q^{(CA)}, q^{(CT)}\}$, as determined in Section 7.2.1. The clutter density and probability of detection were $P_D = 0.9$ and $\beta_2 = 25$. The RMS error e_{RMS} and lost track count N_L are listed in Table 7.14 for each target. Because the trajectory flown by t_1 is identical to the roll-roll maneuver, the filter performance in this case is directly comparable to the $(\beta_2 = 25, P_D = 0.9)$ entry from Table 7.8 (i.e., $e_{RMS} = 167.02$, $N_L = 6$). We find that the performance against the roll-roll maneuver (the blue trajectory in this section) is relatively unaffected by the extension to multiple targets. However, this is not the whole story. Consider the aver-

Table 7.15: PF Multitarget Tracking Accuracy versus σ_n^2

	Target	$\sigma_n^2 = 4$			$\sigma_n^2 = 8$		
		e_{RMS} (m)	N_L	\bar{N}_g^{PF}	e_{RMS} (m)	N_L	\bar{N}_g^{PF}
Generic Particle Filter $N_p = 800$	1	91.50	5	1.02	99.57	4	1.09
	2	102.75	14	1.05	118.93	24	1.13
	3	114.93	16	1.05	128.25	14	1.14
Auxiliary Particle Filter $N_p = 600$	1	99.44	5	1.10	101.09	3	1.21
	2	113.65	22	1.15	110.85	22	1.25
	3	101.18	15	1.16	147.91	15	1.25

age gate counts in Table 7.14. $\bar{N}_g^{EKF} = 0.93$ for t_1 , precisely the same value from the single-target results in Table 7.8. This means that none of the measurements from t_2 or t_3 ever satisfied the gate condition for t_1 (see Equation (7.7)). As such, this remains effectively a single-target experiment for the blue trajectory.

Looking at the average gate counts for t_2 and t_3 , though, we find $\bar{N}_g^{EKF} \approx 1.03$, a substantial difference from the average gate count of t_1 . This means that the gates for t_2 and t_3 overlap near the middle of their trajectories. While their RMS errors are similar to the value of e_{RMS} achieved for t_1 , the number of lost tracks out of 100 simulations is three times larger than the value of N_L for t_1 . Thus, even though the IMM-EKF uses the JPDA algorithm to perform data association jointly, tracking performance is still degraded for tracks whose measurement gates overlap.

Now, we consider the performance of our two particle filters. We discuss track accuracy first. The probability of detection was set to $P_D = 0.9$, to match the IMM-EKF. However, as discussed in Section 7.3, the difference in the dimensions of the EKF and PF measurement spaces prevent us from using the same value for the clutter density. Instead, we chose $\beta_3 = 2$ because it yielded average gate counts comparable to those of the IMM-EKF (with $\beta_2 = 25$). The class library size was set to seven in all cases. Therefore, at initialization, the particle filter for each target was initialized with approximately $N_p/7$ particles per class. The right panel of Figure 7.9 shows an example of the filtered estimates produced by the auxiliary particle filter for each target.

In Table 7.15, tracking results are presented for our generic PF and our auxiliary PF for two different values of σ_n^2 . Recall that σ_n^2 is the variance of the independent identically distributed Gaussian noise processes that affect the in-phase and quadrature channels of our receiver (see Section 6.2). In Section 7.2.2, we considered several different values for this parameter and settled upon $\sigma_n^2 = 8$. However, we also acknowledged that the literature has little to offer on this choice. In this section, we will return to this issue in order to determine the level of performance that would be achievable if $\sigma_n^2 = 4$. It is our expectation that a smaller value of σ_n^2 will lead to improved performance.

Because the blue trajectory is identical to the roll-roll maneuver, the $\sigma_n^2 = 8$ results for t_1 from Table 7.15 can be compared to the ($\beta_3 = 2, P_D = 0.9$) entries from Table 7.9. Recall that the experiments summarized in Table 7.9 assumed that the iden-

tivity of the (single) target was known. In this section, that is no longer the case, and we find that the additional uncertainty due to unknown target class increases the RMS error for both particle filters. It should be noted, though, that part of the degradation in performance can be attributed to a reduction in effective particle count per class. For example, in Section 7.2.2, the generic particle filter devoted $N_p=600$ particles to a single class. In this section, the generic PF shares 800 particles between seven classes. Of course, in the process of resampling, the distribution of particles across classes can change substantially, and the most likely classes will eventually end up with significantly more than $N_p/7$ particles. Nonetheless, in the time it takes for a class to attain a high likelihood and a greater share of the sample set, estimation accuracy is still likely to suffer.

There are several comments that should be made concerning the results in Table 7.15. First, on comparing Tables 7.14 and 7.15, we find that both particle filters outperform the IMM-EKF for all three targets, sometimes by a substantial margin. At first, this might seem surprising on this multitarget task. After all, the IMM-EKF uses joint data association whereas our particle filters use only (single-target) probabilistic data association (see Section 6.3.2). First, to make sure that the particle filter gates are overlapping, such that joint data association would be appropriate, consider the average gate counts \bar{N}_g^{PF} . As was the case with the IMM-EKF, we find that the gate counts are higher for t_2 and t_3 than t_1 . This implies that measurements from t_2 are influencing the track for t_3 and vice versa. So how is it that probabilistic data association can still yield good performance? There are two reasons for this.

First, by including RCS in the measurement vector, it is more difficult for the return from a target to satisfy the PF gate criterion for the wrong track. In Table 7.15, the proximity of t_2 and t_3 during the experiment results in values of \bar{N}_g^{PF} that are 3–5% higher than those of t_1 . However, in Table 7.14, the same trajectories result in values of \bar{N}_g^{EKF} for t_2 and t_3 that are 10% higher than those of t_1 . Second, and more importantly, we must recall that the JPDA is a *suboptimal* assignment algorithm, even for multiple targets. As discussed in Section 5.3.3, optimal data association requires that a separate filter be run for every possible data association hypothesis $\Gamma_{1:k}^{(l)}$, which generally leads to a distinctly non-Gaussian (multimodal) posterior distribution. The JPDA, on the other hand, “re-Gaussianizes” its estimate of the posterior at each scan. When the measurement gates for two or more targets overlap, a Gaussian function can become a poor approximation to the true posterior. Thus, we have the following trade-off.

Because our particle filters can accurately represent multimodal distributions, they are better-equipped to handle the measurement ambiguity from overlapping gates. However, because our PF data association is not performed jointly across all targets, the filters expect a single persistent return per gate. When this assumption is violated, performance can degrade. Our IMM-EKF, on the other hand, is able to perform data association jointly. Thus, when the measurement gates of t_2 and t_3 overlap, the filter expects two persistent returns per gate for each track. However, the IMM-EKF cannot model this ambiguity correctly. Referring to Equation (5.44), we see that the posterior covariance will be artificially inflated to reflect the measurement uncertainty. This de-

creased confidence in the accuracy of its state estimate can lead the EKF to becoming “distracted” by actual false returns.

Comparing the performance of our two particle filters to each other in Table 7.15, we find that the generic particle filter performs slightly better overall. There are two potential reasons for this. First, note that $N_p = 800$ for the generic PF and $N_p = 600$ for the auxiliary PF. These particle counts were chosen so that the computational effort of each algorithm was more closely matched. In this 7-class experiment, 600 particles may simply be too few. Second, and more importantly, the proposal distribution of the APF may be poorly matched to tracking in the presence of clutter and missed detections. In Section 4.3.4, the APF was shown to resample from the previous states $\{\mathbf{x}_{k-1}^{(i)}\}$ in order to focus attention on areas of the state space that seem most promising given \mathbf{z}_k , the current measurement vector. While this might be a good idea in a clutter-free environment, in the presence of data uncertainty, the APF can end up wasting many of its samples “chasing” false returns. If we compare \bar{N}_g^{PF} for both types of particle filters, we find that the average gate count for the APF is indeed higher than that of the generic PF.

Finally, comparing the tracking accuracy versus the choice of σ_n^2 in Table 7.15, we find that, as expected, both particle filters perform better at the lower noise variance ($\sigma_n^2 = 4$). There are two reasons for this. First, the values of \bar{N}_g^{PF} are smaller for $\sigma_n^2 = 4$ than $\sigma_n^2 = 8$. As the scattering coefficient noise variance decreases, it becomes more difficult for false returns to satisfy the gate condition. The resulting reduction in data uncertainty leads to an improvement in tracking performance. Second, classification accuracy also improves as σ_n^2 decreases. We will discuss the matter of classification next, but now we emphasize that tracking and classification proceed jointly in our particle filters. More specifically, those particles whose class labels ζ are incorrect use the wrong kinematic coefficients during prediction. This, of course, leads to a poor match with the actual dynamics of the target. Therefore, anything that degrades classification performance will also degrade tracking accuracy (and vice versa).

We now consider the classification performance of both particle filters. Recall that, for this experiment, t_1 is a class-1 aircraft, t_2 is a class-2 aircraft, and t_3 is a class-3 aircraft. The right panel of Figure 7.9 sheds some light on the classification process. In this plot, a black square is used to identify those scans where the MAP estimates derived using (7.13) were incorrect. Because each class is equally likely at initialization, we expect to find black squares at the beginning of each track. This is exactly what we see in Figure 7.9. Therefore, to prevent initialization from diluting our classification results, we ignore the first 10 scans in computing the average class probabilities for each target (i.e., we take $k_0 = 11$ in Equation (7.14)). This is the same approach that was used in Section 7.4.

The classification results for our generic particle filter and auxiliary particle filter are presented in Tables 7.16 and 7.17, respectively. To quantify the effect that data uncertainty has on classification performance, we refer back to Tables 7.12 and 7.13. Recall, the simulations in Section 7.4 were conducted for a single target in a clutter-free environment with zero probability of missed detection. However, the scattering

Table 7.16: Multitarget Confusion Matrix for Generic PF ($N_p = 800$)

	Correct Class	Average Class Probability						
		1	2	3	4	5	6	7
$\sigma_n^2 = 4$	$t_1: 1$.933	.001	.000	.019	.006	.031	.011
	$t_2: 2$.000	.870	.000	.012	.000	.003	.114
	$t_3: 3$.002	.000	.934	.001	.014	.050	.000
$\sigma_n^2 = 8$	$t_1: 1$.683	.014	.001	.089	.114	.050	.050
	$t_2: 2$.014	.855	.000	.022	.001	.024	.085
	$t_3: 3$.001	.007	.852	.010	.030	.092	.009

Table 7.17: Multitarget Confusion Matrix for Auxiliary PF ($N_p = 600$)

	Correct Class	Average Class Probability						
		1	2	3	4	5	6	7
$\sigma_n^2 = 4$	$t_1: 1$.830	.001	.000	.039	.032	.057	.040
	$t_2: 2$.002	.898	.001	.001	.000	.004	.094
	$t_3: 3$.013	.001	.858	.023	.047	.058	.000
$\sigma_n^2 = 8$	$t_1: 1$.726	.019	.000	.055	.075	.063	.061
	$t_2: 2$.017	.842	.002	.024	.001	.018	.096
	$t_3: 3$.012	.002	.808	.004	.094	.079	.001

coefficient noise variance was taken to be $\sigma_n^2 = 8$. Thus, the bottom three rows of Tables 7.16 and 7.17 may be directly compared to the top three rows of Tables 7.12 and 7.13. For both particle filters, we find that the average probability of class 1 is unaffected by the measurement uncertainty. The same is not true, though, for classes 2 and 3 whose probabilities decrease significantly. While we have already determined that the gates of t_2 and t_3 overlap for part of their trajectories, it is unlikely that this is the primary cause of the degraded performance. If it were the case, we would expect to see t_2 confused much more often for class 3 (and t_3 for class 2) in Tables 7.16 and 7.17. Instead, it is more likely that the presence of false returns and missed detections are simply taking their toll on the classifier.

In terms of class accuracy, the auxiliary PF does not seem to offer an advantage over the generic PF, which uses the prior as its proposal distribution. This is not surprising, because we have already theorized that the proposal distribution of the APF may actually do more harm than good in the presence of false returns. What is more significant in Tables 7.16 and 7.17 is the improvement in class accuracy achieved by reducing σ_n^2 from 8 to 4. For example, the average class probability of class 1 goes from 0.683 to 0.933 for the generic PF. Because RCS is the primary feature for class discrimination, it makes sense that more accurate RCS measurements should translate into improved recognition performance.¹¹

In summary, we have demonstrated that our particle filters are able to track and classify multiple targets in the presence of both clutter and missed detections. In addition,

¹¹Target dynamics via the kinematic coefficients, $\mathcal{A}_c = \{C_{L_\alpha}, C_{D_{\min}}, K_D, S, m\}$, would be a secondary feature for class discrimination.

both filters were shown to outperform the IMM-EKF. There are three main reasons for the improvement in tracking accuracy achieved by the particle filter. First, its ability to model non-Gaussian densities makes it robust in cluttered environments. Second, its ability to model nonlinear systems of almost arbitrary complexity allows us to use an aerodynamically valid flight model. Third, its ability to incorporate RCS as a data feature permits classification to occur simultaneously, which provides the class-specific kinematic coefficients \mathcal{A}_c needed for our flight model.

CHAPTER 8

CONCLUSIONS

In this thesis, we have presented a recursive Bayesian formulation for joint tracking and classification. The specific application we considered was FM radio-based passive air surveillance. For our simulations to be as realistic as possible, we modeled false alarms, missed detections, and targets with overlapping measurement gates. Most importantly, we established a legitimate link between classification and tracking by implementing an aerodynamically valid flight model, which requires vehicle-specific coefficients such as the target's wing area, minimum drag, and mass. Of course, our classifier relies on the tracker for position and orientation information, so that RCS values can be predicted. Thus, there is an important two-way exchange of information between tracker and classifier.

The key feature that bridges the gap between tracking and classification is radar cross section, which we incorporated in our measurement vector. By modeling the true deterministic relationship that exists between RCS and target aspect, we were able to gain both valuable class information and a method for estimating target orientation. However, the lack of a closed-form relationship between RCS and target aspect required that we implement our system using a particle filter.

In Chapter 7, we demonstrated that our particle filter-based system was indeed able to track and classify multiple targets in the presence of both clutter and missed detections. In addition, its tracking accuracy was shown to be superior to the IMM-EKF. There are three main reasons for the improvement in particle filter performance. First, its ability to model non-Gaussian densities makes it robust in cluttered environments. Second, its ability to model nonlinear systems of almost arbitrary complexity allows us to use our aerodynamically valid flight model. Third, its ability to incorporate RCS as a data feature makes our entire joint approach feasible.

The one area where our implementation did not excel was that of computational complexity. In our simulations, our particle filter was found to be two orders of magnitude slower than our IMM-EKF. As such, a future area of research could concentrate on ways to achieve the same performance with fewer particles. For example, N_p could be allowed to vary based upon the effective sample size (or an estimate of it). A second possible area for future research involves the issue of robustness during classification. If an unusually noisy RCS measurement is received, particles corresponding to the true

class may score poorly during the measurement update. If, during resampling, many of them are eliminated, the filter may then diverge. It would be useful to guard against this scenario, possibly by repopulating classes that have been eliminated. However, care would have to be taken that the sample weights were adjusted appropriately so that a valid estimate of the posterior was maintained.

APPENDIX A

DEGENERACY OF IMPORTANCE WEIGHTS

In this appendix, we provide a proof for Lemma 3. We show that the unconditional variance of the importance weights from the SIS algorithm cannot decrease with time.¹ The significance of this result is that, practically speaking, there is no way to avoid the problem of degeneracy in sequential importance sampling. Thus, without resampling, the importance weights will degrade until eventually $\bar{w}_k^{(i)} \approx 0$ for all but a single value of i .

Comparing Equations (4.10) and (4.14), we have $w_k^{*(i)} = w^{(i)}/p(\mathbf{z}_{1:k})$. When implementing the SIS algorithm, the true importance weights $\{w_k^{*(i)}\}$ are hardly ever used because they require evaluation of $p(\mathbf{z}_{1:k})$. However, because $\mathbf{z}_{1:k}$ is considered random in this proof, we must proceed with the true weights. Also, for simplicity, we suppress the superscript i for both $w_k^{*(i)}$ and the imputed state sequence $\mathbf{x}_{0:k}^{(i)}$ in what follows. The proof is combined from [53, 78].

We begin by showing that w_k^* is a martingale in k . Define the auxiliary distribution

$$\tilde{\pi}(\mathbf{x}_k, \mathbf{z}_k | \mathbf{x}_{0:k-1}, \mathbf{z}_{1:k-1}) \triangleq \pi(\mathbf{x}_k | \mathbf{x}_{0:k-1}, \mathbf{z}_{1:k}) \pi(\mathbf{z}_k | \mathbf{x}_{0:k-1}, \mathbf{z}_{1:k-1}) \quad (\text{A.1})$$

$$= \pi(\mathbf{x}_k | \mathbf{x}_{k-1}, \mathbf{z}_k) p(\mathbf{z}_k | \mathbf{z}_{1:k-1}). \quad (\text{A.2})$$

In words, $\tilde{\pi}(\mathbf{x}_k, \mathbf{z}_k | \mathbf{x}_{0:k-1}, \mathbf{z}_{1:k-1})$ is the distribution from which the current state and measurement are drawn, conditioned on the previous state and measurement histories. In going from (A.1) to (A.2), we use the fact that the measurement \mathbf{z}_k is *not* drawn from our proposal distribution. Rather, it is drawn from the true distribution governing the observation process. Furthermore, because the state sequence $\mathbf{x}_{1:k-1}^{(i)}$ (where we have suppressed the superscript i in (A.1)) is imputed from $\mathbf{z}_{1:k-1}$, and \mathbf{x}_0 is independent of the measurement process, we have $p(\mathbf{z}_k | \mathbf{x}_{0:k-1}, \mathbf{z}_{1:k-1}) = p(\mathbf{z}_k | \mathbf{z}_{1:k-1})$. Applying

¹This statement applies in a stochastic sense, where the unconditional variance is taken with the measurement sequence $\mathbf{z}_{1:k}$ treated as random.

the definition in (4.10), we have

$$\begin{aligned} \mathbb{E}_{\tilde{\pi}}[w_k^* | \mathbf{x}_{0:k-1}, \mathbf{z}_{1:k-1}] &= \iint \frac{p(\mathbf{z}_{1:k} | \mathbf{x}_{0:k}) p(\mathbf{x}_{0:k})}{p(\mathbf{z}_{1:k}) \pi(\mathbf{x}_{0:k} | \mathbf{z}_{1:k})} \tilde{\pi}(\mathbf{x}_k, \mathbf{z}_k | \mathbf{x}_{0:k-1}, \mathbf{z}_{1:k-1}) d\mathbf{x}_k d\mathbf{z}_k \\ &= w_{k-1}^* \iint \frac{p(\mathbf{z}_k | \mathbf{x}_k) p(\mathbf{x}_k | \mathbf{x}_{k-1})}{p(\mathbf{z}_k | \mathbf{z}_{1:k-1}) \pi(\mathbf{x}_k | \mathbf{x}_{k-1}, \mathbf{z}_k)} \\ &\quad \times \tilde{\pi}(\mathbf{x}_k, \mathbf{z}_k | \mathbf{x}_{0:k-1}, \mathbf{z}_{1:k-1}) d\mathbf{x}_k d\mathbf{z}_k \end{aligned} \quad (\text{A.3})$$

$$= w_{k-1}^* \iint p(\mathbf{z}_k | \mathbf{x}_k) p(\mathbf{x}_k | \mathbf{x}_{k-1}) d\mathbf{x}_k d\mathbf{z}_k \quad (\text{A.4})$$

$$= w_{k-1}^* \iint p(\mathbf{x}_k, \mathbf{z}_k | \mathbf{x}_{k-1}) d\mathbf{x}_k d\mathbf{z}_k \quad (\text{A.5})$$

$$= w_{k-1}^*. \quad (\text{A.6})$$

To arrive at (A.3), we used the fact that $\mathbf{x}_{0:k}$ is Markov and $\mathbf{z}_{1:k}$ is conditionally independent given $\mathbf{x}_{0:k}$. This allowed us to extract the previous weight w_{k-1}^* from the integral. We also applied (4.25) in order to factor the proposal distribution $\pi(\mathbf{x}_{0:k} | \mathbf{z}_{1:k})$. Substitution of the definition of the auxiliary distribution (A.2) then yields (A.4). Therefore, we have shown that w_k^* is a martingale in k .

Using the fact that w_k^* is a martingale and the variance decomposition

$$\text{var}[X] = \text{var}[\mathbb{E}[X|Y]] + \mathbb{E}[\text{var}[X|Y]], \quad (\text{A.7})$$

we can now prove the lemma.

$$\text{var}_{\pi(\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k-1})}[w_{k-1}^*] = \text{var}_{\pi(\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k-1})}[\mathbb{E}_{\tilde{\pi}}[w_k^* | \mathbf{x}_{0:k-1}, \mathbf{z}_{1:k-1}]] \quad (\text{A.8})$$

$$= \text{var}_{\pi(\mathbf{x}_{0:k}, \mathbf{z}_{1:k})}[w_k^*] - \mathbb{E}[\text{var}_{\tilde{\pi}}[w_k^* | \mathbf{x}_{0:k-1}, \mathbf{z}_{1:k-1}]] \quad (\text{A.9})$$

$$\leq \text{var}_{\pi(\mathbf{x}_{0:k}, \mathbf{z}_{1:k})}[w_k^*], \quad (\text{A.10})$$

where we used (A.6) to obtain (A.8), and the variance decomposition (A.7) to obtain (A.9). Thus, we have obtained the result that the unconditional variance of w_k^* cannot decrease with k . Of course, for a given experiment, the measurement sequence $\mathbf{z}_{1:k}$ is fixed, so we might wonder how the lemma should be interpreted. Using the variance decomposition once again, we have

$$\text{var}[w_k^*] = \text{var}[\mathbb{E}[w_k^* | \mathbf{z}_{1:k}]] + \mathbb{E}[\text{var}[w_k^* | \mathbf{z}_{1:k}]]. \quad (\text{A.11})$$

Then, because

$$\mathbb{E}[w_k^* | \mathbf{z}_{1:k}] = \int \frac{p(\mathbf{x}_{0:k} | \mathbf{z}_{1:k})}{\pi(\mathbf{x}_{0:k} | \mathbf{z}_{1:k})} \pi(\mathbf{x}_{0:k} | \mathbf{z}_{1:k}) d\mathbf{x}_{0:k} = 1, \quad (\text{A.12})$$

substitution of (A.12) into (A.11) yields $\mathbb{E}[\text{var}[w_k^* | \mathbf{z}_{1:k}]] = \text{var}[w_k^*]$. Therefore, for a given measurement sequence, the conditional variance of the true importance weights will tend to increase over time (although not strictly so), resulting in the degeneracy of the posterior estimates.

APPENDIX B

DERIVATION OF EFFECTIVE SAMPLE SIZE

In this appendix, we derive the formula for the effective sample size. Although many sequential importance sampling algorithms use an estimate of N_{eff} to determine when to resample, a complete derivation of N_{eff} is often neglected. The following derivation is based upon the discussion in [48]. For notational convenience, we will suppress the dependence on the time index k .

Consider the state vector $\mathbf{x} \sim p(\mathbf{x}|\mathbf{z})$. We wish to estimate

$$\mu_g = \mathbb{E}_p[g(\mathbf{x})|\mathbf{z}]$$

by Monte Carlo integration, where it is assumed that $\text{var}_p[g(\mathbf{x})|\mathbf{z}] < \infty$. If we are unable to draw samples directly from $p(\mathbf{x}|\mathbf{z})$, we can estimate μ_g using importance sampling. Specifically, let $\pi(\mathbf{x}|\mathbf{z})$ denote the importance density from which we draw N_p independent samples $\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(N_p)}\}$. Applying (4.15) from Section 4.1.3,

$$\hat{\mu}_g \triangleq \frac{\sum_{i=1}^{N_p} w^{(i)} g(\mathbf{x}^{(i)})}{\sum_{j=1}^{N_p} w^{(j)}} = \frac{\sum_{i=1}^{N_p} w^{*(i)} g(\mathbf{x}^{(i)})}{\sum_{j=1}^{N_p} w^{*(j)}} \quad (\text{B.1})$$

provides an asymptotically unbiased estimate of μ_g (under mild restrictions).¹ The second equality in (B.1) follows from Equations (4.10) and (4.14), which yield $w^{(i)} = p(\mathbf{z})w^{*(i)}$. Note that the equivalence between the two estimates in (B.1) only makes sense for a fixed realization of the measurement process (i.e., if \mathbf{z} is not random). In this case, the true importance weights $\{w^{*(i)}\}$ and the unnormalized weights $\{w^{(i)}\}$ differ only by a constant. While the first form of $\hat{\mu}_g$ would be used in practice, the derivation of N_{eff} is simpler if we use the second form.

We wish to choose N_p large enough so that the random variation created by our Monte Carlo approximation has a negligible effect on the fidelity of our estimate of μ_g . To accomplish this, we require that $\text{var}_\pi[\hat{\mu}_g|\mathbf{z}]$ be small relative to the true vari-

¹In this section, we adopt a more compact notation than Section 4.1.3. Specifically, we replace $I(g)$ with μ_g and $\hat{I}_{N_p}(g)$ with $\hat{\mu}_g$, thereby suppressing the dependence on N_p because the sample size will not vary in the current discussion.

ance $\text{var}_p[g(\mathbf{x})|\mathbf{z}]$, as suggested in [53]. We begin by finding an approximation for the variance of $\hat{\mu}_g$.

Let $\mathbf{y}^{(i)} \triangleq w^{*(i)}g(\mathbf{x}^{(i)})$. An approximation for the variance of $\hat{\mu}_g$ can be found by replacing the quotient in (B.1) with a first-order Taylor series approximation in the random variables $w^{*(1)}, \dots, w^{*(N_p)}, \mathbf{y}^{(1)}, \dots, \mathbf{y}^{(N_p)}$. In the statistics literature, this approximation technique is referred to as the *delta method*. The variance of the resulting linearized function will have terms up to second order only. Before proceeding, for $i = 1, \dots, N_p$, we note that

$$\mathbb{E}_\pi[w^{*(i)}|\mathbf{z}] = \int \frac{p(\mathbf{x}|\mathbf{z})}{\pi(\mathbf{x}|\mathbf{z})} \pi(\mathbf{x}|\mathbf{z}) d\mathbf{x} = \int p(\mathbf{x}|\mathbf{z}) d\mathbf{x} = 1, \quad (\text{B.2})$$

$$\mathbb{E}_\pi[\mathbf{y}^{(i)}|\mathbf{z}] = \int \frac{p(\mathbf{x}|\mathbf{z})}{\pi(\mathbf{x}|\mathbf{z})} g(\mathbf{x}) \pi(\mathbf{x}|\mathbf{z}) d\mathbf{x} = \int g(\mathbf{x}) p(\mathbf{x}|\mathbf{z}) d\mathbf{x} = \mu_g, \quad (\text{B.3})$$

where we assume that the support of $p(\mathbf{x}|\mathbf{z})$ is a subset of the support of $\pi(\mathbf{x}|\mathbf{z})$. We now linearize $\hat{\mu}_g$ about the expected values of $\{w^{*(i)}, \mathbf{y}^{(i)}\}_{i=1}^{N_p}$.

$$\begin{aligned} \hat{\mu}_g &= \frac{\sum_{i=1}^{N_p} \mathbf{y}^{(i)}}{\sum_{j=1}^{N_p} w^{*(j)}} \\ &\approx \frac{\sum_{i=1}^{N_p} \mathbf{y}^{(i)}}{\sum_{j=1}^{N_p} w^{*(j)}} \Bigg|_{\substack{\mathbf{y}^{(i)}=\mu_g, \\ w^{*(j)}=1}} + \frac{1}{\sum_{j=1}^{N_p} w^{*(j)}} \Bigg|_{w^{*(j)}=1} \cdot \sum_{i=1}^{N_p} (\mathbf{y}^{(i)} - \mu_g) \\ &\quad - \frac{\sum_{i=1}^{N_p} \mathbf{y}^{(i)}}{\left(\sum_{j=1}^{N_p} w^{*(j)}\right)^2} \Bigg|_{\substack{\mathbf{y}^{(i)}=\mu_g, \\ w^{*(j)}=1}} \cdot \sum_{i=1}^{N_p} (w^{*(i)} - 1) \\ &= \mu_g + \frac{1}{N_p} \sum_{i=1}^{N_p} (\mathbf{y}^{(i)} - \mu_g) - \frac{\mu_g}{N_p} \sum_{i=1}^{N_p} (w^{*(i)} - 1). \end{aligned} \quad (\text{B.4})$$

We will denote the linearized estimate in (B.4) as $\hat{\mu}_{g,lin}$. Because $\mathbb{E}_\pi[\mathbf{y}^{(i)}|\mathbf{z}] = \mu_g$ and $\mathbb{E}_\pi[w^{*(i)}|\mathbf{z}] = 1$ for all i , we immediately have $\mathbb{E}_\pi[\hat{\mu}_{g,lin}|\mathbf{z}] = \mu_g$. To find $\mathbb{E}_\pi[\hat{\mu}_{g,lin}^2|\mathbf{z}]$, we note that, because the sequence $\{\mathbf{x}^{(i)}\}$ is independent and identically distributed (iid), the sequences $\{w^{*(i)}\}$ and $\{\mathbf{y}^{(i)}\}$ are also iid. Furthermore, $w^{*(i)}$ and $\mathbf{y}^{(j)}$ are independent for $i \neq j$. The latter is not true for $i = j$, though, because $w^{*(i)}$ and $\mathbf{y}^{(i)}$ are then both functions of the same random variable $\mathbf{x}^{(i)}$. We can now show

$$\begin{aligned}
\mathbb{E}_\pi[\hat{\mu}_{g,lin}^2|\mathbf{z}] &= \mathbb{E}_\pi \left[\left(\mu_g + \frac{1}{N_p} \sum_{i=1}^{N_p} (\mathbf{y}^{(i)} - \mu_g) - \frac{\mu_g}{N_p} \sum_{i=1}^{N_p} (w^{*(i)} - 1) \right)^2 \middle| \mathbf{z} \right] \\
&= \mu_g^2 + \frac{1}{N_p^2} \sum_{i=1}^{N_p} \mathbb{E}_\pi[(\mathbf{y}^{(i)} - \mu_g)^2|\mathbf{z}] \\
&\quad - \frac{2\mu_g}{N_p^2} \sum_{i=1}^{N_p} \mathbb{E}_\pi[(\mathbf{y}^{(i)} - \mu_g)(w^{*(i)} - 1)|\mathbf{z}] \\
&\quad + \frac{\mu_g^2}{N_p^2} \sum_{i=1}^{N_p} \mathbb{E}_\pi[(w^{*(i)} - 1)^2|\mathbf{z}], \tag{B.5}
\end{aligned}$$

where we have used the independence of the various random variables to eliminate all double sums. Because $\{w^{*(i)}\}$ and $\{\mathbf{y}^{(i)}\}$ are identically distributed sequences, (B.5) can be expressed more simply as

$$\mathbb{E}_\pi[\hat{\mu}_{g,lin}^2|\mathbf{z}] = \mu_g^2 + \frac{1}{N_p} \text{var}_\pi[\mathbf{y}|\mathbf{z}] - \frac{2\mu_g}{N_p} \text{cov}_\pi[\mathbf{y}, w^*|\mathbf{z}] + \frac{\mu_g^2}{N_p} \text{var}_\pi[w^*|\mathbf{z}]. \tag{B.6}$$

Combining (B.6) with $\mathbb{E}_\pi[\hat{\mu}_{g,lin}|\mathbf{z}] = \mu_g$ yields the desired approximation,

$$\begin{aligned}
\text{var}_\pi[\hat{\mu}_g|\mathbf{z}] &= \mathbb{E}_\pi[\hat{\mu}_{g,lin}^2|\mathbf{z}] - \mathbb{E}_\pi^2[\hat{\mu}_{g,lin}|\mathbf{z}], \\
&\approx \frac{1}{N_p} \left(\text{var}_\pi[\mathbf{y}|\mathbf{z}] - 2\mu_g \text{cov}_\pi[\mathbf{y}, w^*|\mathbf{z}] + \mu_g^2 \text{var}_\pi[w^*|\mathbf{z}] \right). \tag{B.7}
\end{aligned}$$

Our linearized approximation indicates that if all the terms in (B.7) are finite, the Monte Carlo variation in $\hat{\mu}_g$ should decrease as N_p^{-1} . Unfortunately, because $\mathbf{y} = w^*(\mathbf{x})g(\mathbf{x})$, our approximation for $\text{var}_\pi[\hat{\mu}_g|\mathbf{z}]$ is dependent on the choice of the function g . Ideally, we would like to derive an effective sample size that could be applied over a broad choice of g . We also wish to take all expectations involving $g(\mathbf{x})$ with respect to the true density $p(\mathbf{x}|\mathbf{z})$ and not the importance density $\pi(\mathbf{x}|\mathbf{z})$. To accomplish both of these, we need to make a further approximation.

We begin by expanding $\text{var}_\pi[\mathbf{y}|\mathbf{z}]$ from (B.7). To simplify notation, we also replace $g(\mathbf{x})$ with g .

$$\begin{aligned}
\text{var}_\pi[\mathbf{y}|\mathbf{z}] &= \mathbb{E}_\pi \left[\left(\frac{p(\mathbf{x}|\mathbf{z})}{\pi(\mathbf{x}|\mathbf{z})} g(\mathbf{x}) \right)^2 \middle| \mathbf{z} \right] - \mathbb{E}_\pi^2 \left[\frac{p(\mathbf{x}|\mathbf{z})}{\pi(\mathbf{x}|\mathbf{z})} g(\mathbf{x}) \middle| \mathbf{z} \right] \\
&= \mathbb{E}_p \left[\frac{p(\mathbf{x}|\mathbf{z})}{\pi(\mathbf{x}|\mathbf{z})} g^2(\mathbf{x}) \middle| \mathbf{z} \right] - \mathbb{E}_p^2 [g(\mathbf{x})|\mathbf{z}] \\
&= \mathbb{E}_p[g^2 w^*|\mathbf{z}] - \mu_g^2. \tag{B.8}
\end{aligned}$$

We now replace $g^2 w^*$ by a second-order Taylor series approximation about the expected values of w^* and g with respect to $p(\mathbf{x}|\mathbf{z})$. To simplify notation, define $\mu_w =$

$E_p[w^*|\mathbf{z}]$. Note that, while $E_\pi[w^*|\mathbf{z}] = 1$, the same is not generally true for μ_w .

$$\begin{aligned}
g^2 w^* &\approx g^2 w^* \Big|_{\substack{g=\mu_g \\ w^*=\mu_w}} + 2g w^* \Big|_{\substack{g=\mu_g \\ w^*=\mu_w}} \cdot (g - \mu_g) + g^2 \Big|_{g=\mu_g} \cdot (w^* - \mu_w) \\
&\quad + \frac{1}{2} \left(2w^* \Big|_{w^*=\mu_w} \cdot (g - \mu_g)^2 + 2 \cdot 2g \Big|_{g=\mu_g} \cdot (g - \mu_g)(w^* - \mu_w) \right) \\
&= \mu_g^2 \mu_w + 2\mu_g \mu_w (g - \mu_g) + \mu_g^2 (w^* - \mu_w) + \mu_w (g - \mu_g)^2 \\
&\quad + 2\mu_g (g - \mu_g)(w^* - \mu_w). \tag{B.9}
\end{aligned}$$

Taking the conditional expectation of the second-order approximation in (B.9) with respect to the true density and substituting the result back into (B.8) yields

$$\text{var}_\pi[\mathbf{y}|\mathbf{z}] \approx \mu_g^2 \mu_w + \mu_w \text{var}_p[g|\mathbf{z}] + 2\mu_g \text{cov}_p[g, w^*|\mathbf{z}] - \mu_g^2. \tag{B.10}$$

Thus, we have succeeded in approximating $\text{var}_\pi[\mathbf{y}|\mathbf{z}]$ using only expectations taken with respect to $p(\mathbf{x}|\mathbf{z})$, the true density. We now accomplish the same for the second term from (B.7).

$$\begin{aligned}
\text{cov}_\pi[\mathbf{y}, w^*|\mathbf{z}] &= E_\pi[\mathbf{y}w^*|\mathbf{z}] - E_\pi[\mathbf{y}|\mathbf{z}] E_\pi[w^*|\mathbf{z}] \\
&= E_\pi \left[g \left(\frac{p(\mathbf{x}|\mathbf{z})}{\pi(\mathbf{x}|\mathbf{z})} \right)^2 \Big| \mathbf{z} \right] - E_\pi \left[g \frac{p(\mathbf{x}|\mathbf{z})}{\pi(\mathbf{x}|\mathbf{z})} \Big| \mathbf{z} \right] E_\pi \left[\frac{p(\mathbf{x}|\mathbf{z})}{\pi(\mathbf{x}|\mathbf{z})} \Big| \mathbf{z} \right] \\
&= E_p[gw^*|\mathbf{z}] - \mu_g \cdot 1 \\
&= \text{cov}_p[g, w^*|\mathbf{z}] + \mu_g \mu_w - \mu_g. \tag{B.11}
\end{aligned}$$

To arrive at the final result, we substitute (B.10) and (B.11) back into Equation (B.7) and combine terms.

$$\begin{aligned}
\text{var}_\pi[\hat{\mu}_g|\mathbf{z}] &\approx \frac{1}{N_p} \left(\text{var}_\pi[\mathbf{y}|\mathbf{z}] - 2\mu_g \text{cov}_\pi[\mathbf{y}, w^*|\mathbf{z}] + \mu_g^2 \text{var}_\pi[w^*|\mathbf{z}] \right) \\
&\approx \frac{1}{N_p} \left(\mu_w \text{var}_p[g|\mathbf{z}] - \mu_g^2 \mu_w + \mu_g^2 + \mu_g^2 \text{var}_\pi[w^*|\mathbf{z}] \right) \\
&= \frac{1}{N_p} \left(E_\pi[w^{*2}|\mathbf{z}] \text{var}_p[g|\mathbf{z}] + \mu_g^2 \text{var}_\pi[w^*|\mathbf{z}] - \mu_g^2 (E_\pi[w^{*2}|\mathbf{z}] - 1) \right). \tag{B.12}
\end{aligned}$$

In (B.12), we used the result that $E_\pi[w^{*2}|\mathbf{z}] = E_p[w^*|\mathbf{z}] = \mu_w$. Because $E_\pi[w^*|\mathbf{z}] = 1$, we recognize that $(E_\pi[w^{*2}|\mathbf{z}] - 1) = \text{var}_\pi[w^*|\mathbf{z}]$. Thus, the second and third terms in (B.12) cancel, leaving us with

$$\begin{aligned}
\text{var}_\pi[\hat{\mu}_g|\mathbf{z}] &\approx \frac{1}{N_p} E_\pi[w^{*2}|\mathbf{z}] \text{var}_p[g|\mathbf{z}], \\
&= \frac{1}{N_p} \text{var}_p[g|\mathbf{z}] (\text{var}_\pi[w^*|\mathbf{z}] + 1). \tag{B.13}
\end{aligned}$$

Recall, we wish to choose N_p such that $\text{var}_\pi[\hat{\mu}_g|\mathbf{z}]$ is small relative to the true

variance $\text{var}_p[g|\mathbf{z}]$. Thus, the ratio

$$\frac{\text{var}_\pi[\hat{\mu}_g|\mathbf{z}]}{\text{var}_p[g|\mathbf{z}]} \approx \frac{\frac{1}{N_p} \text{var}_p[g|\mathbf{z}] (\text{var}_\pi[w^*|\mathbf{z}] + 1)}{\text{var}_p[g|\mathbf{z}]} = \frac{\text{var}_\pi[w^*|\mathbf{z}] + 1}{N_p} \quad (\text{B.14})$$

should be a small number, say ϵ . The effective sample size N_{eff} is defined as the reciprocal of the approximation in (B.14). Then, we have the following (approximate) duality:

$$\frac{\text{var}_\pi[\hat{\mu}_g|\mathbf{z}]}{\text{var}_p[g|\mathbf{z}]} < \epsilon \iff N_{eff} \triangleq \frac{N_p}{\text{var}_\pi[w^*|\mathbf{z}] + 1} > \frac{1}{\epsilon}.$$

Thus, if we wanted the Monte Carlo variation to be less than 1% of the variance of g , we should require that $N_{eff} > 100$ after each measurement update.

The use of N_{eff} as a monitor of filter performance relies on the validity of the two Taylor series approximations used in the derivation. It is straightforward to show that the residual in the approximation used in (B.10) is $E_p[(w^* - \mu_w)(g - \mu_g)^2|\mathbf{z}]$. Thus, the use of N_{eff} (or \hat{N}_{eff}) as an indicator of the ratio $\text{var}_\pi[\hat{\mu}_g|\mathbf{z}]/\text{var}_p[g|\mathbf{z}]$ can be misleading if $g(\mathbf{x})$ differs substantially from μ_g in regions of the state space where $p(\mathbf{x}|\mathbf{z})$ is significant. However, this approximation does have the benefit of yielding an effective sample size N_{eff} that is independent of the function g .

REFERENCES

- [1] M. I. Miller, A. Srivastava, and U. Grenander, “Conditional-mean estimation via jump-diffusion processes in multiple target tracking/recognition,” *IEEE Transactions on Signal Processing*, vol. 43, pp. 2678–2690, November 1995.
- [2] A. D. Lanterman, “Tracking and recognition of airborne targets via commercial television and FM radio signals,” in *Proceedings of SPIE Acquisition, Tracking, and Pointing*, vol. 3692, 1999, pp. 189–198.
- [3] S. Challa and G. W. Pulford, “Joint target tracking and classification using radar and ESM sensors,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 37, pp. 1039–1055, July 2001.
- [4] Y. Boers and H. Driessen, “Integrated tracking and classification: An application of hybrid state estimation,” in *Proceedings of SPIE Signal and Data Processing of Small Targets*, vol. 4473, 2001, pp. 198–209.
- [5] S. Herman and P. Moulin, “A particle filtering approach to FM-band passive radar tracking and automatic target recognition,” in *Proceedings of the IEEE Aerospace Conference*, vol. 4, 2002, pp. 1789–1808.
- [6] N. Gordon, S. Maskell, and T. Kirubarajan, “Efficient particle filters for joint tracking and classification,” in *Proceedings of SPIE Signal and Data Processing of Small Targets*, vol. 4728, 2002, pp. 439–449.
- [7] J. D. Kendrick, P. S. Maybeck, and J. G. Reid, “Estimation of aircraft target motion using orientation measurements,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 17, pp. 254–259, March 1981.
- [8] D. Andrisani, F. P. Kuhl, and D. Gleason, “A nonlinear tracker using attitude measurements,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 22, pp. 533–538, September 1986.
- [9] A. F. M. Smith and A. E. Gelfand, “Bayesian statistics without tears: A sampling-resampling perspective,” *The American Statistician*, vol. 46, pp. 84–88, May 1992.
- [10] A. E. Gelfand and A. F. M. Smith, “Sampling-based approaches to calculating marginal densities,” *Journal of the American Statistical Association*, vol. 85, pp. 398–409, June 1990.

- [11] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *IEE Proceedings-F*, vol. 140, pp. 107–113, April 1993.
- [12] N. J. Gordon and A. Whitby, "Bayesian approach to target tracking in the presence of glint," in *Proceedings of SPIE Signal and Data Processing of Small Targets*, vol. 2561, 1995, pp. 472–483.
- [13] 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.
- [14] D. Avitzour, "Stochastic simulation Bayesian approach to multitarget tracking," *IEE Proceedings-F*, vol. 142, pp. 41–44, April 1995.
- [15] A. Srivastava, "Bayesian filtering for tracking pose and location of rigid targets," in *Proceedings of SPIE Signal Processing, Sensor Fusion, and Target Recognition*, vol. 4052, 2000, pp. 160–171.
- [16] J. Baniak, G. Baker, A. M. Cunningham, and L. Martin, "Silent Sentry™ passive surveillance," Lockheed Martin Mission Systems, Gaithersburg, MD, Tech. Rep., June 1999.
- [17] H. D. Griffiths and N. R. W. Long, "Television-based bistatic radar," *IEE Proceedings-F*, vol. 133, pp. 649–657, December 1986.
- [18] H. Kuschel, "VHF/UHF radar part 1: Characteristics," *Electronics and Communication Engineering Journal*, vol. 14, pp. 61–72, April 2002.
- [19] A. W. Rihaczek, *Principles of High-Resolution Radar*. New York, NY: McGraw-Hill, 1969.
- [20] J. L. Eaves and E. K. Reedy, *Principles of Modern Radar*. New York, NY: Kluwer Academic, 1987.
- [21] J.-S. Chen, "Relative amplitude and phase features for resonance-region radar target identification," in *Proceedings of the IEEE National Aerospace and Electronics Conference*, 1989, pp. 97–101.
- [22] A. A. Ksienski, Y.-T. Lin, and L. J. White, "Low-frequency approach to target identification," *Proceedings of the IEEE*, vol. 63, pp. 1651–1660, December 1975.
- [23] R. F. Harrington, *Field Computation by Moment Methods*. New York, NY: MacMillan, 1968.
- [24] S. M. Rao, D. R. Wilton, and A. W. Glisson, "Electromagnetic scattering by surfaces of arbitrary shape," *IEEE Transactions on Antennas and Propagation*, vol. 30, pp. 409–418, May 1982.

- [25] J. M. Song and W. C. Chew, "Multilevel fast-multipole algorithm for solving combined field integral equations of electromagnetic scattering," *Microwave and Optical Technology Letters*, vol. 10, pp. 14–19, September 1995.
- [26] Center for Computational Electromagnetics, University of Illinois and Demaco Inc., Champaign-Urbana, IL, *User's Manual for FISC (Fast Illinois Solver Code)*, October 1996.
- [27] L. C. Trintinalia, R. Bhalla, and H. Ling, "Scattering center parameterization of wide-angle backscattered data using adaptive Gaussian representation," *IEEE Transactions on Antennas and Propagation*, vol. 45, pp. 1664–1668, November 1997.
- [28] Y. Bresler, "Two-filter formulae for discrete-time non-linear Bayesian smoothing," *International Journal of Control*, vol. 43, no. 2, pp. 629–641, 1986.
- [29] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*. Englewood Cliffs, NJ: Prentice Hall, 1993.
- [30] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, pp. 257–285, February 1989.
- [31] A. Gelb, Ed., *Applied Optimal Estimation*. Cambridge, MA: M.I.T. Press, 1974.
- [32] Y. Bar-Shalom and T. E. Fortmann, *Tracking and Data Association*. Orlando, FL: Academic Press, 1988.
- [33] A. H. Jazwinski, *Stochastic Processes and Filtering Theory*, vol. 64 of *Mathematics in Science and Engineering*. San Diego, CA: Academic Press, 1970.
- [34] M. Athans, R. P. Wishner, and A. Bertolini, "Suboptimal state estimation for continuous-time nonlinear systems from discrete noisy measurements," *IEEE Transactions on Automatic Control*, vol. 13, pp. 504–514, October 1968.
- [35] P. S. Maybeck, *Stochastic Models, Estimation, and Control*. Arlington, VA: Navtech Book and Software Store, 1994.
- [36] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, pp. 174–188, February 2002.
- [37] N. H. Gholson and R. L. Moose, "Maneuvering target tracking using adaptive state estimation," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 13, pp. 310–317, May 1977.
- [38] Y. Bar-Shalom and K. Birmiwal, "Variable dimension filter for maneuvering target tracking," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 18, pp. 621–628, September 1982.

- [39] H. A. P. Blom and Y. Bar-Shalom, "The interactive multiple model algorithm for systems with Markovian switching coefficients," *IEEE Transactions on Automatic Control*, vol. 33, pp. 780–783, August 1988.
- [40] G. A. Watson and W. D. Blair, "IMM algorithm for tracking targets that maneuver through coordinated turns," in *Proceedings of SPIE Signal and Data Processing of Small Targets*, vol. 1698, 1992, pp. 236–247.
- [41] S. Julier, J. Uhlmann, and H. F. Durrant-Whyte, "A new method for the nonlinear transformation of means and covariances in filters and estimators," *IEEE Transactions on Automatic Control*, vol. 45, pp. 477–482, March 2000.
- [42] S. Julier and J. K. Uhlmann, "A general method for approximating nonlinear transformations of probability distributions," Robotics Research Group, University of Oxford, Tech. Rep., November 1996.
- [43] D. L. Alspach and H. W. Sorenson, "Nonlinear Bayesian estimation using Gaussian sum approximations," *IEEE Transactions on Automatic Control*, vol. 17, pp. 439–448, August 1972.
- [44] K. Ito and K. Xiong, "Gaussian filters for nonlinear filtering problems," *IEEE Transactions on Automatic Control*, vol. 45, pp. 910–927, May 2000.
- [45] A. Doucet, "On sequential simulation-based methods for Bayesian filtering," Cambridge University Engineering Department, Tech. Rep. CUED/F-INFENG/TR-310, 1998.
- [46] J. S. Liu, *Monte Carlo Strategies in Scientific Computing*. New York, NY: Springer Verlag, 2001.
- [47] D. Crisan and A. Doucet, "A survey of convergence results on particle filtering methods for practitioners," *IEEE Transactions on Signal Processing*, vol. 50, pp. 736–746, March 2002.
- [48] J. S. Liu, "Metropolized independent sampling with comparisons to rejection sampling and importance sampling," *Statistics and Computing*, vol. 6, pp. 113–119, 1996.
- [49] J. von Neumann, "Various techniques used in connection with random digits," *National Bureau of Standards, Applied Mathematics Series*, vol. 12, pp. 36–38, 1951.
- [50] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equations of state calculations by fast computing machines," *Journal of Chemical Physics*, vol. 21, pp. 1087–1091, 1953.
- [51] W. K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika*, vol. 57, no. 1, pp. 97–109, 1970.

- [52] A. W. Marshall, “The use of multi-stage sampling schemes in Monte Carlo computations,” in *Symposium on Monte Carlo Methods*. New York, NY: Wiley, 1956, pp. 123–140.
- [53] A. Kong, J. S. Liu, and W. H. Wong, “Sequential imputations and Bayesian missing data problems,” *Journal of the American Statistical Association*, vol. 89, pp. 278–288, March 1994.
- [54] M. Pitt and N. Shephard, “Filtering via simulation: Auxiliary particle filters,” *Journal of the American Statistical Association*, vol. 94, no. 446, pp. 590–599, 1999.
- [55] Y. Rui and Y. Chen, “Better proposal distributions: Object tracking using the unscented particle filter,” in *Proceedings of the IEEE CVPR*, vol. 2, 2001, pp. 786–793.
- [56] R. van der Merwe, A. Doucet, N. de Freitas, and E. Wan, “The unscented particle filter,” Cambridge University Engineering Department, Tech. Rep. CUED/F-INFENG/TR-380, August 2000.
- [57] W. R. Gilks and C. Berzuini, “Following a moving target — Monte Carlo inference for dynamic Bayesian models,” *Journal of the Royal Statistical Society B*, vol. 63, no. 1, pp. 127–146, 2001.
- [58] C. Hue, J.-P. Le Cadre, and P. Pérez, “Sequential Monte Carlo methods for multiple target tracking and data fusion,” *IEEE Transactions on Signal Processing*, vol. 50, pp. 309–325, February 2002.
- [59] J. S. Liu and R. Chen, “Sequential Monte Carlo methods for dynamic systems,” *Journal of the American Statistical Association*, vol. 93, pp. 1032–1044, September 1998.
- [60] G. Kitagawa, “Monte Carlo filter and smoother for non-Gaussian non-linear state space models,” *Journal of Computational and Graphical Statistics*, vol. 5, no. 1, pp. 1–25, 1996.
- [61] D. Crisan, “Particle filters — A theoretical perspective,” in *Sequential Monte Carlo Methods in Practice*. New York, NY: Springer, 2001, pp. 17–41.
- [62] J. Carpenter, P. Clifford, and P. Fearnhead, “Improved particle filter for nonlinear problems,” *IEE Proceedings–Radar, Sonar, and Navigation*, vol. 146, pp. 2–7, February 1999.
- [63] D. Crisan, P. Del Moral, and T. Lyons, “Discrete filtering using branching and interacting particle systems,” *Markov Processes and Related Fields*, vol. 5, no. 3, pp. 293–318, 1999.
- [64] S. Blackman and R. Popoli, *Design and Analysis of Modern Tracking Systems*. Norwood, MA: Artech House, 1999.

- [65] A. Marrs, S. Maskell, and Y. Bar-Shalom, "Expected likelihood for tracking in clutter with particle filters," in *Proceedings of SPIE Signal and Data Processing of Small Targets*, vol. 4728, 2002, pp. 230–239.
- [66] S. Mori, C.-Y. Chong, E. Tse, and R. P. Wishner, "Tracking and classifying multiple targets without a priori identification," *IEEE Transactions on Automatic Control*, vol. 31, pp. 401–409, May 1986.
- [67] D. B. Reid, "An algorithm for tracking multiple targets," *IEEE Transactions on Automatic Control*, vol. 24, pp. 843–854, December 1979.
- [68] Y. Bar-Shalom and E. Tse, "Tracking in a cluttered environment with probabilistic data association," *Automatica*, vol. 11, pp. 451–460, 1975.
- [69] Y. Bar-Shalom and A. G. Jaffer, "Adaptive nonlinear filtering for tracking with measurements of uncertain origin," in *Proceedings of the 1972 IEEE Conference on Decision and Control*, 1972, pp. 243–247.
- [70] T. E. Fortmann, Y. Bar-Shalom, and M. Scheffe, "Sonar tracking of multiple targets using joint probabilistic data association," *IEEE Journal of Oceanic Engineering*, vol. OE-8, pp. 173–184, July 1983.
- [71] M. L. Miller, H. S. Stone, and I. J. Cox, "Optimizing Murty's ranked assignment method," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 33, pp. 851–862, July 1997.
- [72] B. Friedland, *Control System Design: An Introduction to State-Space Methods*. New York, NY: McGraw-Hill, 1986.
- [73] B. Etkin, *Dynamics of Atmospheric Flight*. New York, NY: John Wiley and Sons, 1972.
- [74] K. L. Ross, "Coordinated flight," 2001, <http://www.friesian.com/flight.htm>.
- [75] M. Orton and W. Fitzgerald, "A Bayesian approach to tracking multiple targets using sensor arrays and particle filters," *IEEE Transactions on Signal Processing*, vol. 50, pp. 216–223, February 2002.
- [76] J. I. Glaser, "Some results in the bistatic radar cross section (RCS) of complex objects," *Proceedings of the IEEE*, vol. 77, pp. 639–648, May 1989.
- [77] L. C. Potter, D.-M. Chiang, R. Carrière, and M. J. Gerry, "A GTD-based parametric model for radar scattering," *IEEE Transactions on Antennas and Propagation*, vol. 43, pp. 1058–1066, October 1995.
- [78] A. Doucet, N. J. Gordon, and V. Krishnamurthy, "Particle filters for state estimation of jump Markov linear systems," *IEEE Transactions on Signal Processing*, vol. 49, pp. 613–624, March 2001.

VITA

Shawn Michael Herman was born in Evanston, Illinois, on July 8, 1972. He received the B.S. and M.S. degrees in Electrical Engineering from the University of Illinois at Urbana-Champaign (UIUC) in 1994 and 1996, respectively. From 1996 to 1998, he worked in the Speech Recognition Group at Lucent Technologies in Naperville, Illinois. In 1998, he returned to UIUC to pursue his Ph.D. in electrical engineering. During the summers of 1999 and 2000, he interned with the Passive Coherent Location Department at Lockheed Martin Mission Systems in Gaithersburg, Maryland. Following the completion of his Ph.D., he will begin work as a research engineer at Numerica, Inc., in Fort Collins, Colorado.