Target Tracking in Environments of Rapidly Changing Clutter

by

Karl Dutson

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved April 2015 by the
Graduate Supervisory Committee:

Antonia Papandreou-Suppappola, Chair
Daniel Bliss
Narayan Kovvali

ARIZONA STATE UNIVERSITY

May 2015

ABSTRACT

Tracking targets in the presence of clutter is inevitable, and presents many challenges. Additionally, rapid, drastic changes in clutter density between different environments or scenarios can make it even more difficult for tracking algorithms to adapt. A novel approach to target tracking in such dynamic clutter environments is proposed using a particle filter (PF) integrated with Interacting Multiple Models (IMMs) to compensate and adapt to the transition between different clutter densities. This model was implemented for the case of a monostatic sensor tracking a single target moving with constant velocity along a two-dimensional trajectory, which crossed between regions of drastically different clutter densities. Multiple combinations of clutter density transitions were considered, using up to three different clutter densities. It was shown that the integrated IMM PF algorithm outperforms traditional approaches such as the PF in terms of tracking results and performance. The minimal additional computational expense of including the IMM more than warrants the benefits of having it supplement and amplify the advantages of the PF.

ACKNOWLEDGMENTS

I'd like to thank Professor Antonia Papandreou-Suppappola for consistently being the greatest professor in the world. Special thanks to Brian O'Donnell for getting me started, Meng Zhou and Dr. Narayan Kovvali for assisting me along the way, and John Kota for helping me with the final phases of this work.

I thank my family for their continued support, sacrifices, and patience throughout the duration of my years in schooling, which have culminated in this exciting project and fascinating research endeavor. I express abundant gratitude to my family for standing by faithfully through so many years of my academic journey.

I thank all of the teachers, tutors, advisors, and leaders who have taught me, guided me on this path, and helped me accomplish things I never thought were possible.

And on that note, above all, I must thank again my diligent advisor, professor, and trusted friend, Antonia, to whom my gratitude is endless,
and to whom I will forever be indebted.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

2–D - Two–Dimensional

CD - Clutter Density

CRLB - Cramer-Rao Lower Bound

FA - False Alarm

FIM - Fisher Information Matrix

IMM - Interacting Multiple Model

IMM PF - Integrated PF and IMM algorithm

MSE - Mean-Squared Error

PDA - Probabilistic Data Association

PDF - Probability Distribution Function

PF - Particle Filter

RMSE - Root Mean-Squared Error

SIR-PF - Sequential Importance Resampling PF

SNR - Signal-to-Noise Ratio

UKF - Unscented Kalman Filter

Chapter 1

INTRODUCTION

1.1 Motivation and Background

A common and inevitable obstacle in most realistic target tracking scenarios is the presence of clutter, or unwanted signal reflections. The term clutter is a general reference to anything other than the target of interest. One could imagine tracking an aircraft, and in this instance some examples of clutter could be weather, trees, other aircraft, birds, rain, mountains, towers, and countless other possibilities. Perhaps less commonly, in more unique and less conventional tracking mediums, clutter could even be rocks, debris, fish (echolocation for subaquatic targets and tracking), buildings (tracking low flying or ground restricted targets in urban environments), waves or water (a land-based sensor tracking ships at sea), and soil discontinuities (for ground penetrating radar or GPR) (Guo, 2008; Sira, 2006; El-Shenawee, 2002; Takahashi, 2011).

Clutter is problematic and challenging because pulse reflections that bounce off objects other than the target can be misinterpreted as having been reflected from the target (or targets) of interest. Thus, the theoretically ideal case would be a complete lack of clutter, such that the target is the only non–negligible object in the tracking environment, and therefore all reflections of the transmitted signal could be guaranteed returns from the target. This, of course, is rarely the case in any

practical setting. Moreover, as the number of nearby clutters increases, so too does the probability that a received signal echo is not from the desired target.

Furthermore, drastic, sudden changes in clutter density can make tracking even more difficult. Exploring the solution to this problem is the essence of this thesis. To this end, in this thesis, we consider neighboring environments, each with vastly different clutter densities, and attempt to track a target passing through said environments.

Considering this latter problem, one can imagine that there may be many possible target tracking applications where different models are needed at different time steps. Perhaps the most standard example is that of a target moving with constant velocity which then turns, signifying a transition between a state of constant velocity to a state of changing velocity and constant acceleration (Boers, 2003). Other diverse examples exist, many of which involve different clutter frequencies, densities, or types (Mazor, 1998).

One particular such example is found within the scope of Ground Penetrating Radar (GPR) research and applications, in which soil heterogeneity varies across a continuum (Takahashi, 2011). Different degrees of soil heterogeneity produce varying amounts of unwanted reflections from clutter, and transitions across different regions or soil layers in which soil density changes can make the effects from this type of clutter even more pronounced.

One approach to handle and accurately model such systems, where clutter density variation is inevitable, is to employ different models at different times, using a hybrid state estimation scheme. The term for such an approach is an “Interacting Multiple Model” (IMM). Using IMMs entails mode switching, where an algorithm

switches or selects one of multiple models or "modes" based on certain conditions, parameters, or thresholds. One of the major challenges in this arena is for the researcher to consider all of the intricacies in the problem and choose the most balanced sensitivity such that the right model(s) work in the right case(s).

Additionally, an IMM can function as a "*self–adjusting variable bandwidth filter*," which lends itself well to tracking moving targets (Mazor, 1998). Moreover, IMMs have been shown to be extremely cost–effective, flexible estimators that can offer high performance for low computational demands (Mazor, 1998). As such, IMMs have been applied to a wide assortment of tracking problems. Some prominent and diverse application areas where multiple models have been used include financial engineering, motion analysis in computer vision, and home insurance fraud detection (Vladimir, 2000). In spite of this, no work has been seen on using an IMM in the context of clutter variation.

We propose a novel approach to target tracking in dynamic clutter environments using a particle filter with an integrated IMM to account for the transition between different clutter densities. It accounts for clutter variation with the ability to dynamically switch between different models in response to environmental changes. The IMM PF will work by initializing particles mostly into one mode, then switching modes based on changes in clutter (according to predetermined sensitivity settings based on the application). This will be explained in further detail later.

1.2 Organization

This thesis is organized as follows: in Chapter 2, we provide a brief overview of the state space representation for nonlinear target tracking using particle filters, we explore the difficulties of compensating for clutter, and the potential for using multiple models to improve tracking performance in the wake of variable clutter densities. In Chapter 3, we outline the problem of environments that have non-static clutter densities, and propose how to use IMMs to improve tracking performance under such dynamically changing environmental conditions. Chapter 4 provides the simulation results of all these efforts, and Chapter 5 concludes the thesis with a summary of our findings and the directions for future work.

Chapter 2

TARGET TRACKING FORMULATION AND BAYESIAN ESTIMATION

2.1 State Space Representation for Target Tracking

The scenario considered and explored in this thesis is that of a single sensor tracking a single target moving with constant velocity along a two-dimensional (2-D) trajectory.

The coordinates for the position of the target in a 2–D plane at time step k are given by (x_k, y_k) , and $(\dot{x}_k, \dot{y}_k)$ are the coordinates for the velocity components of the target at the same time step. Thus, the state of the target at a given time step k can be represented by the 4×1 state vector

$$\mathbf{x}_k = [x_k \;\; y_k \;\; \dot{x}_k \;\; \dot{y}_k]^T ,$$

where T denotes the vector transpose, and $\mathbf{x}_k$ represents the state of the entire system at time step k. Since we are assuming the target moves with constant velocity, we can invoke a linear, discrete–time model, and the state space representation of the system is then given by the following state equation:

$$\mathbf{x}_k = \mathbf{F}\,\mathbf{x}_{k-1} + \mathbf{w}_k \, .$$

The constant matrix F in this equation is given by:

$$\mathrm{F} = \begin{bmatrix} 1 & \delta t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \delta t \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

where δt is the time duration between state transitions.

The vector $\mathbf{w}_k$ is the error model random process that is usually assumed to be a zero–mean uncorrelated Gaussian noise vector, whose process noise covariance matrix Q is given by

$$\mathbf{Q} = q \begin{bmatrix} \frac{\delta t^3}{3} & 0 & \frac{\delta t^2}{2} & 0 \\ 0 & \frac{\delta t^3}{3} & 0 & \frac{\delta t^2}{2} \\ \frac{\delta t^2}{2} & 0 & \delta t & 0 \\ 0 & \frac{\delta t^2}{2} & 0 & \delta t \end{bmatrix},$$

where q is a noise scaling constant.

At each time step k, the transmitted signal reflected off the moving target is received with a time delay τ_k and a Doppler shift ν_k. This information is used to infer the target's range and range rate (Richards, 2010). The time it takes the signal to bounce off the target and return to the transmitter, τ_k, is proportional to its range, which is thus given by $r_k = c\tau_k/2$. Equivalently, its range rate is determined from the echoed Doppler shift according to $\dot{r}_k = c\nu_k/2f_c$, where f_c is the carrier

frequency and c is the speed of light in vacuum, which for our radar application we assume is the velocity of the transmitted signal's propagation (Richards, 2010).

We consider the most general, arbitrary representation and then modify it for our specific case. In general, the measurement is represented as $\mathrm{z}_k = h_k\left(\mathrm{x}_k, \mathrm{v}_k\right)$. Assuming perfect detection, the measurement originated from the target can be given by $\mathrm{z}_k = h_k\left(\mathrm{x}_k\right) + \mathrm{v}_k$. The random process $\mathbf{v}_k$ is the measurement noise vector; it is assumed to be a zero–mean additive Gaussian random noise vector with covariance matrix $\mathbf{R}$. We make the assumption that the measurement noise represented by $\mathbf{v}_k$ is uncorrelated with the process noise $\mathbf{w}_k$.

If our relationship were linear, this would become

$$\mathrm{z}_k = \mathrm{H}\mathrm{x}_k + \mathrm{v}_k ,$$

and the matrix H would simply be the identity matrix. However, in our case, the relationship is non–linear, and $\mathrm{z}_k = h_k\left(\mathrm{x}_k, \mathrm{v}_k\right)$ reflects that the function $h\left(\mathrm{x}_k, \mathrm{v}_k\right)\colon \mathbb{R}^4 \to \mathbb{R}^3$ represents the nonlinear relationship between position and velocity, and range and range rate:

$$h\left(\mathrm{x}_k, \mathrm{v}_k\right)\colon \begin{bmatrix} x_k \\ y_k \\ \dot{x}_k \\ \dot{y}_k \end{bmatrix} + \mathrm{v}_k \quad \to \quad \begin{bmatrix} r_k \\ \dot{r}_k \\ \theta_k \end{bmatrix} + \mathrm{v}_k .$$

This function maps from Cartesian coordinates to range, range rate, and bearing angle, and it does so via the following relationships (van Trees, 1992; Bourgeois, 2007):

$$
\begin{aligned}
r_k &= \sqrt{(x_k - x^{(s)})^2 + (y_k - y^{(s)})^2} \\
\dot{r}_k &= (\dot{x}_k(x_k - x^{(s)}) + \dot{y}_k(y_k - y^{(s)}))/r_k \\
\theta_k &= \arctan\left(\frac{y_k}{x_k}\right).
\end{aligned}
$$

Here, $(x^{(s)}, y^{(s)})$ is the 2-D position of each of the s possible radar sensors in a surveillance region. For our application, we assume a monostatic radar located in the 2-D plane, and, for simplicity, we assume the radar's position coordinates to be at the origin. Applying these simplifications, the equation pair above reduces to:

$$
\begin{aligned}
r_k &= \sqrt{x_k^2 + y_k^2} \\
\dot{r}_k &= \left(\dot{x}_k x_k + \dot{y}_k y_k\right)/r_k
\end{aligned}
$$

Putting this all together, our entire system is represented by:

$$
\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ \dot{x}_{k+1} \\ \dot{y}_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & \delta t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \delta t \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_k \\ y_k \\ \dot{x}_k \\ \dot{y}_k \end{bmatrix} + q \begin{bmatrix} \frac{\delta t^3}{3} & 0 & \frac{\delta t^2}{2} & 0 \\ 0 & \frac{\delta t^3}{3} & 0 & \frac{\delta t^2}{2} \\ \frac{\delta t^2}{2} & 0 & \delta t & 0 \\ 0 & \frac{\delta t^2}{2} & 0 & \delta t \end{bmatrix}
$$

And the generalized Markov process state representation:

$$
\begin{aligned}
x_k &= f_k\left(\mathrm{x}_{k-1}, \mathrm{w}_{k-1}\right) \\
\mathrm{z}_k^{(s)} &= h_k^{(s)}\left(\mathrm{x}_k, \mathrm{v}_k\right)
\end{aligned}
$$

becomes the more simplified system as:

$$
\begin{aligned}
\mathrm{x}_k &= \mathrm{F}\mathrm{x}_{k-1} + \mathrm{w}_k \\
\mathrm{z}_k &= h\left(\mathrm{x}_k\right) + \mathrm{v}_k
\end{aligned}.
$$

2.2 Particle Filter

A particle filter (PF) works by generating hundreds or thousands of hypotheses (as to the target's state coordinates), evaluating each one, then repeating and refining the process.

It is important to clarify the term "particle." By particle we are simply referring to one of many unique hypotheses generated. Because so many hypotheses are in play at once – usually thousands – it is generally easier to conceptualize and visualize each of these unique hypotheses as a point in the state space.

Implementing a PF for the purpose of tracking generally requires the researcher to do so along four major steps. First, an initial assumption is made on the probability density function (PDF) of the state at time step $k = 0$ and samples or particles are drawn from the PDF; the PDF is usually assumed to be uniform over the expected values for range and range rate. Next, the likelihood of each particle is evaluated with respect to the previous measurement of the target. Then, numerical weights are assigned according to how likely each particle is relative to, and based upon, the current measurement of the target's state. Particles that seem unlikely (i.e., too far away from the last measurement, or with a velocity that is too high) are labeled with a small weight value, and likewise, more probable particles are given higher values. These procedures occur at each time step, and are repeated at each time step thereafter.

The fourth and final step is a very critical one, and is known as resampling. The resampling sub–algorithm sifts through all of the current particles, and evaluates each one based upon its weight assigned in the previous time step.

Particles with lower weights are discarded, while those with higher weights are kept for posterity, before a new batch of particles (hypotheses) are generated. However, this time the generation step is not done randomly throughout the entire space, as in the first step. Rather, the next batch of N_s particles is generated based upon the remaining particles and their (higher) weights. In short, this resampling step effectively discards all of the insignificant particles, and substitutes them with clones of the "good" particles. The details of how to effectively implement the resampling step are provided in the Algorithm 1 pseudocode.

Once the particles are regenerated and a full new set of N_s hypotheses are in effect, the weights are reset so the reevaluation process can be repeated.
Hence it is necessary for resampling to occur at every time step, along with all of the other calculations outlined above. For this reason, this type of Particle Filter is known as a Sequential Importance Resampling Filter (SIR–PF).

Algorithm 1 Resampling Algorithm

$\left\{\mathbf{x}_k^{*(j)}, \tilde{w}_k^{(j)}\right\}_{j=1}^{N_s} = \textbf{Resample}\left(\left\{\mathbf{x}_k^{(i)}, w_k^{(i)}\right\}_{i=1}^{N_s}\right)$

Let $c_1 = w_k^{(1)}$

for $i = 2$ to N_s **do**

  $c_i = c_{i-1} + w_k^i$

end for

Let $m = 1$

Draw $u_1 \sim \mathcal{U}(0, 1/N_s)$

for $j = 1$ to N_s **do**

  $u_j = u_1 + (j-1)N_s^{-1}$

  while $u_j > c_m$ **do**

    $m = m + 1$

  end while

  $x_k^{*(j)} = x_k^{(m)}$

  $\tilde{w}_k^{(j)} = N_s^{-1}$

end for

return

In this way, particles that are likely "survive" and are propagated forward in the state space, according to the same dynamics that govern the behavior and motion of the target. Thus the state estimation process is continually refined at each time step, and the PF can quickly and accurately converge onto estimates that are near the true state (please see Figures 1–4 in Chapter 4, for immediate evidence/examples of this).

To summarize, the four main steps in the particle filter algorithm are:

1. Generate and distribute particles
2. Calculate likelihood values
3. Assign weights
4. Resample

Algorithm 2 provides the pseudocode on the computational implementation of these steps in much more detail.

Algorithm 2 Sequential Importance Resampling (SIR) Algorithm

$\left\{\mathbf{x}_k, \mathbf{x}_k^{*(j)}\right\}_{j=1}^{N_s} = \textbf{SIR}\left(\left\{\mathbf{x}_{k-1}^{(i)}\right\}_{i=1}^{N_s}, \mathbf{z}_k\right)$

for $i = 1$ to N_s **do**

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

Update Weights $w_k^{*(i)} = w_{k-1}^{(i)} p(\mathbf{z}_k|\mathbf{x}_k^{(i)})$

end for

Calculate $s = \sum_{i=1}^{N_s} w_k^{*(i)}$

for $i = 1$ to N_s **do**

Normalize Weights $w_k^{(i)} = s^{-1} w_k^{*(i)}$

end for

$\left\{\mathbf{x}_k^{*(j)}, \tilde{w}_k^{(j)}\right\}_{j=1}^{N_s} = \textbf{Resample}\left(\left\{\mathbf{x}_k^{(i)}, w_k^{(i)}\right\}_{i=1}^{N_s}\right)$

Estimate State $\hat{\mathbf{x}}_k = \sum_{j=1}^{N_s} \mathbf{x}_k^{*(j)} \tilde{w}_k^{(j)}$

return

The revelation of resampling was a very important step in the history and development of the PF. Implementing this SIR–PF algorithm at each time step k is now possible with modern computing advances, and it is nearly always the ideal choice for target tracking. Thus we will select the SIR–PF as our algorithm of choice for our application: the two–dimensional tracking of a single target over a region of rapidly or immediately changing clutter density.

2.3 Evaluating Clutter

In Chapter 1 we introduced clutter conceptually; we will now quantify clutter mathematically and symbolically.

Recall that clutter increases the probability that a received echo is a false positive (and thereby decreases the probability that a signal return is actually from the target). Moreover, spikes in clutter frequency ramp up the probability that a received signal echo is not from the desired target.

In this paper we consider the two–dimensional tracking of a single target over a region of rapidly or immediately changing clutter density, and the possibilities for dealing with such scenarios, and their associated clutter, will be explored. We consider the problem in the most general sense, before reducing it to our specific case of interest.

The amount of clutter in an environment is assumed to be distributed uniformly throughout, so clutter density can be interpreted as an average number of clutter objects per unit volume. During any discrete time step *k,* it is possible for a given sensor *s* to detect $m_k^{(s)}$ clutter objects, and with this the likelihood function is given by (Bar-Shalom, 1975):

$$p(\boldsymbol{Z}_k \mid \boldsymbol{x}_k) = (1 - P_{D,k})\Pr\left(m_k\right)V_k^{-m_k} + \frac{P_{D,k}\Pr(m_k - 1)V_k^{-(m_k-1)}}{m_k}\sum_{z_k \in \boldsymbol{Z}_k} p(z_k \mid x_k)$$

Here P_D is the probability of detection, such that $P_{D,k,s}$ is the probability of detection per sensor *s* at time step *k*. The quantity $\Pr\left(m_k^{(s)}\right)$ is the probability that $m_k^{(s)}$ measurements are from clutter, and is modeled using a Poisson random process, as per usual for probabilistic date association (Bar–Shalom, 2009):

$$\Pr\left(m_k^{(s)}\right) = \frac{1}{m_k^{(s)}!}\left(\rho V_k^{(s)}\right)^{m_k^{(s)}} e^{-\rho V_k^{(s)}}$$

The value ρ can be thought of as the clutter density, or the average number of clutter objects per unit volume. Throughout this paper, we will use ρ to indicate clutter density, and ρ_n will indicate the clutter density within region *n.* The quantity V_k above is the validation gate volume, which is computed according to (Musicki, 2004):

$$\begin{aligned} V_k &= \mathbf{vol}\left(\mathcal{V}_k^{(\gamma)}\right) \\ &= \mathbf{vol}\left(\left\{\mathbf{z}_k \in \mathbb{R}^{n_z} : (\mathbf{z}_k - \hat{\mathbf{z}}_k)^T \mathbf{S}_k^{-1} (\mathbf{z}_k - \hat{\mathbf{z}}_k) \le \gamma^2\right\}\right) \\ &= \gamma^2 \rho \pi \sqrt{\prod_{i=1}^{n_x} \lambda_i \left\{\mathbf{P}_{\mathbf{z}_k \mathbf{z}_k} + \mathbf{CRLB}\left(\boldsymbol{w}_k\right)\right\}} \end{aligned}$$

In words, this implies that V_k is proportional to the product of the eigenvalues of summing the Cramer–Rao Lower Bound (CRLB) and the measurement covariance. Here, the validation gate volume is a 2–D ellipsoid that encompasses the true measurement and all nearby clutters.

The scaling factor and standard deviation multiplier γ is typically assigned a value of 4 or 5, which is suitable for 99% of all cases (Fortmann, 1985). Throughout all calculations and simulations, we will use $\gamma = 5$. Note that in addition to γ, V_k also scales proportional to the clutter density ρ.

As uncertainty increases, the validation gate volume must expand in order to sufficiently compensate for and adequately secure the possibility of encompassing the true target with high probability. Likewise, as measurements become more confident, the validation gate volume will contract, as less volume is needed to surround the true measurement with high certainty, and thus the number of clutter objects it encompasses will decrease. In this sense, the number of clutters or clutter density does not change at each time step; rather, the validation gate volume simply expands or contracts to include more or less clutter, respectively. An important, practical note is that larger validation gate volumes correspond to exponentially increasing computation costs.

As one can see, increases in clutter frequency make the tracking scenario more challenging. We also mentioned previously that additionally, sudden changes in the amount of nearby clutter – i.e., clutter density – can make it even more difficult for tracking algorithms to adapt. If such changes occur, all of the equations in this section must be completely recalculated with a different value of ρ. And, this must be done immediately, at each time step, or else the estimates could be inaccurate or even meaningless. Exploring a potential solution to this problem is an objective of this thesis (now that we have laid out the theoretical framework).

Chapter 3

IMM AND CLUTTER

3.1 The Varying Clutter Problem

In Section 1.1 the concept of clutter was introduced. Section 2.3 outlined the mathematical framework for handling clutter and minimizing the number of False Alarms (FA). As we saw, mitigating the influence of clutter can be difficult. Throughout all of these descriptions, we only considered cases of static clutter density ρ. Unfortunately, this does not realistically reflect the environment of many applications. Ample settings present the problem of clutter levels ρ that change over time.

For instance, some examples might include an aircraft traveling in calm weather that abruptly enters a harsh storm, suddenly passes over (or through – consider a helicopter) a forest or city, or a boat or submarine that moves from calm waters to turbulent currents. Many of the examples of clutter given in Section 1.1 also present the problem of clutter variation, or could in certain situations.

An additional layer of complexity is added when either the current tracking environment changes, or when the target changes environments. So consider, to this end, two neighboring environments, each with different clutter densities, ρ_1 and ρ_2, such that $\rho_2 >> \rho_1$. This would prove quite difficult for the standard particle filter or other tracking algorithm to adapt to. However, an IMM PF could offer a potential solution by having two or more predetermined models – one (or more) for each of the

different clutter environments – and switching between them at the appropriate time, as the conditions change. Given that the difference between clutter densities ρ_1, ρ_2 is large (which it would be for most applications of this sort; otherwise the clutter change would not be so detrimental to the tracking), this disparity should be more than sufficient to provide a transition threshold strong enough to trigger the IMM into switching models.

We have shown that there is strong potential to support the utilization of an IMM with a PF. The adaptability of having multiple selectable models allows for much more versatility, and the ability to handle greater amounts of clutter overall. It also increases options for many possibilities and applications. These research results and observations are demonstrated in the next chapter, which summarizes our simulations and findings through representative example plots.

3.2 Integrating IMMs with the Particle Filter

In order to combine the PF with an IMM, several modifications were made to the existing, classic PF algorithm. We covered the mechanics of how this algorithm works in Section 2.2, as well as the state space representation in Section 2.1. To include IMMs, this state space representation is modified as described below.

Previously, the state space representation was held within a vector containing the positions and velocities of each particle at each time step. From hereon, each particle will now have an additional property associated with it: a designation of which mode it is in; i.e., which clutter density it is utilizing at that time step. For each environment considered, each particle – or state space hypothesis of the target's true state, taken from a distribution – will exist in one of these associated clutter modes or states. Thus, through each trial of the Monte Carlo simulation, for each particle, there is an associated mode state attached/assigned to it. The particle's state must exist in one of these modes, and the clutter density associated with it.

With these mode associations in place, the PF algorithm was modified in three major ways to accommodate the addition of IMMs. First, mode transition calculations were added at the beginning of the algorithm. Second, parts of the likelihood calculations were built upon to accommodate more than one clutter density in the computations. Finally, the PF's resampling sub-algorithm is extended so that mode states are resampled along with particle positions and velocities, such that only the particles with the most likely estimates in both regards are given greater weights and passed on. These changes are outlined in more detail below.

Without loss of generality, for discussion purposes, we will assume a tracking scenario and simulation with only two regions and two different clutter densities; but the changes described were also implemented for three regions and more.

Considering the first modification, mode change transitions, prior to the first time step of each simulation, particles are initialized into one of the available modes, based on a prior (but variable) probability. Then for each time step thereafter, through each trial of a Monte Carlo simulation, each particle transitions modes according to a two state Markov chain model that has prior probabilities assigned (see state diagram in Figure 3.1 below). For the majority of the simulations that were run, the random probability of switching modes was set to 5% (and thus the standalone probability of remaining in the same mode as the previous time step was 95%). Given these probabilities, the mode state may change at this point in the simulation during each time step; that is, mode assignments are not completely fixed, and there are fluctuations which make transitions possible. Other probabilities could be used, and were, such as a switching rate of 10%. A balance must be struck between having flexible switching potential and having a system that is too unstable due to transitions that occur too freely. For instance, setting both the probability of switching and that of remaining in the previous state to 50% would not allow the algorithm to work by any means; particles would switch randomly at each time step. For this reason, more conservative switching probabilities were used, such as 5% and 10%. An example case of the state space representation is given in Figure 3.2, for when the transition probability is set to 5%.

In both Figures 3.1 and 3.2, c_k is the clutter mode state of that particle at that time step, the π_{ab} values are mode transition probabilities (i.e., the probability of transitioning from state *a* to *b*), and π is the transition probability matrix.

Generalized State Representation of Mode Transitions

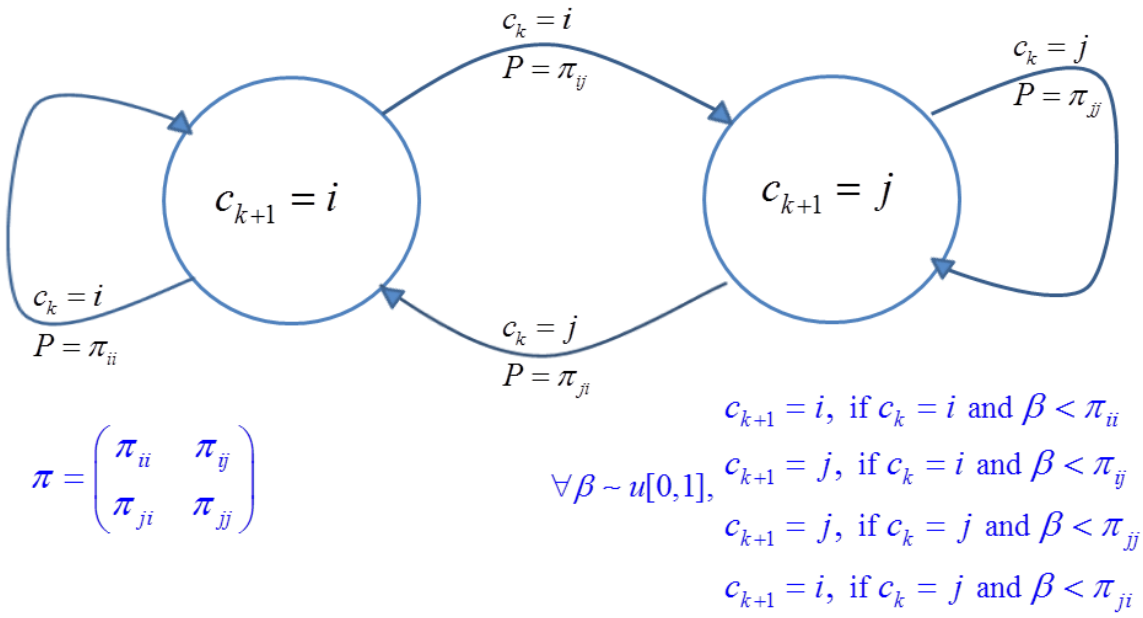


Figure 3.1 *Generalized state diagram illustrating mode transition probabilities for a two environment scenario, with two arbitrary, non-equal clutter densities and arbitrary probabilities of randomly switching. Here* c_k *is the clutter mode state of at that time step, the* π_{ab} *values are mode transition probabilities (i.e., transitioning from state a to b), and* π *is the transition probability matrix.*

Example Instance of State Representation of Mode Transitions

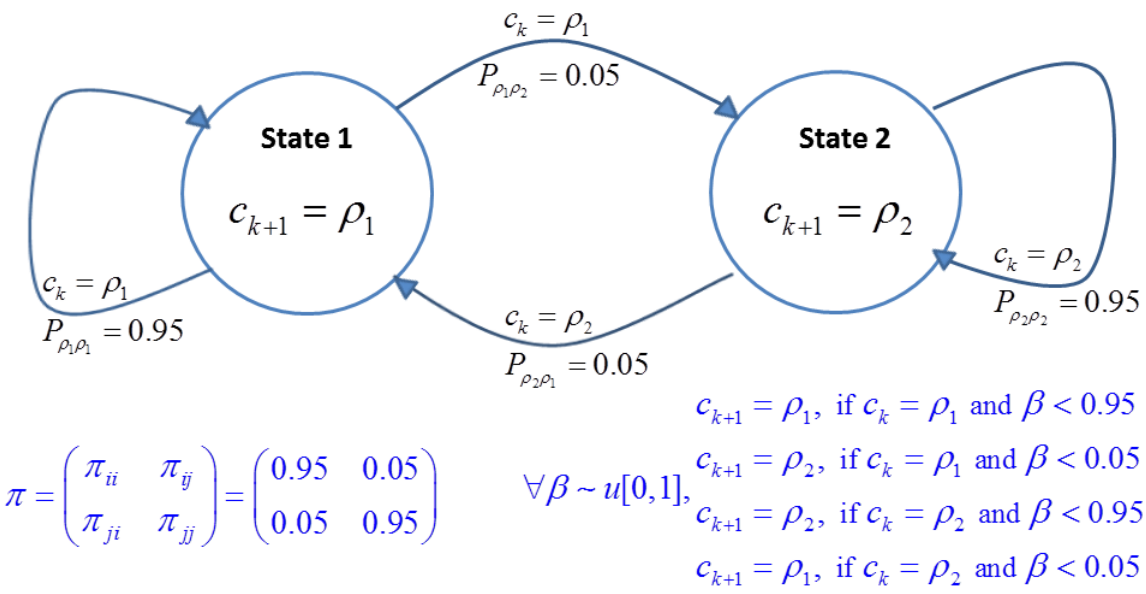


Figure 3.2 *An example state diagram illustrating mode switching probabilities for a particular two environment scenario, with two different clutter densities, and with probability of randomly switching set to 5%. Here c_k is the clutter mode state of at that time step, the π_{ab} values are mode transition probabilities (i.e., transitioning from state a to b), and π is the transition probability matrix.*

No additional changes were made to the PF algorithm until the point where the likelihood values are calculated, where another major addition/step takes place. The likelihood sum itself stays the same, and does not change. Thus, in the following likelihood function calculation (Bar-Shalom, 1975):

$$p(\boldsymbol{Z}_k \mid \boldsymbol{x}_k) = (1 - P_{D,k}) \Pr(m_k) V_k^{-m_k} + \frac{P_{D,k} \Pr(m_k - 1) V_k^{-(m_k - 1)}}{m_k} \sum_{z_k \in \boldsymbol{Z}_k} p(z_k \mid x_k)$$

The sum on the far right side of the equation is calculated as normal, but the coefficient and added terms change now with the IMM and new densities from

different clutter. So the full likelihood, and the probability that m_k measurements originate from clutter, is now dependent on the mode state (and thus the clutter density). Thus, each mode each particle is assigned to changes the values of $\Pr(m_k)$ and $\Pr(m_k - 1)$. For instance, the $\Pr(m_k)$ term becomes "m_k given ρ," such that we now have $\Pr(m_k \mid \rho_t)$, for a given clutter state or mode. This state is assigned and these probabilities are calculated based on which ρ is chosen to be most ideal.

Thus, during each iteration, the algorithm first determines which mode state that particle is currently associated with. Then, depending which mode state it is in, will calculate the FA probabilities accordingly. Hence the original FA probability splits into multiple models, as conceptually illustrated by the mathematical visualization in Figure 3.3, for a generalized number of clutter densities $\rho_1, \rho_2, \rho_3, \ldots \rho_n$.

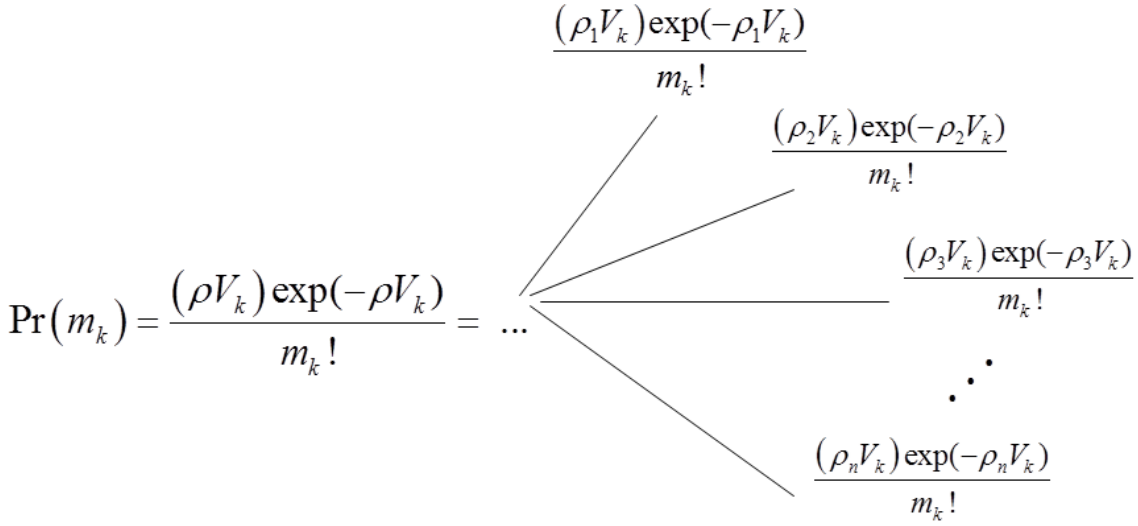


Figure 3.3 *A technically incorrect but visually representative depiction of the original probability FA/clutter detection expression splitting into multiple models, based on different clutter densities.*

More compactly, and mathematically, we have:

$$\Pr\left(m_k\right) = \frac{1}{m_k!}\left(\rho V_k\right)e^{-\rho V_k} \quad \rightarrow$$

$$\Pr\left(m_k \mid \rho_t\right) = \frac{\left(\rho_t V_k\right)\exp(-\rho_t V_k)}{m_k!}, \text{ for } t = 1, 2, \ldots, n,$$

where n is the total number of different clutter densities and associated environments. One of these multiple selectable models is chosen for that particle, based on which of the available clutter densities is being utilized at that time step. Using the calculated probability based on the mode state, the likelihood calculation proceeds as normal, as indicated in the equation above, for that particular clutter density.

Finally, the third modification addresses how the algorithm selects which clutter density (and thus which state) to use. The same inquiry could be posed of the original PF, concerning how it selects which state estimates are close to the target, and which are far away. In both cases, the answer is the resampling step. The new IMM PF algorithm determines which clutter mode to use during the resampling sub-routine. In the same way particles are resampled and reweighted based on accuracy, so too are the mode states. The state/clutter density value which is weighted most heavily based on the previous measurement is kept for posterity, and those which are least likely are thrown out; at the end of the resampling step most particles will be initialized into the state that is most likely. In this way, the PF weights are the fundamental moving gears for the PF (and the IMM PF), and resampling utilizes the PF weights to reassign the clutter states/modes.

In short, there are only a few places where additional IMM components are inserted into the original, standard PF algorithm (corresponding to a minimal increase in computational expense, yet a substantial gain in tracking versatility). These add-ons are implemented alongside the standard PF algorithm steps; specifically in the likelihood calculations, and during resampling. Additionally, there is an entirely new added step: the calculation of mode state transitions that follow a Markov chain model with prior probabilities assigned. These are the three major additions to the PF algorithm which make the IMM PF work.

Chapter 4

SIMULATIONS

To explore the PF and its potential use with IMM, a general PF algorithm was implemented using Matlab code to track synthetically generated target trajectory data, for a target moving in a 2-D plane. As such, four independent target parameters were tracked in each scenario: position and velocity in both x and y dimensions.

Performance/measurement errors were calculated by taking the difference of the measured values from that of the true state, at each time step, and then taking the square root of the absolute value of this devation. A sample plot of such errors calculated at each time step k is provided in Figure 4.1.

In addition to measuring performance, RMSE plots were used to determine the optimal number of particles to use in subsequent simulations, and to prove that this particle number was sufficient to validate future results (Figure 2). Multiple test simulations were run to show this using different amounts of particles; one such instance is provided below. In this case the particle number was varied from between 50 to 2,000 particles.

Figure 1 plots the RMSE vs. time, Figure 2 shows RMSE vs. time for different particle numbers, and Figures 3, 4 plot the actual, physical tracking error and deviations from the true state.

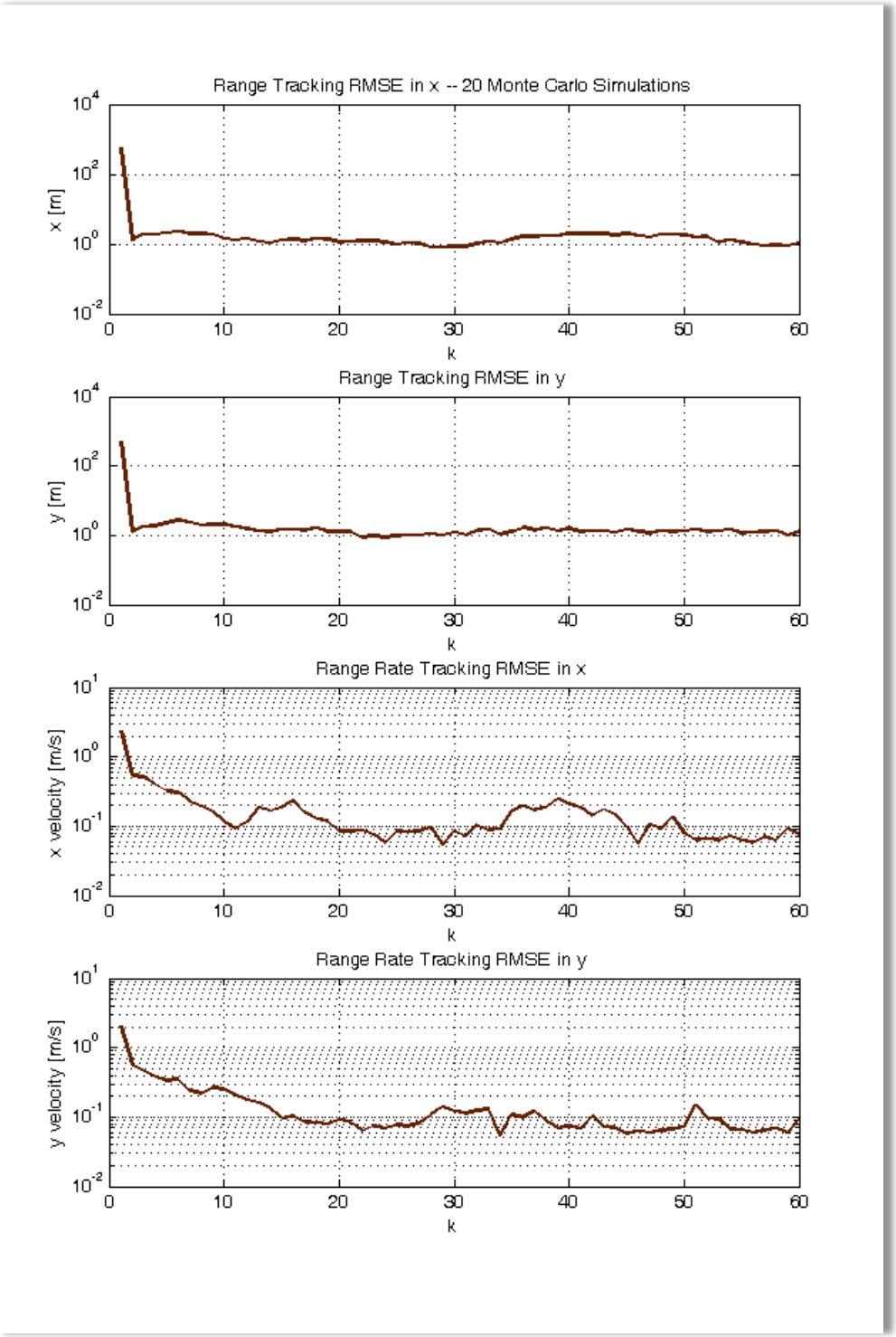


Figure 4.1 *Sample of an RMSE plot, averaged over 20 simulations. These errors are on the high end, but are representative of the error threshold seen throughout the simulations.*

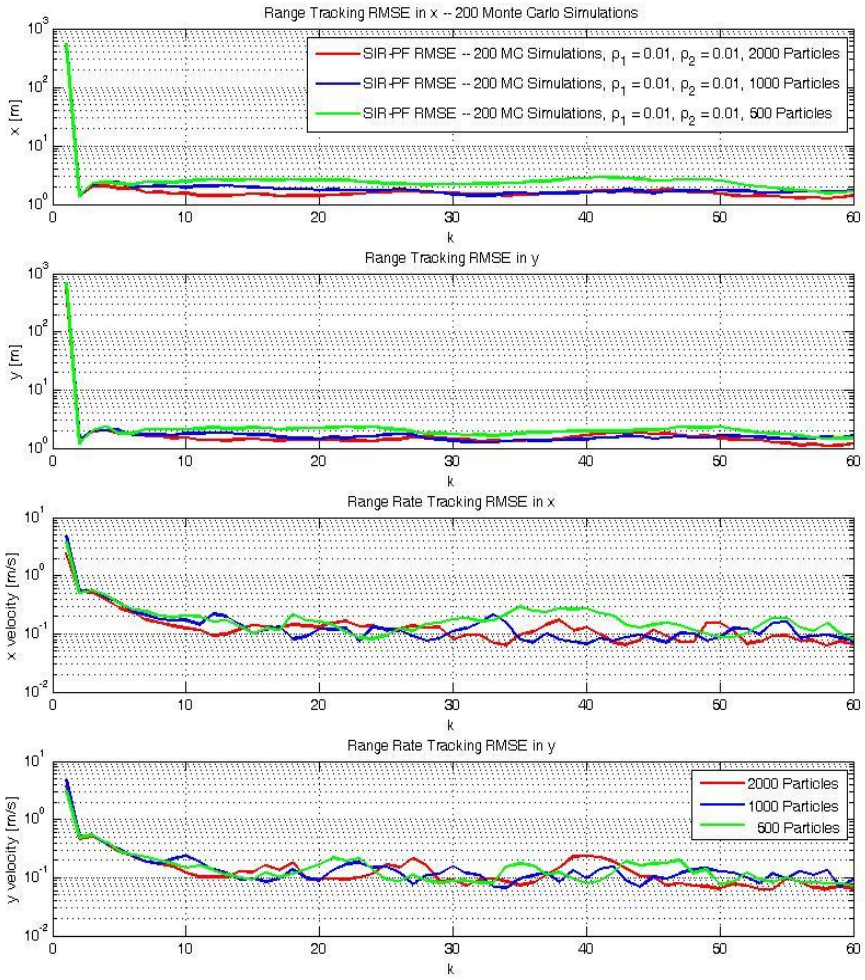


Figure 4.2 *Various RMSE results as a function of particle number. Figure shows that while there are significant decreases in error as particle number increases, the differences are likely negligible beyond 500, and definitely beyond 1000, and do not justify the additional computational costs. Therefore using 1000 particles is sufficient.*

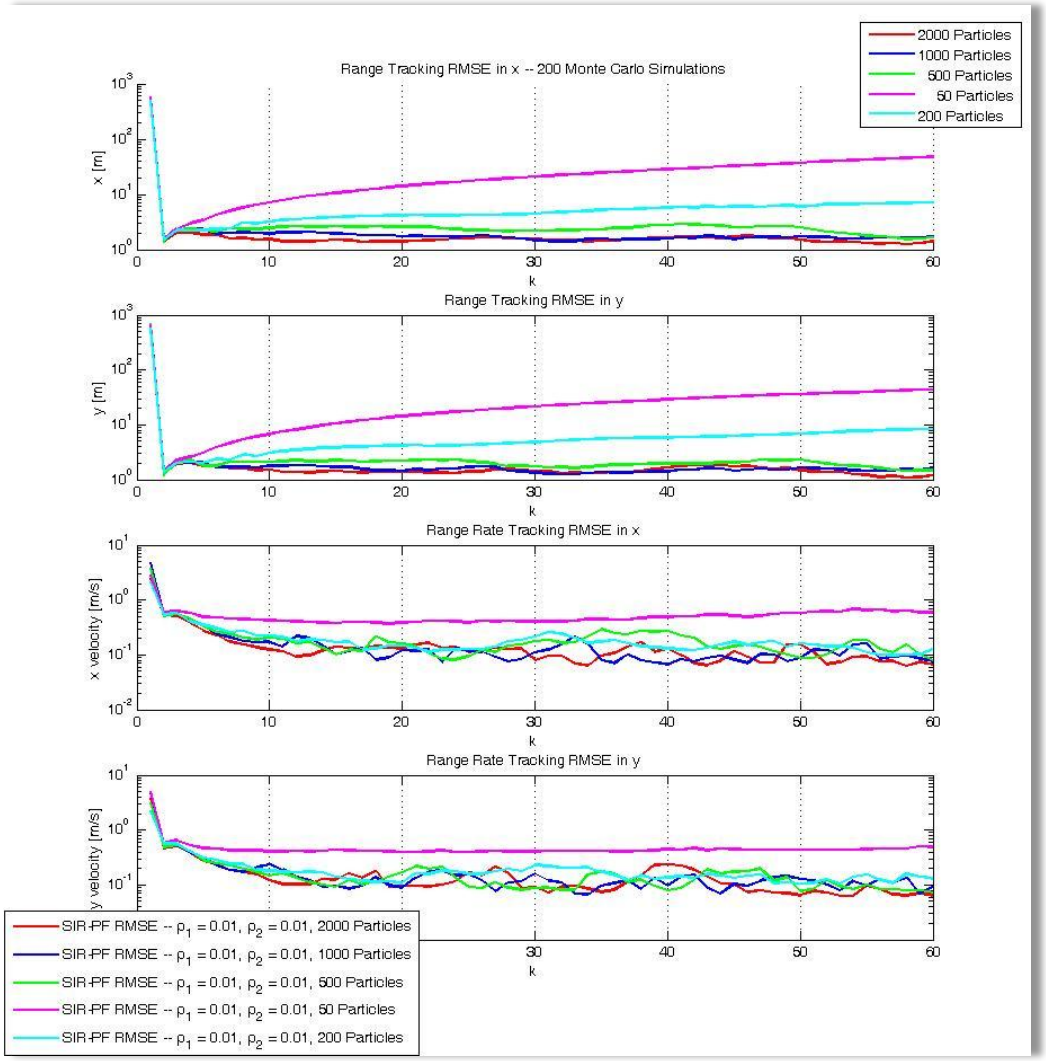


Figure 4.3 *Various RMSE results as a function of particle number. Figure shows that while there are significant decreases in error as particle number increases, the differences are likely negligible beyond 500, and definitely beyond 1000, and do not justify the additional computational costs. Therefore using 1000 particles is sufficient.*

Multiple test simulations were run using different amounts of particles; one such instance is provided. In this case the particle number was varied from between 50 to 2,000 particles. Figure 4.3 shows that performance improved significantly as the particle number is increased from 50 to 500. But a point of dimishing returns is reached between 500–1000 particles. Note that the error is (relatively) quite high in both range and velocity tracking with 50 particles, and in position tracking error it

is still high even at 200 particles. Beyond this, however, differences in performance/error appear negligible. In order to err on the conservative side, 1000 particles were used in all subsequent simulations and results throughout this paper, and in continuded research endeavors, unless specified otherwise.

With the proper number of particles in place, as confirmed above, the most accurate and ideal tracking simulations could be run. Some representative results of such simulations are provided in Figures 3 and 4. These plot the measurement estimates (green line) atop the true state (black triangles), for each of the four components in the state space. Note that the error was consistently off by only about 1 m in position, in both x and y, and approximately 0.1 m/s for both velocity components, across nearly all simulations. This level of error is more than acceptable for target tracking applications.

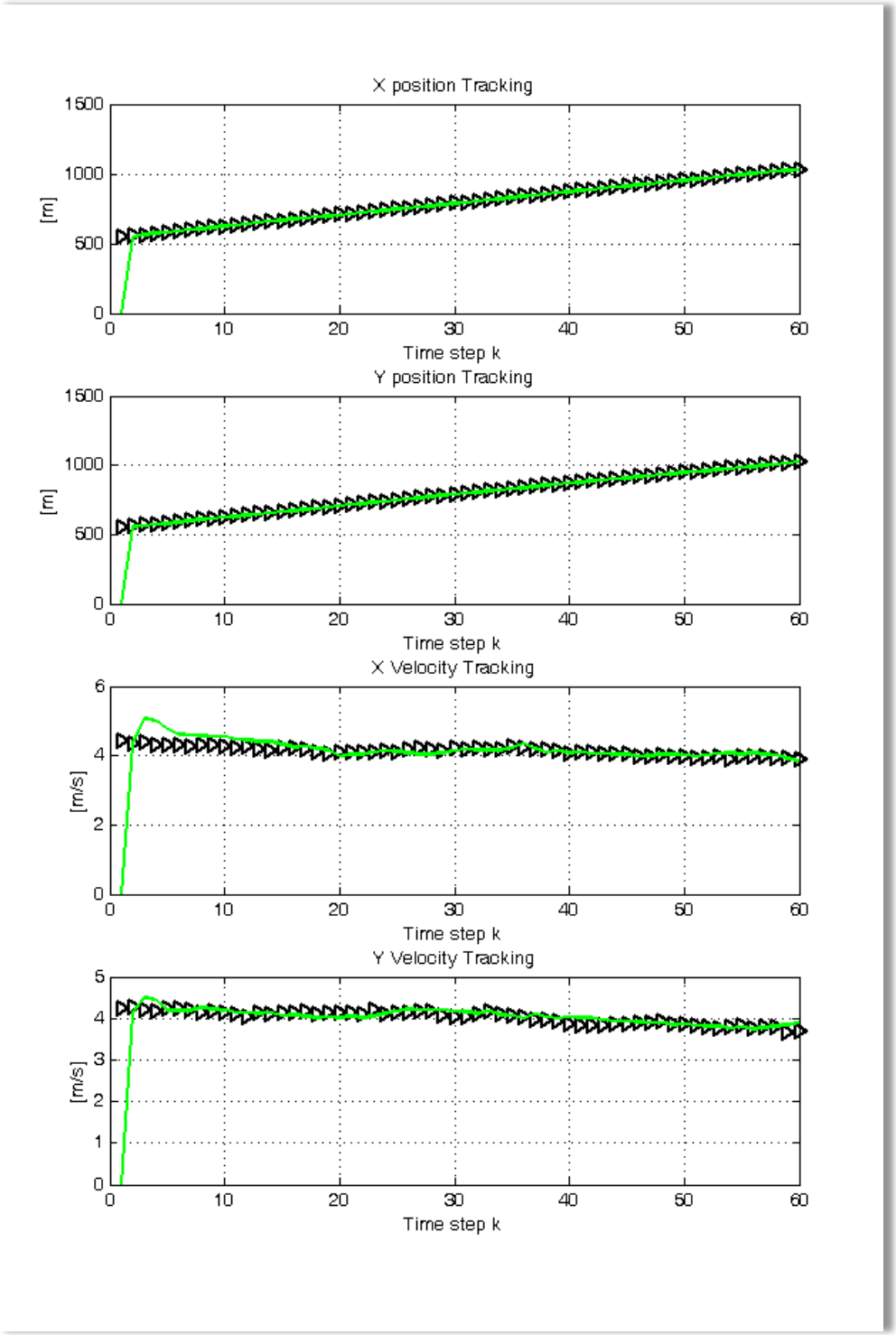


Figure 4.4 *Plots of the measurement estimates atop the true state, for each of the four components in the state space; the result of one MC simulation. A relative lack of clutter allows for exceptionally accurate tracking in some instances.*

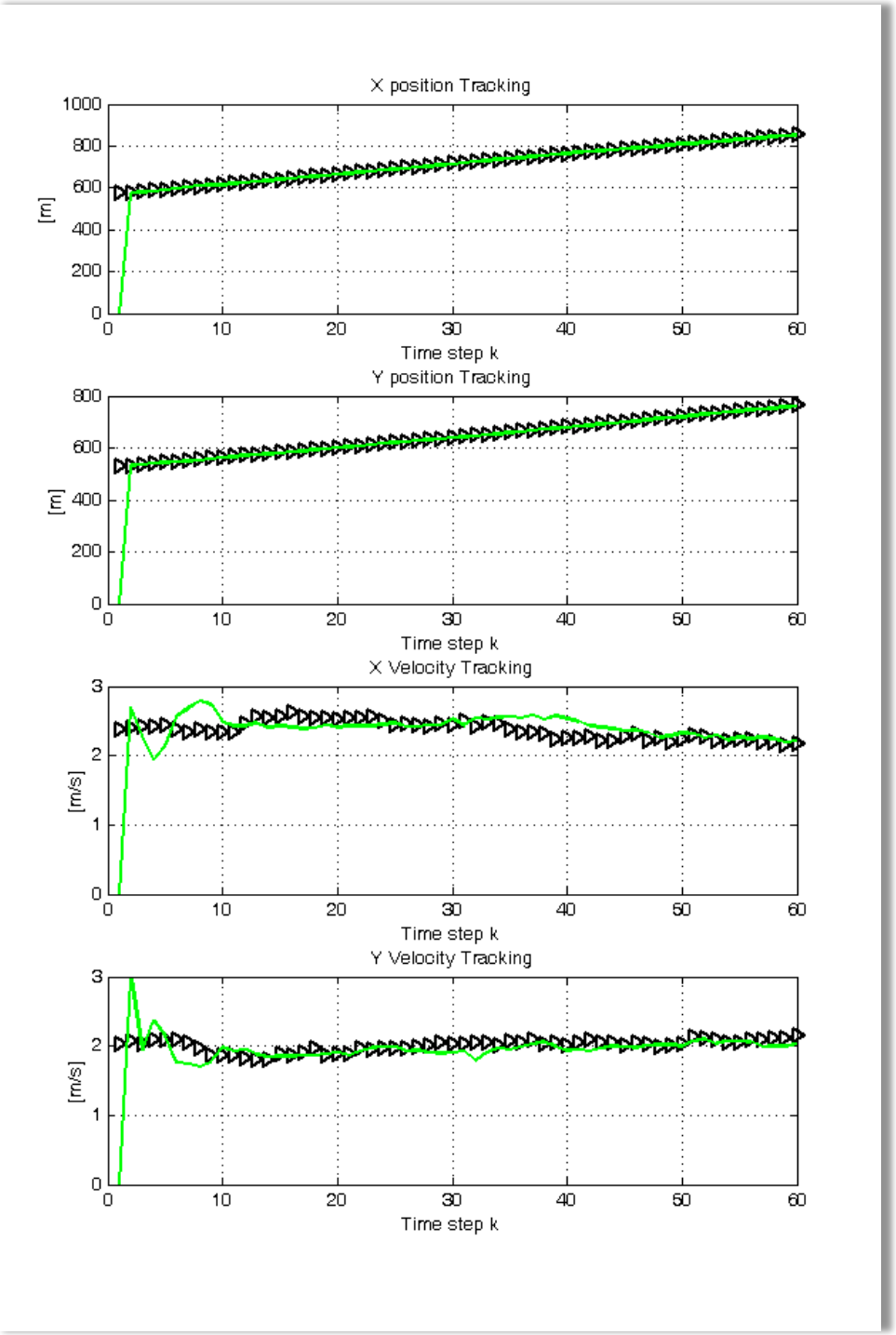


Figure 4.5 *Plots of the measurement estimates atop the true state, for each of the four components in the state space, the results of one MC simulation. The position tracking is solid (compare above Figure 3), but deviations in the velocity estimates begin to appear due to heavier clutter.*

Figure 4.1 plotted the RMSE vs. time, and Figures 4, 5 plotted the actual, physical tracking error and deviationts from the true state. The next figure type, which will be presented frequently throughout the remainder of this chapter, combines both of the aforementioned figure types, and illustrates the target's entire trajectory in state space through concentrations of clutter.

For instance, consider Figure 4.6. This diagram simultaneously presents several apsects of a tracking scenario. First, we see, within a 2-D plane, the target's true trajectory, and the estimates of its trajectory, at each time step. These combine to plot the full trajectory of the estimated and true states in the state space. In another layer, the plot also shows the relative amount of clutter through which the target passes. This particular figure shows two different clutter regions, one much more dense than the first. (In general, the amount of clutter plotted is not literal, but a scaled version of the relative clutter disparity between the regions.) In this case we also see that the drastic change in clutter – coupled with the high degree of clutter in the second region – only causes the tracking algorithm to deviate slightly from the target's true state, near the end of the simulation.

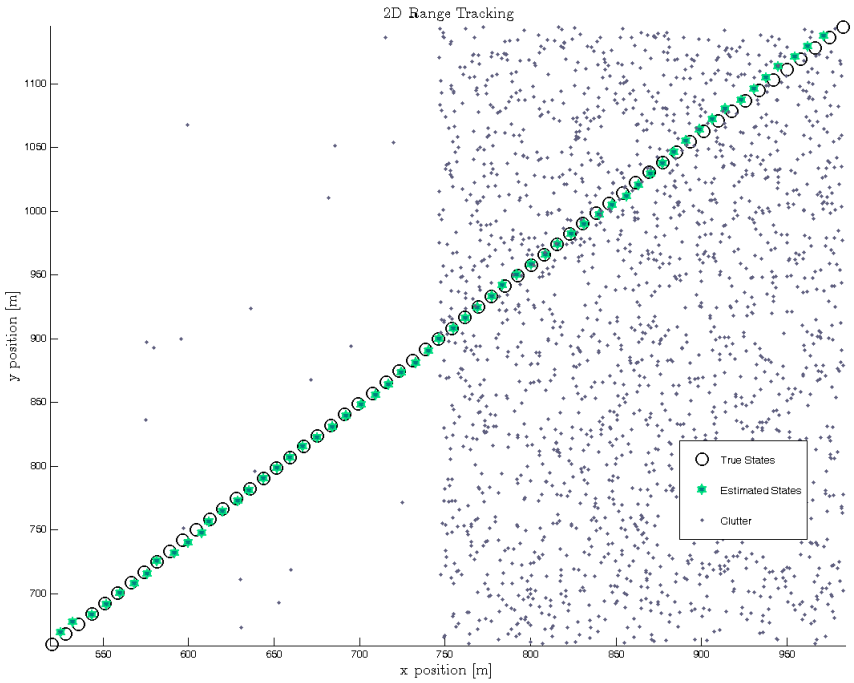


Figure 4.6 *A target's trajectory, with the calcuated state estimates, crossing through two regions of varying clutter.*

Figure 4.6 could be used to represent a tracking scenario either with the PF or the IMM; the true state, estimated state, and clutter densities are independent of the tracking algorithm used. However, the next figure type (which correlates with Fig. 6), is exclusive to the IMM, and is only generated when this algorithm is implemented.

The first of many of these is Figure 4.7, which pairs with Figure 4.6. This indicates that the IMM algorithm does well – meaning on average particles transition to the correct model during the right time epoch – when the clutter densities of the different regions are sufficiently different. This is also demonstrated in the first five mode probabililty plots, Figures 4.7 – 4.11. They outline the

transition that occurs from when the densities differ significantly, to when they are equal.

Staring in Figure 4.7, we see that there is a sharp and instantaneous transition. This indicates that the algorithm is adapting perfectly, switching on cue and selecting the correct model at the correct time based on the clutter variations it encounters. The model can easily adapt to the drastic change in ρ, making a sharp and unanimous model transition almost instantaneously.

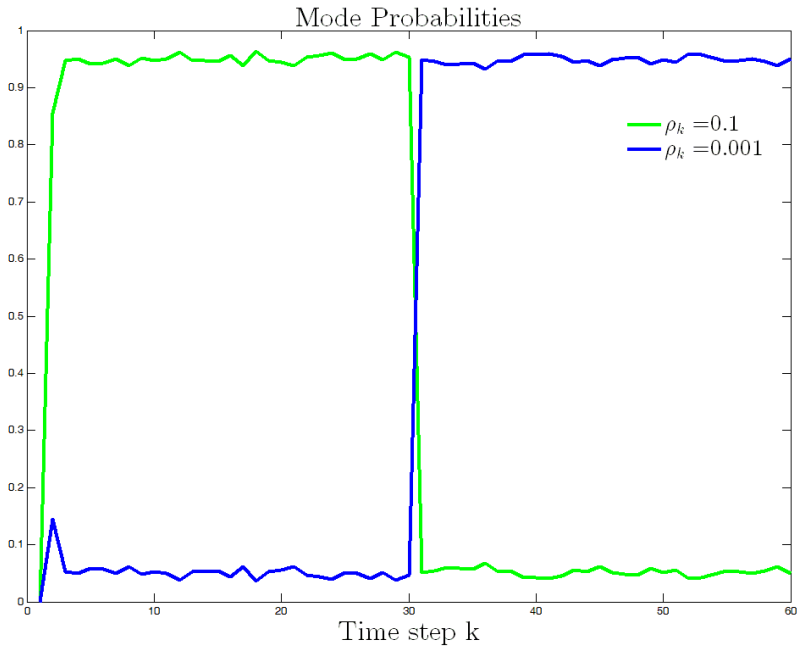


Figure 4.7 *Plot of mode transition probabilities over time, for vastly different clutter densities, for 1 Monte Carlo Simulation. The model can easily adapt to the drastic change in* ρ *, making a sharp and unanimous model transition almost instantaneously.*

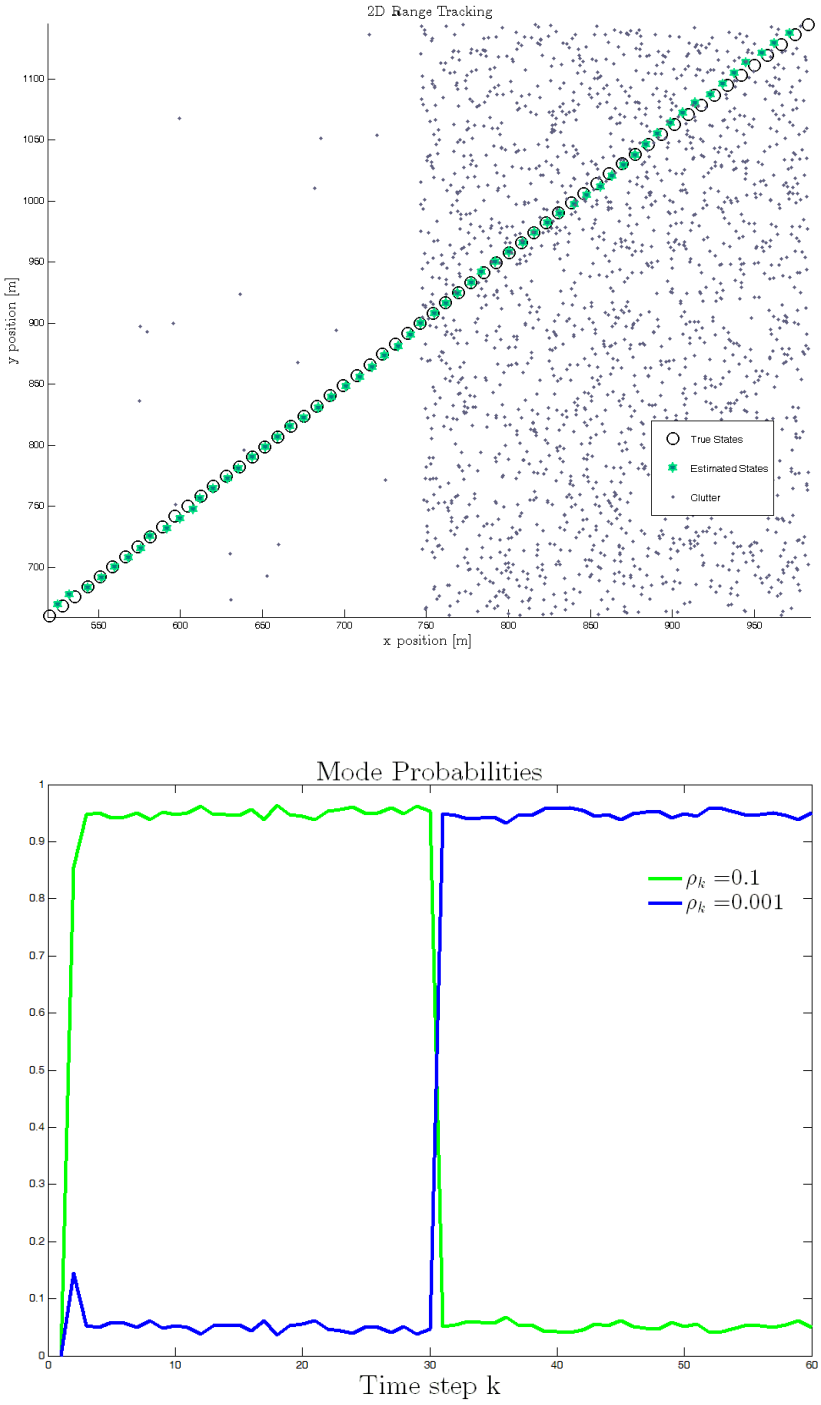


Figure 4.6, 4.7 *A Combining Figures 6 and 7 to show the correlation between clutter variaion and mode transition. The color variation indicates that the model switches modes precisely when transitioning clutter regions.*

Figure 4.8 illustrates the behavior of the IMM PF for clutter densities that differ by a small amount. In such cases, the IMM is still able to transition models, though the correct model is not weighted as heavily, given the ambiguity that follows from a non-drastic clutter differential. If we continue this trend, and bring the clutter density values closer together, then make them equal (as in Figures. 4.8 – 4.10), the model eventually, invetably, does not switch at all; the model transition rate approaches an ambiguous 50% as $\rho_1 \to \rho_2$.

The inherrent symmetry in the graphs affirm that all probabilities sum to unity at all times, as expected/required.

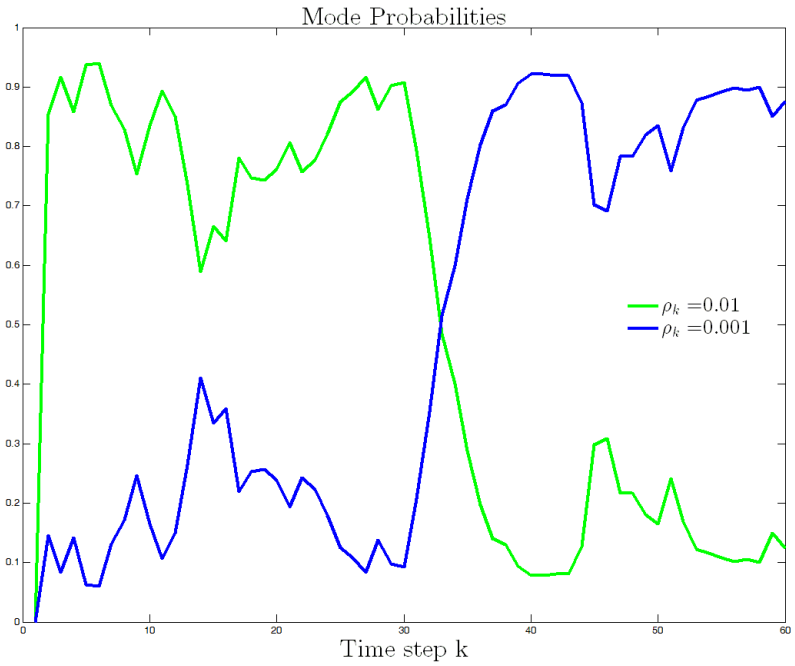


Figure 4.8 *Plot of mode transition probabilities over time, for clutter densities that differ by a small amount, 1 MC Simulation. The IMM is still able to transition models.*

If we hold both ρ values constant across several randomized Monte Carlo runs, we see that these fluctauations tend to smooth over repeated runs, and that, on average, the IMM tends toward assigning appropriate weights to the most suitable model. In these following (three) figures, we have ommitted plots of range tracking, as these do not provide any noteworthy or additional information, since the equal or nearly equal values of ρ imply a near uniform clutter dispersement throughout the state space.

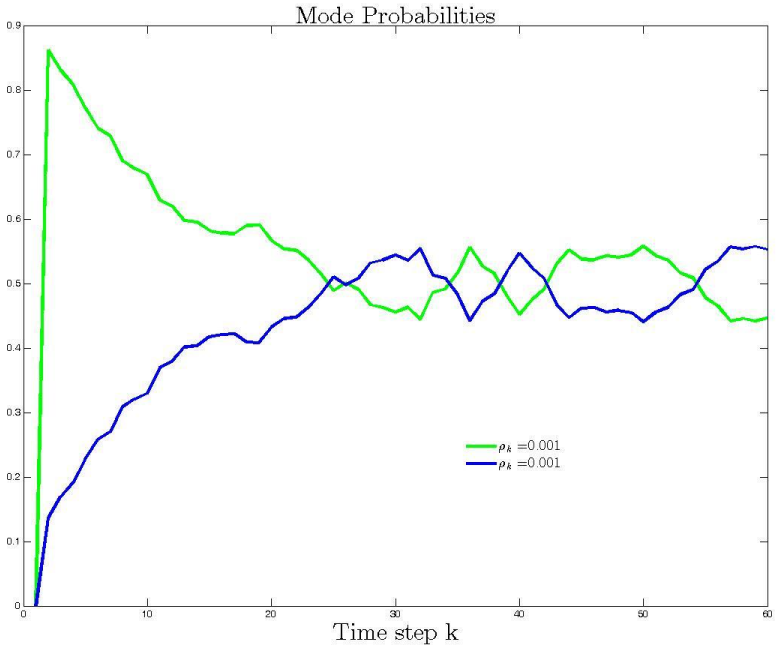


Figure 4.9 *Plot of mode transition probabilities over time, for equal clutter densities, for 1 Monte Carlo Simulation. The values oscillate around a 50% transition probability, as expected. The probabilities approach this ratio as rho's approach equality.*

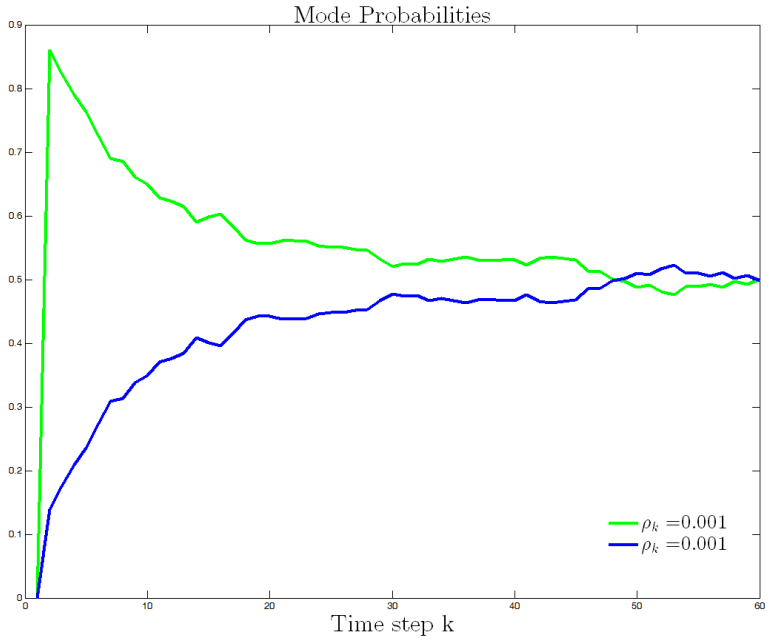


Figure 4.10 *Plot of mode transition probabilities over time, for equal clutter densities for 20 Monte Carlo Simulations. The values oscillate around a 50% transition probability, as expected.*

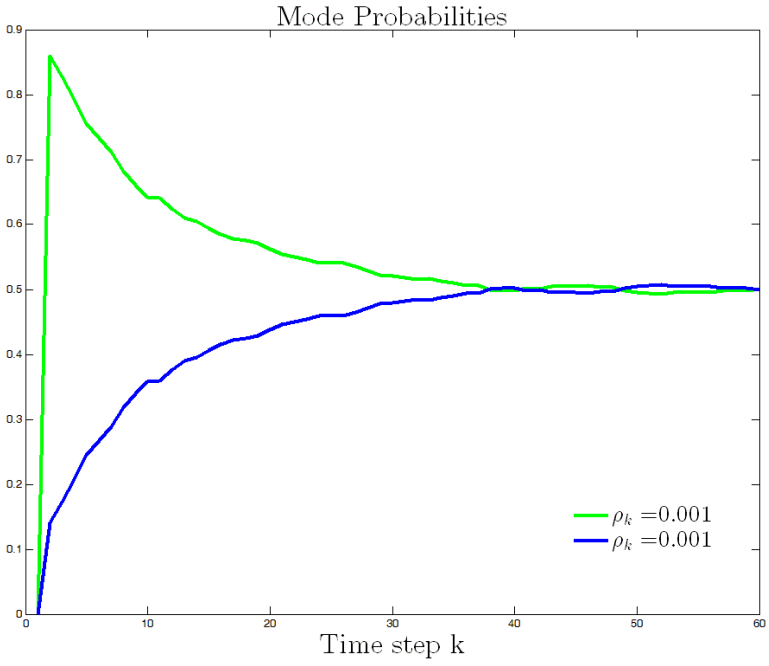


Figure 4.11 *Plot of mode transition probabilities over time, for equal clutter densities, after 200 Monte Carlo Simulations. The values oscillate around a 50% transition probability, as expected. Compare with the lone MC simulation of* ***Fig. 4.9***

Note that these cases do not imply that the IMM PF performs poorly; rather, it tracks accurately in all of these instances. This only means that clutter densities are so close it simply does not matter which model the IMM selects – the perforamnce will be approximately the same either way given the similarity of the clutter frequencies.

Note, also, that regardless of the number of Monte Carlo simulations performed, or the amount of the "smoothless" (or lack thereof), the mode probablities always sum to unity, at each time step, in all figures – as required.

Our tour through the aforementioned figures has explored and verified that the IMM PF functions correctly in the more "trivial" cases where clutter variation is negligible or nonexistent. We now proceed to the cases of greater complexity, as one of the major research goals was to show that the IMM PF could perform well in more complicated scenarios of dynamically changing clutter. This is first done by introducing the addition of a third, and followed by a fourth region, into the tracking environment, each with it's own clutter density (which may or may not be equal to that of the others). Figures 4.12 – 4.19 take us on a tour through various tracking scenarios of increased complexity, and further applicability. See the titles of each page or plot pair for a detailed description. Note that in cases where the target is initialized amidst heavy clutter, tracking performance decreases, causing deviations, and introducing errors. The IMM algorithm still performs well in spite of this. And the following plots show that with the IMM, tracking and mode transitions occur accurately and timely throughout a gauntlet of diverse test trials.

Transition Through 3 Different Clutter Regions

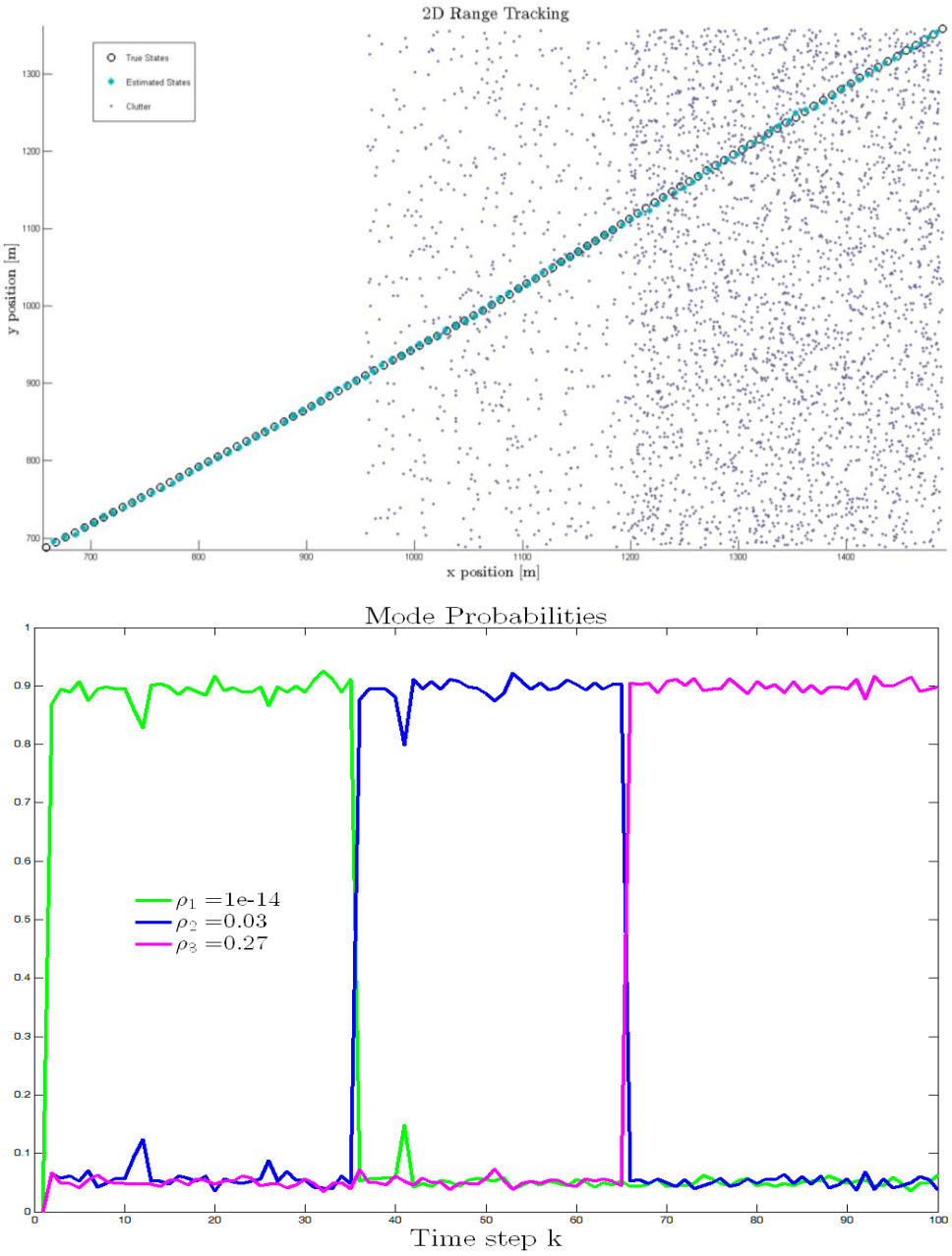


Figure 4.12 *State space plot (top) showing a target passing through a total of three regions, each with a very different amount of clutter, with the IMM switching models in sync (bottom). The exact numberical difference in clutter density can be seen in this bottom Mode-Switching Probabilities figure.*

Transition Through 3 Different Clutter Regions

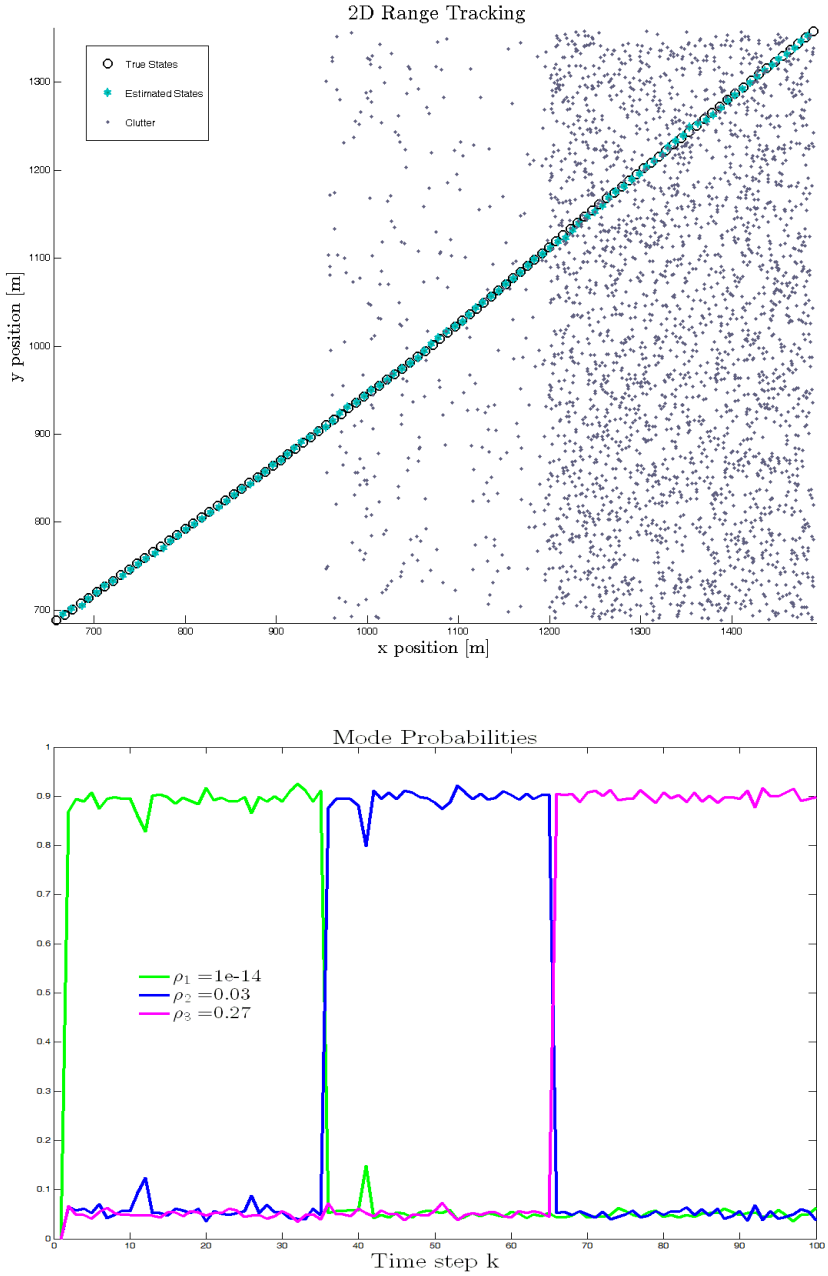


Figure 4.12 *State space plot (top) showing a target passing through three regions of exponentially ascending clutter density, with the IMM switching models in sync (bottom).*

Switching In and Out of a Cluttered Region
(Two Environments)

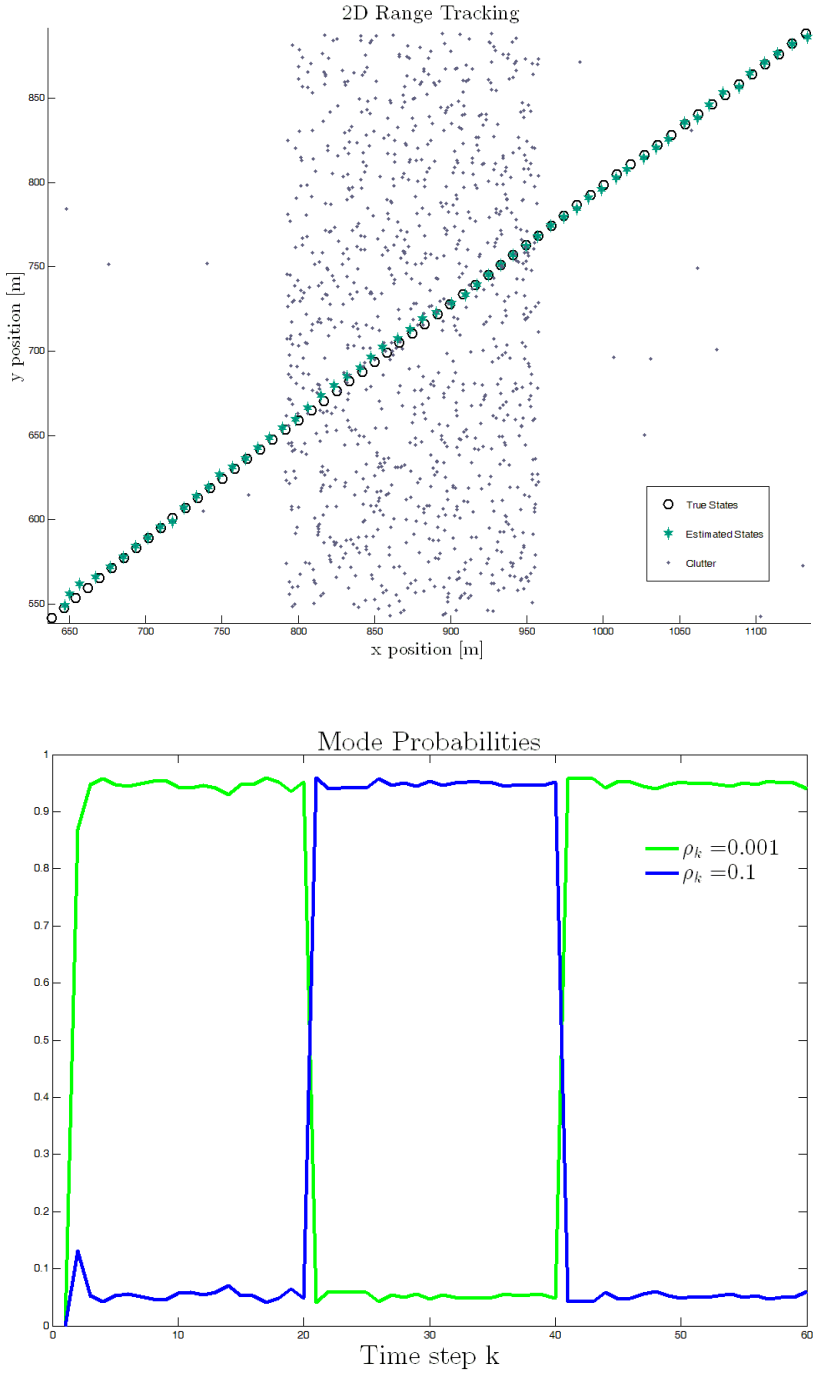


Figure 4.13 *State space plot (top) showing a target passing in and out of a high clutter area, with the IMM switching models in sync (bottom).*

Switching In and Out of a Cluttered Region
(Three Environments)

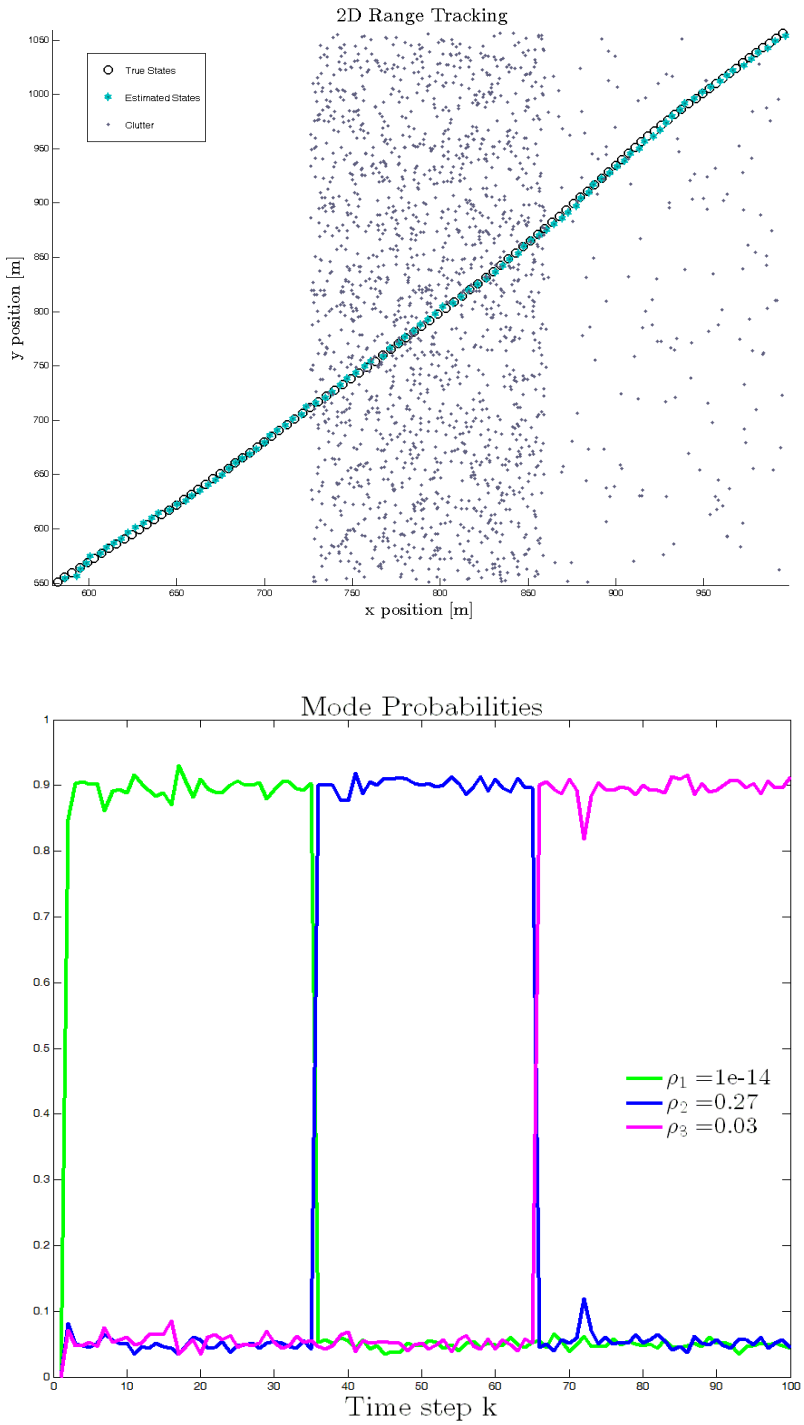


Figure 4.14 *State space plot (top) showing a target passing through a total of three regions, each with a very different amount of clutter. In the case the amount of clutter is not acsending, but peaks in the middle region. The IMM still switches models in sync (bottom).*

As with the cases of two different regions having similar clutter concentrations, when the clutter densities ρ_1, ρ_2, ρ_3 are close enough in value, model switches are inconsequential, and so the IMM converges to the steady state ratio of equal transition probabilities,

The following figures illustrate this, for cases wherein two regions have similar amounts of clutter, while a third region has a clutter density difference that is sufficiently drastic to warrant a model transition.

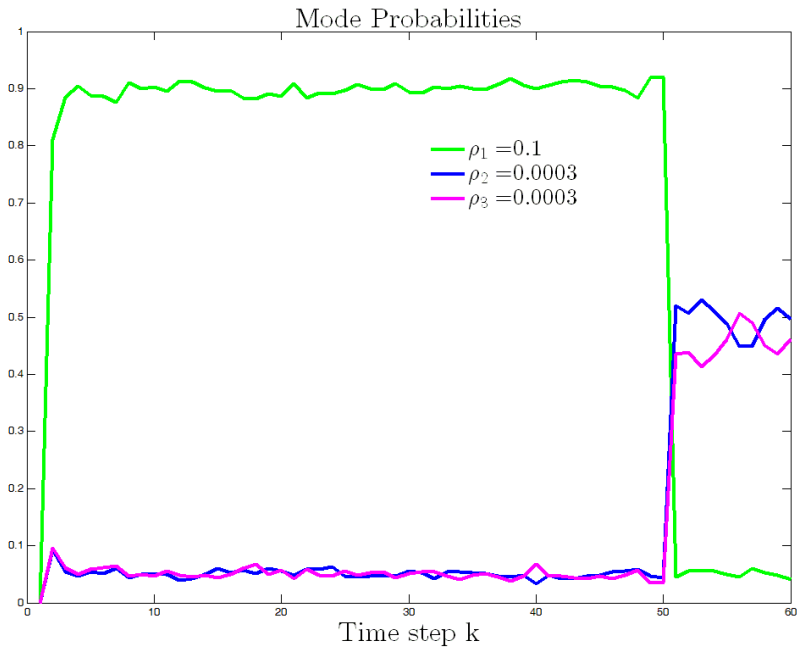


Figure 4.15 *As $\rho_1 >> \rho_2$, ρ_3, the IMM PF can easily distinguish the most optimal of the three models. Because ρ_1 and ρ_2 are equal, the algorithm oscillates around a 50% switching rate between the two.*

Figure 4.15 shows a scenario where the target spends most of its time in a region of heavy clutter, before entering one with significantly less for a brief period. Since

$\rho_1 >> \rho_2, \rho_3$ the IMM PF can easily distinguish the most optimal of the three models. Likewise, the transition from this model is immediate despite how late it occurs in the tracking scenario, which is an indicator of excellent performance for this triple model IMM. Because ρ_1 and ρ_2 are equal, the IMM's choice between the two associated models is arbitrary for the duration of the simulation. The inverse of this scenario is shown in the following figure.

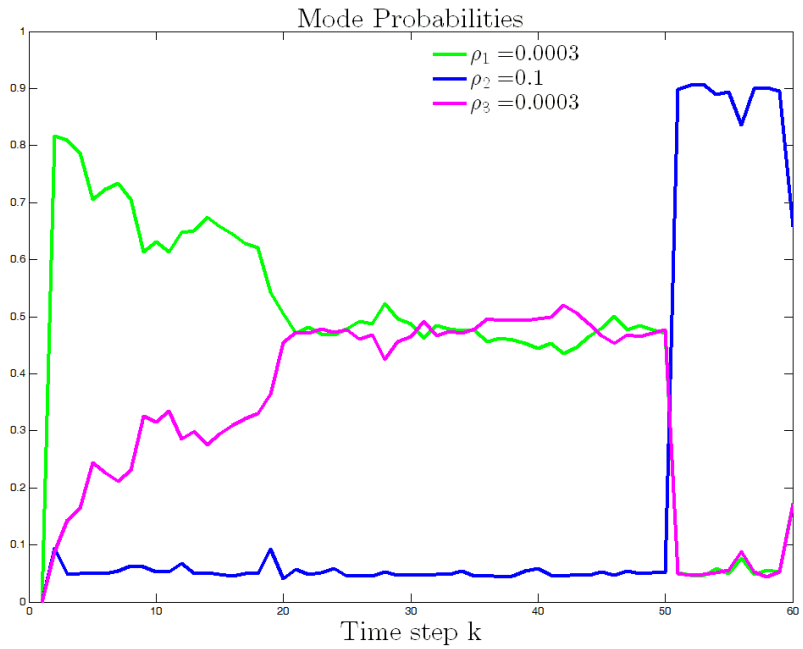
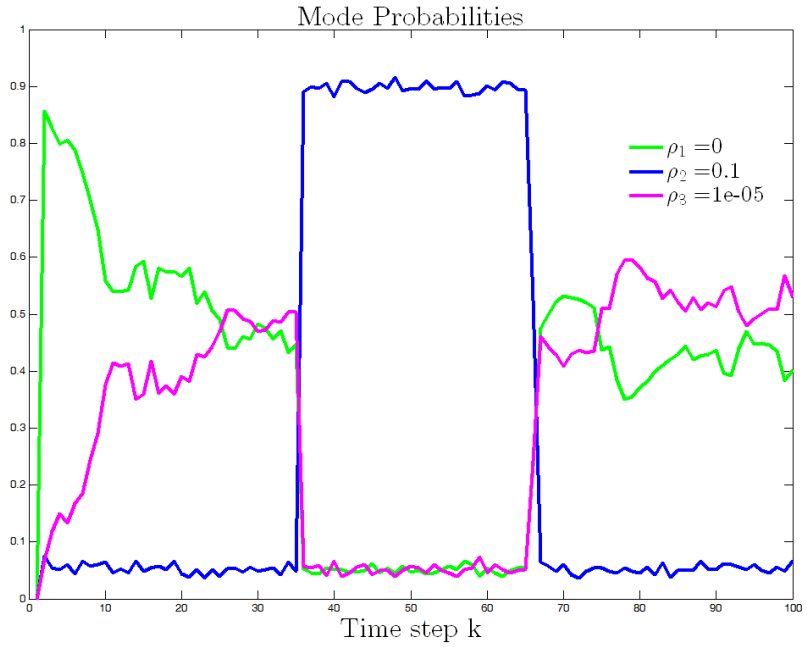


Figure 4.16 *As $\rho_2 >> \rho_1$, ρ_3, the IMM PF can easily distinguish the most optimal of the three models. Because ρ_1 and ρ_3 are equal, the algorithm oscillates around a 50% switching rate between the two.*



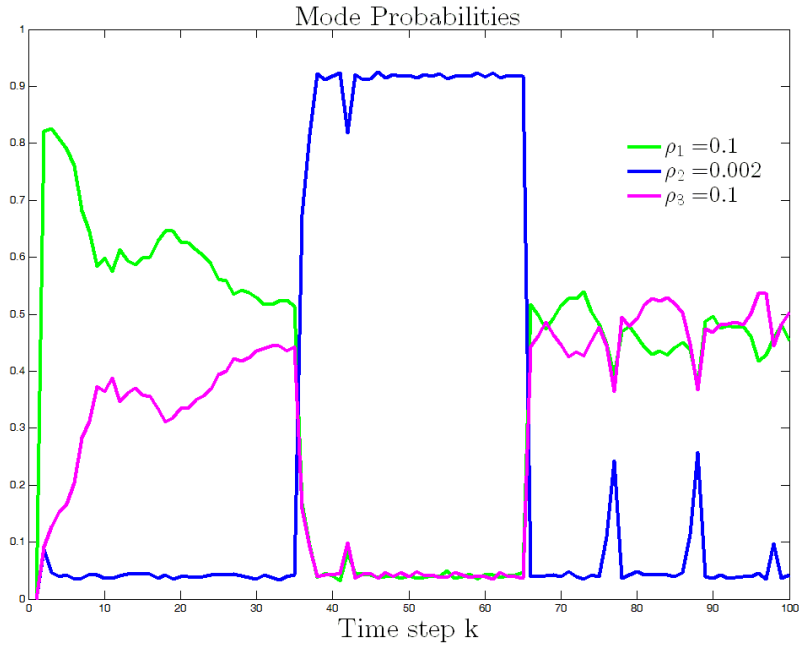


Figure 4.17*Additional examples of accurate, timely mode-switching in complicated tracking scenarios.*

4 Regions, 3 Transitions, 2 Repeated Clutter Densities

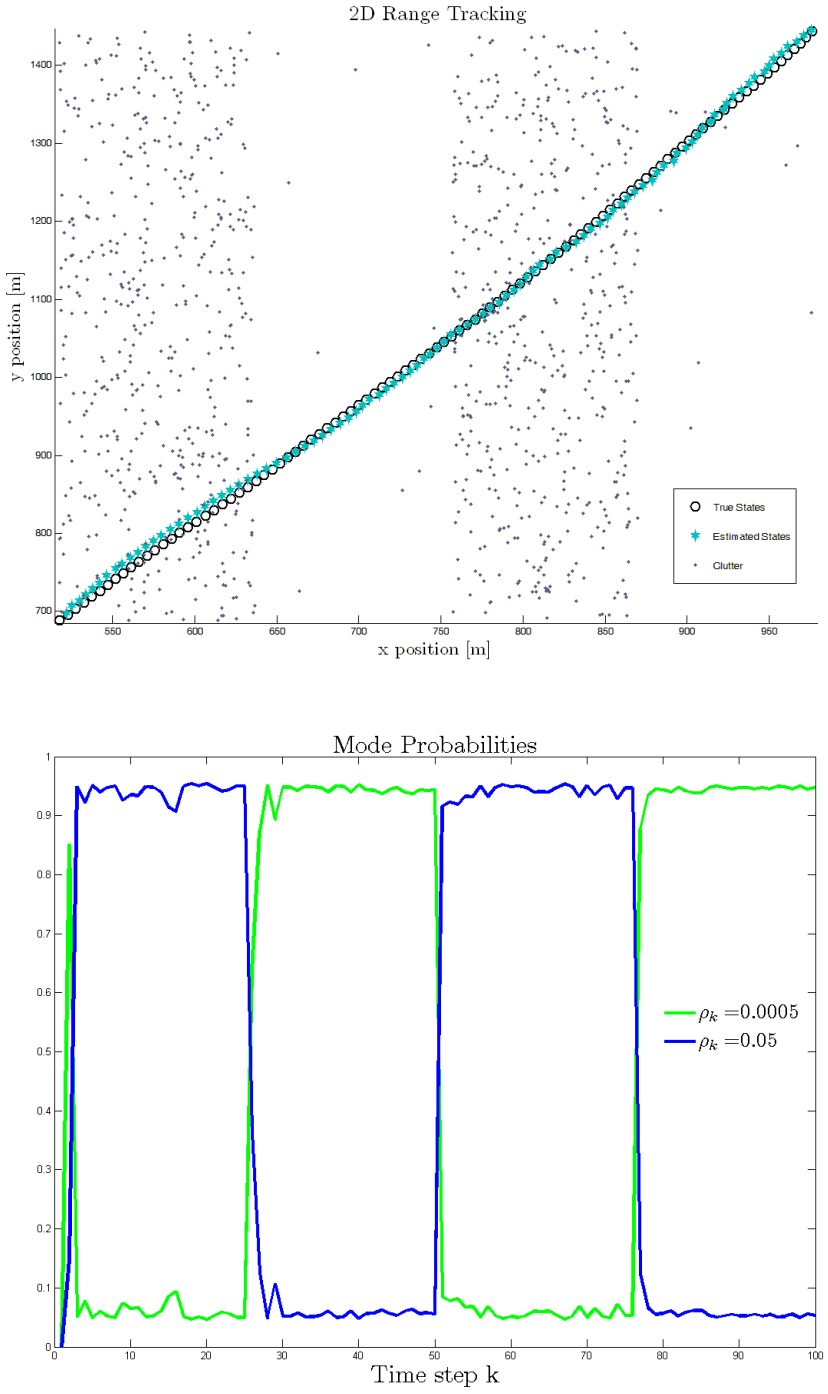


Figure 4.19 *State space plot (top) showing a target passing through a total of four regions of alternating clutter, with the IMM switching models in sync (bottom). Note that initializing the target amidst heavy clutter decreases tracking performance, causes deviations, and introduces errors. The IMM is still able to recover in spite of this.*

At this point a fair question would be how the performance of the IMM compares to that of the standard PF, and whether the improvements are significant. This is indeed the case, and simulations were run to verify this.

To prove that the IMM was indeed more efficient than the standard, classic PF, tests began with four basic PF test cases, to be referenced against two equivalent IMM test case settings. This was confined to a testing scenario with only two different clutter regions, as shown in Figures 4.6 and 4.7. Because there are two regions with different clutter densities – one high and one low – there are two distinct possible configuration arrangements. Namely, the target passes through a region of high clutter to low clutter, or vice versa. Within these two possibilities, are three further possibilities regarding the *a priori* assumptions within the PF clutter algorithm: the algorithm assumes a clutter density that is lower, higher, or equal to that of the environment. Since there are only two clutter environments being considered, this amounts to a total of four distinct possibilities (the case of assumed and actual densities being equal counts twice, once for each region). Table 1 summarizes these facts, and may be referenced for clarity.

In cases where the assumed density is equal or equivalent to the actual (or "true") clutter density, error will be low and the tracking will be accurate. Conversely, however, if the assumed clutter density does not match that of the environment, the error will be higher, and tracking accuracy will suffer. The worst cases are when the clutter is assumed to be much lower than it actually is; in these instances the error will be the highest, as the algorithm has grossly underestimated the amount of clutter and cannot compensate. If the PF does not have an assumed ρ_n value that appropriately matches the current tracking environment and the

clutter density therein, the performance can be quite poor, and failure is likely. An instance of such a case is plotted below:

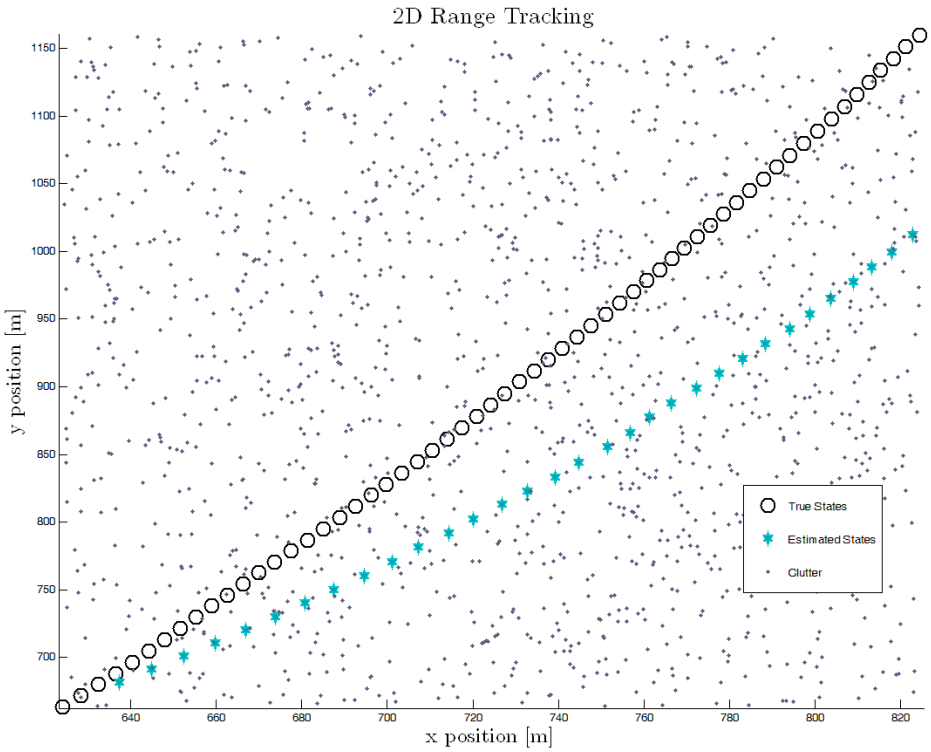


Figure 4.20 *State space plot of a tracking scenario within a region of uniform but high clutter density. With the IMM disabled, the PF's estimate of the target's true state is quickly deflected by the high concentration of clutter.*

In the cases where the clutter was assumed to be very high, but is in fact much lower, the standard PF algorithm does not perform as poorly, since it anticipated and overcompensated for a much higher amount of clutter than what was actually present. Within such a case tracking is still possible, and the error is less overall, which we have qualitatively entitled relatively "medium" error, as compared to the other two cases mentioned.

Finally, when adding the IMM into the mix, two additional possibilities are presented, each of equal error, and independent of the density chosen. This, because in this case there is no assumed environmental clutter density, since the IMM will adapt to whichever clutter density it encounters, and this illustrates the power of the added IMM algorithm. For this reason, the resultant error is qualitatively "low."

If we combine all of these facts we see there are a total of six possibilities (two of which are essentially identical in purpose). These observations are summarized in the following table, for two arbitrary clutter density values ρ_1 and ρ_2, where $\rho_2 >> \rho_1$.

Table 1 – Qualitative Performance Summary
of PF vs. IMM PF for 2 Regions

True Density	Algorithm	Assumed Density	Relative RMSE
ρ_1	PF	ρ_1	Low
	PF	ρ_2	High
	IMM	N/A Adaptable	Low
ρ_2	PF	ρ_1	Medium
	PF	ρ_2	Low
	IMM	N/A Adaptable	Low

Note that the above table and observations point out that the IMM does not always outperform the standard PF – and this is not the claim of this thesis. Indeed, there are many cases wherein the performance will be comparable. For instance, in the two–environment scenario given above, there are cases where the performance is equal at best. However, the IMM is shown to be more robust and reliable, making it the more dependable and adaptable choice in confronting a tracking objective.

The table above reflects a summary of many simulations run to show that overall, on average the RMSE of the IMM was lower than or at least equal to that of the PF. One of many such simulation instances is provided below, which was implemented considering the six possibilities outlined in Table 1 above.

Different simulation scenarios were compared using either the PF or the IMM PF, and for each scenario, the clutter density was fixed to one value throughout the track or it varies with time between two values during different time frames of the track. In particular, the different scenarios are: (a) PF with knowledge of true density, either ρ_1 or ρ_2; (b) PF using clutter density ρ_1 when the true density was ρ_2; (c) PF using clutter density ρ_2 when the true density was ρ_1; (d) IMM PF with only one clutter density, either ρ_1 or ρ_2, over the track duration. The two clutter densities were selected to be ρ_1=0.0001 and ρ_2=0.01. The performance comparison results are demonstrated in Figure 4.21. Each of the tracking scenario simulations below were the result of 20 Monte Carlo trials.

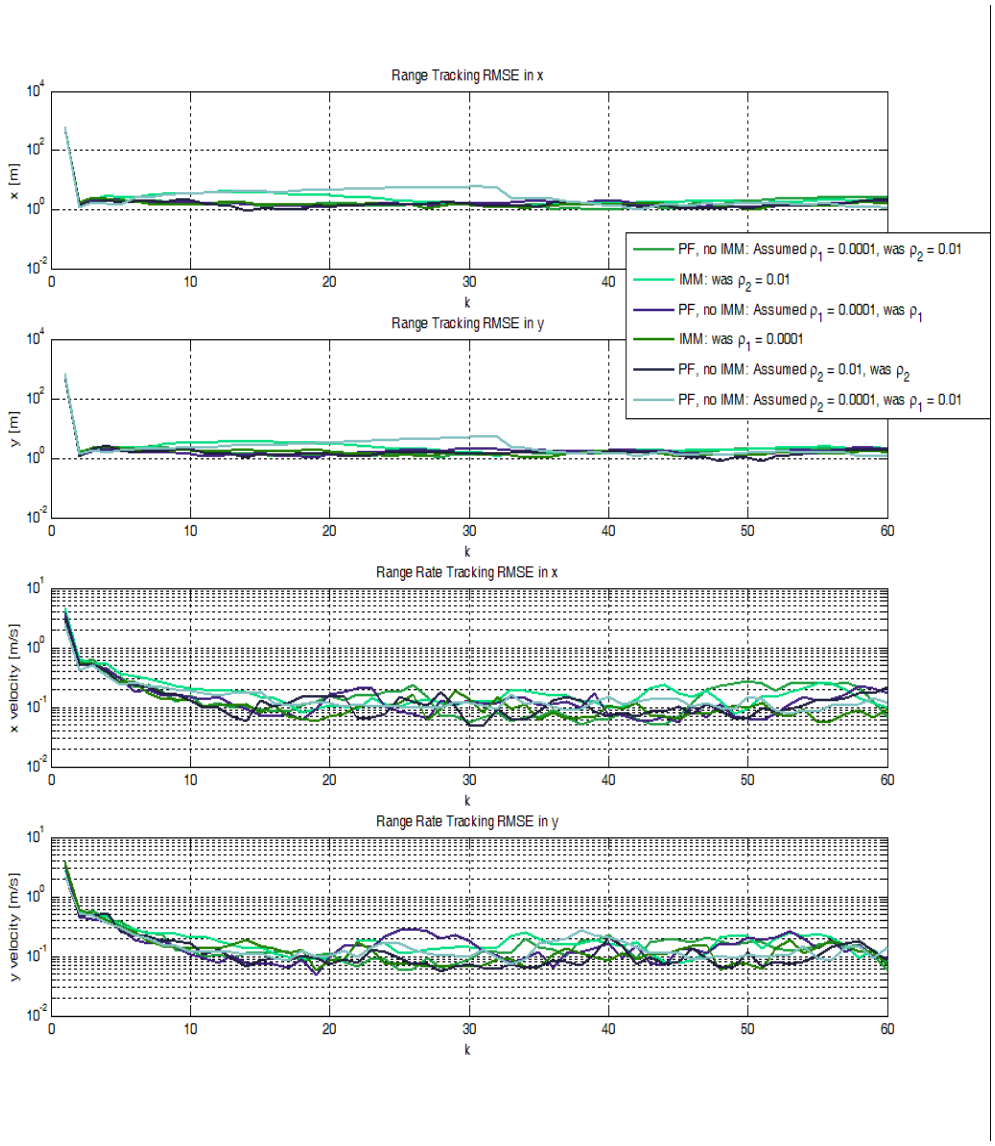


Figure 4.21 *Performance comparison of PF vs. IMM PF, plotting RMSE values for each algorithm, from simulations with different assumed vs. actual clutter densities. Each of the tracking scenario simulations above are the result of 20 Monte Carlo trials.*

As before, the scenarios above are extended to three regions instead of just two. The results and underlying themes are parallel, though the details shed more insight on overall performance. These observations are summarized in the following table, for three arbitrary clutter density values ρ_1, ρ_2, and ρ_3, where $\rho_3 >> \rho_2 >> \rho_1$.

Table 2 – Qualitative Performance Summary
of PF vs. IMM PF for 3 Regions

True Density	Algorithm	Assumed Density	Relative RMSE
ρ_1	PF	ρ_1	Low
	PF	ρ_2	High
	PF	ρ_3	High
	IMM	N/A – Adaptable	Low
ρ_2	PF	ρ_1	Medium
	PF	ρ_2	Low
	PF	ρ_3	High
	IMM	N/A – Adaptable	Low
ρ_3	PF	ρ_1	Medium
	PF	ρ_2	Medium
	PF	ρ_3	Low
	IMM	N/A – Adaptable	Low

As mentioned above, though the overarching patterns are parallel, this table illustrates a very important point: that the IMM becomes more stable, reliable, and accurate compared to the standard PF as the number of clutter density regions increases. We saw in the previous table, that there was only a 50% chance (which is still high) of assuming the "wrong" rho, and suffering increased error; and in one instance (ρ_2), the worst possibility (assuming ρ_2 when $\rho = \rho_1$) was only a "medium" error rate compared to the relatively "low" IMM error rate. However, with the addition of another region – equivalent to another dimension in the problem – this

quickly changes. For any given region, the lone PF, ill-equipped without the addition of the IMM algorithm, essentially only has a 33% chance of achieving the low error rate as the IMM PF (whereas before, with only two different clutter regions, the probability for equaling IMM performance was 50%). Moreover, the penalty and error disparity for a wrong choice is much worse: consider the case of ρ_1, where an incorrect assumption results in a 2/3 chance of "high" error compared to the low error of the IMM. The IMM's results remain predictable and stable – at a minute additional computational cost – whereas the performance of the PF varies widely. And from the above two tables it is clear that this pattern only continues and amplifies; the PF will perform worse and the IMM PF will outperform as dimensions of different clutter are added to the tracking scenario.

Finally, it is again necessary to emphasize that in the cases of "low" relative error, the results are not comparable in these instances because the IMM does worse, but because there is a negligible difference in clutter concentration, and thus model transitioning is inconsequential. In these cases the IMM PF essentially reduces to the standard PF, so there is really no basis for comparison or possibility for outperformance; both algorithms will perform the same in these scenarios.

Chapter 5

CONCLUSIONS AND FUTURE WORK

The simulation results in Chapter 4 strongly support the utilization of the new IMM PF. It has been shown to be both robust and rapid in the contexual applications we explored. The versality of having multiple selectable models allows for much more adaptibility, and the capacity to handle greater amounts of clutter overall. It also increases options for many possibilities and applications.

Future work in this area will include, first and foremost, the incorporation of acceleration into the model. This will allow us to explore target turning, and having the target maneuver in and out of various regions (transitions). Also, this would imply having additional IMM algorithms, simultanesouly utilizing mutliple IMMs (as the transition from constant velocity to turning requires an IMM algorithm).

Additionally, another possibility is using more than three clutter regions and clutter densities. It is unclear whether four or more clutter concentrations could be assigned values that are far enough apart to allow the IMM PF to make the right transitions, but there would be no drawbacks to using this algorithm in these cases, and it would certainly be interesting to investigate. A corollary exploration would involve assessing the way the computational costs of the algorithm scale with increasing model number, which could prove interesting.

Finally, we believe that the IMM PF can be optimized by exploring the precise transition probabilities between models within the algorithm. These parameter configurations were not configured extensively, but by doing so it is possible the algorithm could become more sensitive, and thus more able to outperform the non IMM PF in an even wider range of scenarios and applications, with the switching becoming even more rapid.

REFERENCES

Arulampalam, M., Maskell, S., Gordon, N. and Clapp, T. (2002). A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Trans. Signal Process.*, 50(2), pp.174-188.

Bar-Shalom, Y. (1990). Tracking and Data Association. *J. Acoust. Soc. Am.*, 87(2), p.918.

Bar-Shalom, Y. (2001). Kalman filtering techniques for radar tracking. *Automatica*, 37(6), pp.957-958.

Bar-Shalom, Y. and Tse, E. (1975). Tracking in a cluttered environment with probabilistic data association.*Automatica*, 11(5), pp.451-460.

Bar-Shalom, Y., Daum, F. and Huang, J. (2009). The probabilistic data association filter. *IEEE Control Systems Magazine*, 29(6), pp.82-100.

Bates, M., Simmons, J. and Zorikov, T. (2011). Bats Use Echo Harmonic Structure to Distinguish Their Targets from Background Clutter. *Science*, 333(6042), pp.627-630.

Blom, H. and Bar-Shalom, Y. (1988). The interacting multiple model algorithm for systems with Markovian switching coefficients. *IEEE Transactions on Automatic Control*, 33(8), pp.780-783.

Boers, Y. and Driessen, H. (2002). Hybrid state estimation: a target tracking application. *Automatica*, 38(12), pp.2153-2158.

Boers, Y. and Driessen, J. (2003). Interacting multiple model particle filter. *IEE Proceedings - Radar, Sonar and Navigation*, 150(5), p.344.

Bojilov, L., Alexiev, K. and Konstantinova, P. (2002). An Accelerated IMM-JPDA Algorithm for Tracking Multiple Maneuvering Targets in Clutter. *Information & Security: An International Journal*, 9, pp.141-153.

Bourgeois, B. (2007). *Using Range and Range Rate for Relative Navigation*. Ft. Belvoir: Defense Technical Information Center.

Bourgeois, B. and McDowell, P. (2004). Intervessel navigation using range and range rate. *J. Acoust. Soc. Am.*, 115(5), p.2616.

Candy, J. (2007). Bootstrap Particle Filtering. *IEEE Signal Process. Mag.*, 24(4), pp.73-85.

Candy, J. (2009). *Bayesian signal processing*. Hoboken, N.J.: Wiley.

Chen, B. and Tugnait, J. (2001). Tracking of multiple maneuvering targets in clutter using IMM/JPDA filtering and fixed-lag smoothing. *Automatica*, 37(2), pp.239-249.

Cohen, L. (1995). *Time frequency analysis*. Englewood Cliffs, NJ: Prentice Hall.

Dasgupta, A. and Raftery, A. (1998). Detecting Features in Spatial Point Processes with Clutter via Model-Based Clustering. *Journal of the American Statistical Association*, 93(441), pp.294-302.

Doucet, A., De Freitas, N. and Gordon, N. (2001). *Sequential Monte Carlo methods in practice*. New York: Springer.

Edla, S., Kovvali, N. and Papandreou-Suppappola, A. (2014). Electrocardiogram Signal Modeling With Adaptive Parameter Estimation Using Sequential Bayesian Methods. *IEEE Trans. Signal Process.*, 62(10), pp.2667-2680.

El-Shenawee, M. and Rappaport, C. (2002). Monte Carlo simulations for clutter statistics in minefields: AP-mine-like-target buried near a dielectric object beneath 2-D random rough ground surfaces. *IEEE Trans. Geosci. Remote Sensing*, 40(6), pp.1416-1426.

Fortmann, T., Bar-Shalom, Y., Scheffe, M. and Gelfand, S. (1985). Detection thresholds for tracking in clutter--A connection between estimation and signal processing. *IEEE Transactions on Automatic Control*, 30(3), pp.221-229.

Guo, R., Qin, Z., Li, X. and Chen, J. (2008). Interacting Multiple Model Particle-type Filtering Approaches to Ground Target Tracking. *JCP*, 3(7).

Gupta, I. and van der Merwe, A. (2000). A novel signal processing technique for clutter reduction in GPR measurements of small, shallow land mines. *IEEE Trans. Geosci. Remote Sensing*, 38(6), pp.2627-2637.

Ho, K. and Gader, P. (2002). A linear prediction land mine detection algorithm for hand held ground penetrating radar. *IEEE Trans. Geosci. Remote Sensing*, 40(6), pp.1374-1384.

Kirubarajan, T. and Bar-Shalom, Y. (2004). Probabilistic Data Association Techniques for Target Tracking in Clutter. *Proc. IEEE*, 92(3), pp.536-557.

Kirubarajan, T., Bar-Shalom, Y., Pattipati, K. and Kadar, I. (2000). Ground target tracking with variable structure IMM estimator. *IEEE Trans. Aerosp. Electron. Syst.*, 36(1), pp.26-46.

Kosuge, Y. and Matsuzaki, T. (2002). The gate size estimation method and the optimal gate shape for target tracking. *Electronics and Communications in Japan (Part III: Fundamental Electronic Science)*, 85(5), pp.10-22.

Kovalenko, V., Yarovoy, A. and Ligthart, L. (2007). A Novel Clutter Suppression Algorithm for Landmine Detection With GPR. *IEEE Trans. Geosci. Remote Sensing*, 45(11), pp.3740-3751.

Mazor, E., Averbuch, A., Bar-Shalom, Y. and Dayan, J. (1998). Interacting multiple model methods in target tracking: a survey. *IEEE Trans. Aerosp. Electron. Syst.*, 34(1), pp.103-123.

Missaoui, O., Frigui, H. and Gader, P. (2011). Land-Mine Detection With Ground-Penetrating Radar Using Multistream Discrete Hidden Markov Models. *IEEE Trans. Geosci. Remote Sensing*, 49(6), pp.2080-2099.

Musicki, D. and Suvorova, S. (2008). Tracking in clutter using IMM-IPDA-based algorithms. *IEEE Trans. Aerosp. Electron. Syst.*, 44(1), pp.111-126.

Park, K., Park, S., Kim, K. and Ko, K. (2013). Multi-Feature Based Detection of Landmines Using Ground Penetrating Radar. *PIER*, 134, pp.455-474.

Richards, M., Scheer, J. and Holm, W. (2010). *Principles of modern radar*. Raleigh, NC: SciTech Publishing.

Ristic, B. and Arulampalam, S. (2004). *Beyond the Kalman filter*. Boston, MA: Artech House.

Rong Li, X. and Jilkov, V. (2003). Survey of Maneuvering Target-Tracking . Part I: Dynamic Models. *IEEE Trans. Aerosp. Electron. Syst.*, 39(4), pp.1333-1364.

Rubinstein, R. (1981). *Simulation and the monte carlo method*. New York: Wiley.

Ruixin Niu, Willett, P. and Bar-Shalom, Y. (2001). Matrix CRLB scaling due to measurements of uncertain origin. *IEEE Trans. Signal Process.*, 49(7), pp.1325-1335.

Sangston, K. (1988). *Coherent Detection of Radar Targets in K-Distributed, Correlated Clutter*. Report 9130. Washington, D.C.: Naval Research Laboratory.

Schneider, A. (1985). Observability of Relative Navigation Using Range-Only Measurements. *IEEE Trans. Aerosp. Electron. Syst.*, AES-21(4), pp.569-581.

Sigalov, D., Michaeli, T. and Oshman, Y. (2012). Tracking a Splitting Target in Clutter Using the IMM Methodology. In: *IEEE 27-th Convention of Electrical and Electronics Engineers in Israel*. Haifa, Israel: Technion - IIT.

Stewart, J. (2003). *Calculus*. Belmont, CA: Thomson/Brooks/Cole.

Takahashi, K., Igel, J. and Preetz, H. (2011). Clutter Modeling for Ground-Penetrating Radar Measurements in Heterogeneous Soils. *IEEE J. Sel. Top. Appl. Earth Observations Remote Sensing*, 4(4), pp.739-747.

Tugnait, J. (2004). Tracking of multiple maneuvering targets in clutter using multiple sensors, IMM, and JPDA coupled filtering. *IEEE Trans. Aerosp. Electron. Syst.*, 40(1), pp.320-330.

Van Trees, H. (2004). *Detection, Estimation, and Modulation Theory, Part III, Radar-Sonar Signal Processing and Gaussian Signals in Noise*. New York, NY: John Wiley & Sons.

Vasuhi, S., Vaidehi, V. and P. R, M. (2009). Multiple Maneuvering Targets Tracking Using Kalman and Real-Time Particle Filter A Comparison. *International Journal of Engineering and Technology*, 1(3), pp.224-230.

Xuan Feng, Sato, M., Yan Zhang, Cai Liu, Fusheng Shi, and Yonghui Zhao, (2009). CMP Antenna Array GPR and Signal-to-Clutter Ratio Improvement. *IEEE Geosci. Remote Sensing Lett.*, 6(1), pp.23-27